Assessing and Mitigating Bit-Level Preservation Risks

NDSA Infrastructure Working Group
INTRODUCTION:

A Framework for Addressing Bit-Level Preservation Risk

Mark Evans
<mark.evans@tessella.com>
Digital Archiving Practice Manager,
Tessela Inc.

Micah Altman
<Micah_Altman@alumni.brown.edu>
Director of Research, MIT Libraries
Threats to Bits

- Physical & Hardware
- Insider & External Attacks
- Software
- Media
- Organizational Failure
- Curatorial Error
Do you know where your data are?

- How is content stored?
- How is content Replicated?
- How is content audited?
Encoding

Priscilla Caplan
<pcaplan@ufl.edu>
Assistant Director for Digital Library Services,
Florida Virtual Campus
Compression

• Many types of compression:
  – Format based file compression, e.g. JPEG2000
  – Tape hardware compression at the drive
  – NAS compression via appliance or storage device
  – Data deduplication

• Is it lossless?
• Is it transparent?
• Is it proprietary?
• What is effect on error recovery?
Compression Tradeoffs

• Tradeoffs
  – Space savings allows more copies at same cost
  – But makes files more sensitive to data corruption

• Erasure coding in cloud storage
  – Massively more reliable
  – But dependent on proprietary index
Encryption

• Two contexts:
  – Archiving encrypted content
  – Archive encrypting content

• Reasons to encrypt:
  – Prevent unauthorized access
    • Especially in Cloud and on tape
  – To enforce DRM
  – Legal requirements (HIPAA, state law)
    • Though only required for transmission, not “at rest”
Encryption Concerns

- Increased file size
- Performance penalty
- Additional expense
- But makes files more sensitive to data corruption
- May complicate format migration
- May complicate legitimate access
- Risk of loss of encryption keys
- Difficulty of enterprise level key management
- Obsolescence of encryption formats
- Obsolescence of PKI infrastructure
Redundancy & Diversity

Andrea Goethals
<andrea_goethals@harvard.edu>
Manager of Digital Preservation and Repository Services,
Harvard University
Failures WILL happen

- Real problem: failures you can’t recover from!
- A few mitigating concepts: redundancy & diversity
Redundancy (multiple duplicates)

• Ecology
  – Redundancy hypothesis = species redundancy enhances ecosystem resiliency

• Digital preservation
  – Example: Multiple copies of content
Diversity (variations)

• Finance
  – Portfolio effect = diversification of assets stabilizes financial portfolios

• Ecology
  – Response diversity = diversification stabilizes ecosystem processes

• Digital preservation
  – Examples: different storage media, storage locations with different geographic threats
What can “fail”? What can’t?

• Likely candidates
  – Storage component faults
    • Latent sector errors (physical problems)
    • Silent data corruption (higher-level, usually SW problems)
    • Whole disks
  – Organizational disruptions (changes in finances, priorities, staffing)
<table>
<thead>
<tr>
<th>Data loss risks (impact &amp; likelihood?)</th>
<th>Redundancy &amp; diversity controls (costs?)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Environmental factors</td>
<td>Replication to different data centers</td>
</tr>
<tr>
<td>e.g. temperature, vibrations affecting multiple devices in same data center</td>
<td></td>
</tr>
<tr>
<td>Shared component faults</td>
<td>Replication to different data centers or redundant components, replication software systems</td>
</tr>
<tr>
<td>e.g. power connections, cooling, SCSI controllers, software bugs</td>
<td></td>
</tr>
<tr>
<td>Large-scale disasters</td>
<td>Replication to different geographic areas</td>
</tr>
<tr>
<td>e.g. earthquakes</td>
<td></td>
</tr>
<tr>
<td>Malicious attacks</td>
<td>Distinct security zones</td>
</tr>
<tr>
<td>e.g. worms</td>
<td></td>
</tr>
<tr>
<td>Human error</td>
<td>Different administrative control</td>
</tr>
<tr>
<td>e.g. accidental deletions</td>
<td></td>
</tr>
<tr>
<td>Organizational faults</td>
<td>Different organizational control</td>
</tr>
<tr>
<td>e.g. budget cuts</td>
<td></td>
</tr>
</tbody>
</table>
Added software to the list of components... bugs in the software can also cause correlated failure.

- Micah
  micah, 7/16/2012
BIT-LEVEL FIXITY

Karen Cariani
<karen_cariani@wgbh.org>
Director WGBH Media Library and Archives,
WGBH Educational Foundation

John Spencer
<jspencer@bmschase.com>
President, BMS/Chase LLC
Bit-Level Fixity

• Fixity is a “property” and a “process” (as defined from the 2008 PREMIS data dictionary)

• It is a “property”, where a message digest (usually referred to as a checksum) is created as a validation tool to ensure bit-level accuracy when migrating a digital file from one carrier to another
Bit-Level Fixity

• It is also a “process”, in that fixity must be integrated into every digital preservation workflow
• Fixity is common in digital repositories, as it is easily put in the ingest and refresh migration cycles
• Fixity of digital files is a cornerstone of archival best practices
So what’s the problem?

- While bit-level fixity solutions are readily available, there remains a large constituency of content creators that place minimal (or zero) value on this procedure.
- Legacy IT environments, focused on business processes, are not “standards-driven”, more so by vendors, budgets, and poorly defined archival workflow strategies.
So what’s the problem?

• A vast majority of commercial digital assets are stored “dark” (i.e. data tape or even worse, random HDDs), with no fixity strategy in place

• For private companies, individuals, and content creators with digital assets, bit-level fixity remains a mystery – a necessary outreach effort remains
So what’s the problem?

• Major labels, DIY artists, indie labels, amateur and semi-professional archivists, photographers, oral histories, and born-digital films usually ignore the concept of fixity

• All of the these constituencies need guidance to engage fixity into their daily workflow or suffer the consequences when the asset is needed NOW to monetize…
Overview:

Auditing & Repair

Micah Altman
<Micah_Altman@alumni.brown.edu>
Director of Research, MIT Libraries
Audit [aw-dit]:

An independent evaluation of records and activities to assess a system of controls

Fixity mitigates risk only if used for auditing.
Functions of Storage Auditing

- Detect corruption/deletion of content
- Verify compliance with storage/replication policies
- Prompt repair actions
Bit-Level Audit Design Choices

• Audit regularity and coverage: on-demand (manually); on object access; on event; randomized sample; scheduled/comprehensive

• Fixity check & comparison algorithms

• Auditing scope: integrity of object; integrity of collection; integrity of network; policy compliance; public/transparency auditing

• Trust model

• Threat model
Repair

Auditing mitigates risk only if used for repair.

Design Elements
• Repair frequency
• Repair algorithm
• Repair duration
# LOCKSS Auditing & Repair

*Decentralized, peer-2-peer, tamper-resistant replication & repair*

<table>
<thead>
<tr>
<th>Regularity</th>
<th>Scheduled</th>
</tr>
</thead>
<tbody>
<tr>
<td>Algorithms</td>
<td>Bespoke, peer-reviewed, tamper resistant</td>
</tr>
<tr>
<td>Scope</td>
<td>- Collection integrity</td>
</tr>
<tr>
<td></td>
<td>- Collection repair</td>
</tr>
<tr>
<td>Trust model</td>
<td>- Publisher is canonical source of content</td>
</tr>
<tr>
<td></td>
<td>- Changed contented treated as new</td>
</tr>
<tr>
<td></td>
<td>- Replication peers are untrusted</td>
</tr>
<tr>
<td>Main threat models</td>
<td>- Media failure</td>
</tr>
<tr>
<td></td>
<td>- Physical Failure</td>
</tr>
<tr>
<td></td>
<td>- Curatorial Error</td>
</tr>
<tr>
<td></td>
<td>- External Attack</td>
</tr>
<tr>
<td></td>
<td>- Insider threats</td>
</tr>
<tr>
<td></td>
<td>- Organizational failure</td>
</tr>
<tr>
<td><strong>Key auditing limitations</strong></td>
<td>- Correlated Software Failure</td>
</tr>
<tr>
<td></td>
<td>- Lack of Policy Auditing, public/transparent auditing</td>
</tr>
</tbody>
</table>
# DuraCloud Auditing & Repair

*Storage replicated across cloud providers*

<table>
<thead>
<tr>
<th>Regularity</th>
<th>On-demand</th>
</tr>
</thead>
<tbody>
<tr>
<td>Algorithms</td>
<td>Combination of bespoke algorithms and cloud provider</td>
</tr>
<tr>
<td>Scope</td>
<td>Object integrity only (no repair)</td>
</tr>
<tr>
<td>Trust model</td>
<td>- Content distributor (DuraCloud client) is completely trusted</td>
</tr>
<tr>
<td>Main threat models</td>
<td>- Media failure</td>
</tr>
<tr>
<td></td>
<td>- Physical Failure</td>
</tr>
<tr>
<td><strong>Key auditing limitations</strong></td>
<td>- Limited range of threat models (e.g. software, curatorial failure).</td>
</tr>
<tr>
<td></td>
<td>- Lack of scheduled auditing; collection integrity checks; policy auditing; repair.</td>
</tr>
</tbody>
</table>
# iRODS Auditing & Repair

*Rules-based federated storage grid*

<table>
<thead>
<tr>
<th>Regularity</th>
<th>Scheduled, On-event</th>
</tr>
</thead>
<tbody>
<tr>
<td>Algorithms</td>
<td>Bespoke, peer-reviewed</td>
</tr>
</tbody>
</table>
| Scope            | - Collection integrity  
                  | - Collection repair  
                  | - Micro-service policy auditing |
| Trust model      | - Operator is implicitly trusted for content (by default)  
                  | - More complex relationships possible through federation, microservices |
| Main threat models | - Media failure  
                    | - Physical Failure  
                    | - Policy implementation failure (auditing) |
| **Key auditing limitations** | - Limited range of threat models (e.g. software, curatorial failure) – some addressable through federation and microservices.  
                                | = Lack of policy auditing, transparent/public auditing (by default) |
### TRAC-Aligned policy auditing as a overlay network

<table>
<thead>
<tr>
<th>Regularity</th>
<th>Scheduled; Manual</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fixity algorithms</td>
<td><em>Relies on underlying replication system</em></td>
</tr>
</tbody>
</table>

**Scope**
- Collection integrity
- Network integrity
- Network repair
- High-level (e.g. trac) policy auditing

**Trust model**
- External auditor, with permissions to collect meta-data/log information from replication network
  - Replication network is untrusted

**Main threat models**
- Software failure
- Policy implementation failure
  (curatorial error; insider threat)
- Organizational failure
- *Media/physical failure through underlying replication system*

**Key auditing limitations**
- Relies on underlying replication system, (now) LOCKSS, for fixity check and repair
Summary:

Micah Altman  
<Micah_Altman@alumni.brown.edu>  
Director of Research, MIT Libraries
# Methods for Mitigating Risk

## Local Storage
- **Physical:** Media, Hardware, Environment
- **Formats**
- **File Transforms:** compression, encoding, encryption
- **File Systems:** transforms, deduplication, redundancy

## Replication
- Diversification of copies
- Number of copies

## Verification
- Fixity
- Audit
- Repair

---

**Image:**
- A diagram showing methods for mitigating risk in local storage, replication, and verification.
How can we choose?

• Clearlv state decision problem

• Model connections between choices & outcomes

• Empirically calibrate and validate
The Problem

Keeping risk of object loss fixed -- what choices minimize $? 

“Dual problem”

Keeping $ fixed, what choices minimize risk?

Extension

For specific cost functions for loss of object:

Loss(object_i), of all lost objects

What choices minimize:

Total cost = preservation cost + \( \text{sum}(E(\text{Loss})) \)

Are we there yet?
Measurements

• Media – MBTF theoretical and actual
• File transformations:
  – compression ratio
  – partial recoverability
• Filesystem transformations:
  – Deduplication
  – Compression ratio
• Diversification
  – Single points of failure
  – Correlated failures
• Copies, Audit, Repair
  – Simulation models
  – Audit studies
Questions*

What techniques are you using?

What models guide the “knobs”?

Contact the NDSA Infrastructure Working Group:

www.digitalpreservation.gov/ndsa/working_groups/

* Thanks to our moderator:

  Trevor Owens <trow@loc.gov>, Digital Archivist, Library of Congress