Using the SafeArchive System: TRAC-Based Auditing of LOCKSS

Micah Altman; Institute for Quantitative Social Science, Harvard University; Cambridge, M.A., U.S.A.
Jonathan Crabtree; Odum Institute, University of North Carolina, Chapel Hill; Chapel Hill, N.C., U.S.A.

Abstract

The goals of SafeArchive are to make distributed replication easier, and to automate compliance with formal replication and storage policies. In this article, we describe the process of automated archival policy auditing in detail. First, we provide an overview of the SafeArchive system and we describe how a curator can use the tools to generate an archival policy schema and monitor it, simply. Second we identify specific TRAC criteria that can be verified automatically, and additional criteria that can be supported through integrated documentation. Third, we discuss the technical implementation of the system including the policy schema; how information used in the auditing process is obtained from a set of LOCKSS peers without modifying the LOCKSS trust model or configuration; and how the software is organized into components.

The Need for Policy-Based Replication

Verified geographically-distributed replication of content is an essential component of any comprehensive digital preservation plan. This requirement has emerged as a necessity for recognition and certification as a trusted repository – in order to be fully trusted, an organization must have a managed process for creating, maintaining, and verifying multiple geographically distributed copies of its collections. This requirement has been embodied in Trustworthy Repositories Audit & Certification (TRAC), an emerging ISO standard, and in other best practices [1]. Furthermore, an organizationally distributed, collaborative approach is required to minimize threats from internal attack, economic failure, and organizational failure [2].

The LOCKSS (Lots of Copies Keeps Stuff Safe) system [3] has been widely adopted by libraries and archives for replication and preservation. LOCKSS is simple to install and administer (in part because it is a self-contained system), has minimal hardware requirements, and can readily replicate content on existingweb-sites. Each LOCKSS peer (a.k.a. "cache") independently harvests content from content owners – and the peers collaborate to check the integrity of the content, and to repair caches that lose or corrupt previously harvested content.

The peer-to-peer (P2P) model that LOCKSS employs, along with careful attention to security in its design, make it resistant to operator error, outside attacks,. insider attacks, and the failure of a single institution. Like many P2P designs, however, it provides no way for peers to make credible resource commitments to each other; to ensure that peers always preserve new content from a particular source; or to ensure that a minimum number of copies are made of a particular collection.

Furthermore, although LOCKSS provides low-level verification of the integrity of content across caches, it does not support the auditing required by archival standards such as TRAC. For example, there is no supported mechanism for a content owner or the owner of a participating LOCKSS peer to determine how many copies of an item are replicated in the network, or how frequently these are verified. Nor is it possible for a content owner to easily determine what caches harvest their content, or the completeness and freshness of such replication.

Overview of the SafeArchive System

The SafeArchive system fills this gap by coordinating and auditing existing groups of locks peers. Without requiring a single authority, this allows a group of institutions to establish actionable and mutually verifiable policies governing the replication of content of interest to those institutions. This solution provides the reliability of a top-down replication system with the resilience of a peer-to-peer model.

The SafeArchive system is an open source and available at:

www.safearchive.org

SafeArchive is based on a prototype [2] developed by the Data-PASS partners [4,5], and funded by the Library of Congress This prototype established feasibility and the core operational use cases for the system. The SafeArchive has been completely rewritten and redesigned for production use.

Abstractly, the system is designed to create a virtual overlay network on top of a peer-to-peer replication network, to support provisioning, monitoring, and TRAC-based auditing (Fig 1.):

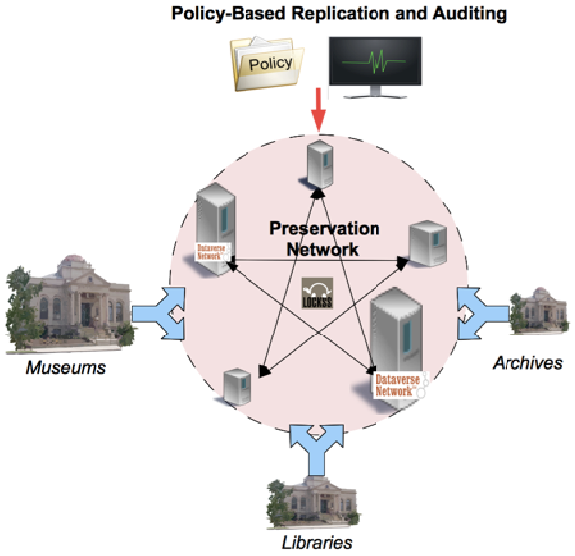

***Figure 1**. Conceptual diagram of Institutional Use of SafeArchive*

Operationally, users of the system can: (1) Analyze any LOCKSS network; (2) check that collections are replicated, valid, and up-to-date; (3) create formal replication policies; (4) and audit the network for current and historical TRAC compliance. With the next release, scheduled for this summer, the SafeArchive system will also be able to automatically manage and repair a LOCKSS network based on a specified replication policy.

Furthemore,the SafeArchive system is designed to collaborate with the Dataverse Network [6] system. Curators who store content in Dataverse can easily expose content for replication by LOCKSS and SafeArchive through a simple graphical interface.

Institutionally, the SafeArchive enables memory institutions and preservation collaborations to formalize their replication policies and inter-archival replication commitments; represent these in machine-readable form; and to continuously audit any set of public or private LOCKSS hosts for compliance.

How the SafeArchive System Works

Generally speaking, the system coordinates six activities:

(1) Collaborating institutions agree on a replication policy. This records the resource commitments, descriptions of the collections to be preserved, and desired replication guarantees (such as number of copies, frequency of verification, and freshness of content).
(2) Institutions make collections of content ("archival units") available through the web, e.g. as web pages or through the Dataverse Network.
(3) LOCKSS caches harvest the collections from their original source repositories, using standard protocols such as HTTP or OAI-PMH.
(4) LOCKSS caches coordinate peer-to-peer to monitor and maintain the integrity of the network. Caches repair content that is corrupted and restore content when hosts are replaced.
(5) SafeArchive monitors network, assesses it against the stated replication policy, and produces an audit trail. The system also allerts collaborating when formal policies are not met.
(6) In future, SafeArchive will also coordinate harvesting of the locks caches by "inviting" members of the network to harvest content that is under-replicated. This will be used to automatically configure a network based on a policy schema, to reconfigure and repair the network as the number of participating caches, collections and institutions changes intentionally or unintentionally.

The SafeArchive system is designed to give curators the ability to easily define preservation policy, examine the content of the preservation network, and generate regular audit reports that support TRAC compliance. The tools are designed to be easy to use and once initial configurations are complete the system provides automated reports of policy compliance. And all changes to the policy schema instance and the machine-readable audit reports are versioned and stored permanently – so that there is complete history of compliance.

Using the System

The The SafeArchive system can be configured to audit existing LOCKSS caches that are either standalone (public) or configured in a Private LOCKSS Network (PLN). In compliance with the LOCKSS trust model the system requires local LOCKSS owners to allow access, and limits access to the minimum required. To allow the SafeArchive system the ability to audit the local caches owners must provide authentication information (and allow network access, if blocked by a firewall). The LOCKSS system supports creation of limited-privilege non-accounts for this auditing purpose. In no case, can SafeArchive cause content to be deleted from a cache or perform "super-user" operations.

Once the SafeArchive system has been provided the location and authentication information of LOCKSS caches to be audited, curators can specify the policies their organizations would like to monitor for compliance. The curator begins by answering a web-based questionnaire that is designed to automatically populate the machine-readable policy schema. (Alternatively, the curator can use a standard XML authoring tool to create a policy schema instance.)

As a part of this survey the curator defines the frequency of automated reporting. The reports are delivered using a user-friendly template and provide a simple view of policy compliance. A detailed machine-readable audit report is also available for automated processing.

The reports contains both "audit" summaries that reflect compliance with archival policy, and "operational" information that can be used for diagnostics and performance analysis.

Table 1: Example Report Fragments

Preservation Network Summary	
Mean Up Time for Hosts	37d:7h:18m:44s
Number of Hosts Reachable	7
Number of Hosts NOT Reporting	0
Number of Unique AUs	6
Total Number of Replicates	34
Total Disk Space in Use	2.28 TB
Total Disk Space Free	19.7 TB

Verified Replicates			
ID	**AU Name**	**Verified Replicates**	**Conforms to Policy**
1	ICPSR	0	*FALSE*
2	IQSS Dataverse	5	TRUE
3	Odum Dataverse	4	TRUE
4	Roper Collection	5	TRUE

For example, the sample audit report fragments in Table 1 shows segments of the report sent to administrators and archivists to monitor their preservation network. The audit report fragment shows that all archival units are within the parameters expected by the policy, except one. This archival unit, which is the process of verification, does not yet have any verified copies and fails the minimum replication requirements. The operational report fragment shows (among other things) the amount of content held by the network and the amount of storage space available.

Table 2 provides more detailed descriptions of the categories of information available in the reports:

Table 2: SafeArchive Report Information

Category	Explanation
Versioning	Provides date and version information for reports and related policy schema.
Collection Replication Policies	Reports conformance with replication policies (number of replicas, verification, freshness, and distribution requirements)
Host Storage Policies	Reports whether storage provided by caches meets institutional commitments to the network
Network Operations Summary	Summarizes size of content held in network, available storage, andoverall availability of caches.
LOCKSS Diagnostics	Shows behavior of each participating cache, including crawl and poll behavior

Aligning Replication Policies and TRAC

There are two different ways in which the system aligns with and supports TRAC: In this section we identify TRAC criteria that can be verified automatically, and additional criteria that can be supported through integrated documentation.

First, proper use of SafeArchive can provide evidence of TRAC compliance. Because the system automatically audits (and in the future, will reconfigure the network) for compliance with collection integrity, replication and freshness guarantees it provides supporting evidence for compliance with the general TRAC areas of archival storage & preservation (B4); independent audit mechanisms (B2); appropriate system infrastructure (C1); and disaster planning and recover (C3).

Specifically, Table 3 shows the direct evidence of compliance produced in the audit trail:

Table 3: Trac Criteria Directly Supported by SafeArchive

Trac Criterion	SafeArchive Support
B.4.4	SafeArchive uses LOCKSS mechanisms to continually monitor integrity. SafeArchive audit trails document integrity failures.
B.2.12, B4.5	SafeArchive audit trails document replication actions taken to decrease risk and repair actions made to restore collection integrity.
B.6.4	SafeArchive design ensures that access to replicated collections is restricted to hosts that have demonstrated the ability to access the original content.
C1.1	SafeArchive is built on Linux, and on a well-supported set of open-source components. It functions on well-supported operating systems and core infrastructural software.
C1.2	SafeArchive audit trails demonstrate that sufficient backup storage is being provided by multiple institutions.
C1.3	The audit trail demonstrates that the number, distribution, and freshness of copies meets policy.
C1.4	LOCKSS mechanisms automatically synchronize copies of digital objects.
C1.5	LOCKSS mechanisms detect bit corruption and loss. SafeArchive audit trail demonstrates the frequency of verification of content.
C1.6	SafeArchive provides tools for generating automatic reports and alerts if data is corrupted or lost.
C3.1-4	SafeArchive replication contributes to disaster planning and recovery.

Second, TRAC-relevant documentation describing the participants in a SafeArchive network can be included in the SafeArchive policy, as documentation. The policy schema includes "hooks" which can be used to document properties (either directly or by reference to external documentation) that are relevant to a TRAC assessment, and to link this documentation to specific TRAC elements.

This second type of documentation is *not auditable by the SafeArchive system. Instead it is included so that the policy instance is a complete self-contained document* about the replication network. This aids participants in the replication network in fully understanding the trustworthiness of the system and the institutions hosting it.

For example, the host portion of the policy schema can be used to document the system security of hosts in the network; and the Network portion may be used to document the

preservation mission of the organization running SafeArchive. Table 4 provides more specific recommendations for documentation that can be included in a SafeArchive policy:

Table 4: Recommended Documentation Elements

Trac Criterion	Where to Document in SafeArchive Policy
A1-A4	Institutions should document the organizational infrastructure of the *Virtual Organization* running SafeArchive in the **Network** sections of the policy document.
A5	Institutions may document terms specific to a particular collection in the **Archival Unity** section for that collection
B2.5, B2.7, B5.2	Institutions may provide documentation on naming, identifiers, and other context for a particular collection in the **Archival Unit** section for that collection
C1.7-C1.10	Institutions may provide documentation on system setup, maintenance and other system information for each LOCKSS cache in the **Host** section of the policy, for each host in the network.

Technical Details

In this section we describe technical details of the system, including the policy schema, mechanisms for collecting information about the LOCKSS network, and software components that make up the system.

Policy Schema Examples

The policy schema is comprised of three sections. The first section of the schema defines the **Network**. It groups information that can be used to identify and describe network components. The required fields in this group are network name, description and contact. Additional information used to document network level TRAC compliance is recommended. This can also specify minimum requirements for archival replication across the network, which supplements the replication requirements in the **Archival Unit** section.

The second section of schema defines the participating **Hosts**. This includes identification and contact information for the participating LOCKSS caches; and the storage commitment that the owner of the host has made to the network. In addition these fields can record operating parameters; and can contain TRAC documentation related to that host.

The final section of the schema defines the **Archival Units** (collections of content) that are replicated and monitored by the system. An example of this section is shown in Figure 2, below. This section records identification information and references to the LOCKSS plugins that are to harvest that collection. This section also contains the replication policies for that collection, including: update frequency, storage commitment by the network to the collection, and the number of desired replicas.

Optionally, this section can contain documentation about terms of use of the collection, and other TRAC documentation relevant to the content.

```
<archivalUnits>
    <au au_id="edu|harvard|iq|dvn|lockss|plugin|DVNOAIPlugin>
        <auIdentity>
            <name>ICPSR</name>
        </auIdentity>
        <auCapabilities>
            <numberReplicates min="3"/>
            <verificationFrequency maxDays="21"/>
            <replicationDuration maxDays="21"/>
            <updateFrequency minDays="7"/>
            <storageRequired max_size="2000"/>
        </auCapabilities>
        <auTerms/>
    </au>
```

Figure 2. Sample from a SafeArchive policy schema

For ease of creation, SafeArchive includes a tool that creates the schema through an online questionnaire and probes of participating caches.

Auditing LOCKSS Caches

Information used in the auditing process is obtained directly from a set of LOCKSS peers, without violating the LOCKSS trust model or requiring additional patches to the LOCKSS installation. We summarize the details in this section.

The SafeArchive system includes a network monitoring component responsible for monitoring each cache participating in the network. The workhorse of the monitoring component is the Cache Status Extractor, which interrogates each LOCKSS cache to collect all of the information necessary to support policy reporting and auditing. Data collected by the Extractor is stored in the Network Monitor database tables. And the frequency of extractor activity is controlled by cron (automatic) and console (manual) services provided by the Network Monitor. The Extractor gather information on caches. and the archival units they contain. The extractor also collects information on harvests and polls performed by the LOCKSS caches as part of their normal operation. These data will be used later to verify that preservation policies have been met.

The Cache Status Extractor retrieves information through a standard HTTP servlet request provided by LOCKSS. In order to access the servlet, the system requires an account with limited privileges on each cache. This account and password is created using supported LOCKSS mechanisms, and is used only to gather information. This information is returned in XML format, or in plain text.

The Extractor uses LOCKSS-specific business logic to summarize and translate the raw results so that it can be more easily used in locks results. For example, since the LOCKSS system uses dynamic polling, the process of determining the number of verified collections is somewhat complex. In some case many replicas of a collection may exist but may not

actually have been verified at the time of an audit. To determine which archival units have actually been verified, the Extractor collects information about all of the polls in which a LOCKSS cache has participated, and then analyzes this polling data to determine when a particularly Archival Unit has been agreed to by a sufficient number of caches.

Finally the Cache Status Extractor places both raw and processed information (via EJB) into MYSQL tables for use by other system components. The action of these components is described in the next section.

System Components

The SafeArchive software is written primarily in Java using EJB, Glassfish, and JSF; is backed by a MySQL database; and runs under Linux. The Eclipse BIRT system is used as a framework for report generation. The entire system is organized into a set of semi-independent components, shown in Figure 3:

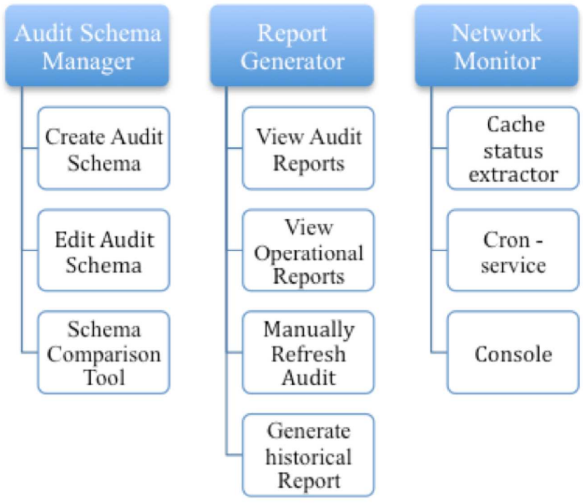

***Figure 3**. Components of the current system*

Gathering information on the network is accomplished by the Network Monitor component. In this section we describe how other components of the system act on this information to provide auditing, reporting, and reconfiguration.

The Audit Schema manager is the heart of the auditing functionality. The Audit Schema manager facilitates creation and editing of formal policies posing a series of preservation requirements questions to the archive manager through a web interface. Information gathered through the Network Monitor are used to provide reasonable default values and to pre-populate selections. The output of this interview is a well-defined preservation policy rule set, expressed in XML that specifies formally the policies to be audited. This output is versioned and placed into a secure UNIX file system for use by other tools and inspection at a future occasion.

The Audit Schema component also provides a schema comparison tool. This tool compares the schematize policy with the actual state of the preservation network (as determined by the Network Monitor), and produces a set of machine-readable "diffs" that enumerate all differences between the actual and desired states.

The Report Generator component creates formatted audit reports and operational reports. High-level policy reports can be generated either directly from the Network Monitor data or from the "diffs" produced. This template is stored in an XML design format. More detailed operational reports are generated directly from the Network Monitor data, using a read-only EJB connection, and the BIRT report generator framework. Reports can be generated on-demand through a "console" service, and generated automatically through a "cron" service.

Two additional components are in development, as illustrated in figure 4:

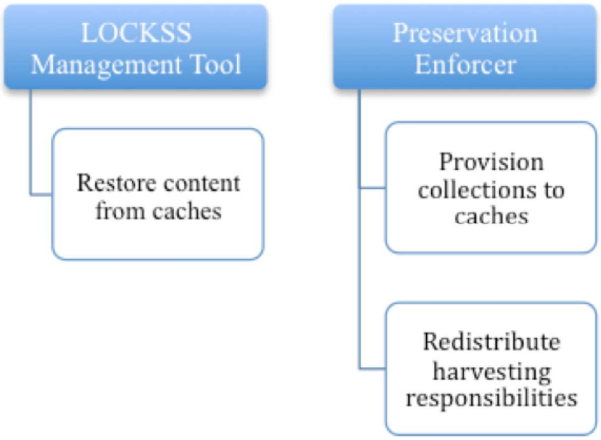

***Figure 4**. Components in Planning*

At present, if content needs to be retrieved from the system, it must be retrieved directly from one of the participating caches holding that content using LOCKSS-native mechanisms. The LOCKSS management tool will coordinate the location of an appropriate cache and the restoration of content from it.

The preservation enforcer component will allow the system to make "adjustment" requests to individual LOCKSS caches. These will invite caches to start or discontinue harvesting particular content using the standard LOCKSS administrative interface. While this requires a higher level of privileges that simple monitoring of the network, the privileges are still limited. All invitations to stop and start harvesting are auditable by the cache owners, and the system can never be used to delete existing content on the caches. Thus the LOCKSS trust model is honored. We are working with the LOCKSS team to fine-tune an appropriate set of permissions and interface mechanisms.

Summary

The SafeArchive system provides a way to ensure that replicated collections are both institutionally and geographically distributed and to allow for the development of increasingly measurable and auditable trusted repository requirements. Designed as a virtual overlay network on LOCKSS, the system provides the auditability and reliability of a top-down replication

system with the resilience of a peer-to-peer model. This enables any library, museum, or archive to audit that its content is being replicated across an existing LOCKSS network in conformance with documented archival policies; and to allow groups of collaborating institutions to automatically and verifiably replicate each others' content consistent with a set of expressed commitments. The result is that archives can more easily collaborate to preserve content through geographically and institutionally replication; which mitigates against technical and organizational threats to preservation.

Acknowledgements

The project is a collaborative effort of the Data-PASS Partners: The International Consortium for Political and Social Research, U. Michigan; The Roper Center for Public Opinion Research, Research at the University of Connecticut, the Howard W. Odum Institute at the University of North Carolina-Chapel Hill, NARA, and the Institute of Quantitative Social Science, Harvard University. It is managed through the Institute of Quantitative Social Science, and works in collaboration with the LOCKSS project at Stanford University.

The project is sponsored by the Institute of Museum and Library Services (IMLS), under award #LG-05-09-0041-09.

References

[1] RLG-National Archives and Records Administration Task Force, Trustworthy Repositories Audit and Certification: Criteria and Checklist (TRAC) Ver 1.0 (Chicago, IL: Center for Research Libraries http://www.crl.edu/PDF/trac.pdf). (2007)

[2] Micah Altman ; Bryan Beecher,; Jonathan Crabtree; Leonid Andreev, Ed Bachman, Adam Buchbinder , Steve Burling, Patrick King, Marc Maynard, "A Prototype Platform for Policy-Based Archival Replication", *Against The Grain* v. 21(2). Pgs. 44-47. (2009)

[3] Vicky Reich & David S. H. Rosenthal, 2001. "LOCKSS: A Permanent Web Publishing and Access System", *D-Lib Magazine* 7(6).

[4] Altman, M., Adams, M., Crabtree, J., Donakowski, D., Maynard, M., Pienta, A., & Young, C. (2009). "Digital preservation through archival collaboration: The Data Preservation Alliance for the Social Sciences." *The American Archivist*. 72(1). Pgs. 169-182. (2009).

[5] Myron Gutmann, Abrahamson, M, Adams, M.O., Altman, M, Arms, C., Bollen, K., Carlson, M., Crabtree, J., Donakowski, D., King, G., Lyle, J., Maynard, M., Pienta, A., Rockwell, R, Timms-Ferrara L., Young, C.. "From Preserving the Past to Preserving the Future: The Data-PASS Project and the challenges of preserving digital social science data", *Library Trends* 57(3). Pgs. 315-337. (2009)

[6] Merce Crosas., "The Dataverse Network: An Open-Source Application for Sharing, Discovering and Preserving Data", *D-Lib Magazine* 17(1/2). (2011)

Author Biography

Micah Altman (Ph.D. California Institute of Technology) is Senior Research Scientist in the Institute for Quantitative Social Science in the Faculty of Arts and Sciences at Harvard University and Archival Director of the Henry A. Murray Research Archive. Dr. Altman conducts research in social science informatics, social science research methodology, and American politics, focusing on the intersection of information, technology, and politics; and on the dissemination, preservation, and reliability of scientific knowledge.

JONATHAN CRABTREE is Assistant Director for Archives and Information Technology at the Odum Institute for Research in Social Science at University of North Carolina, Chapel Hill. Crabtree joined the institute 15 years ago and is responsible for designing and maintaining the technology infrastructure that supports the institute's wide array of services. He is also completing an advanced degree in the School of Information & Library Science at UNC

