LOTS OF COPIES KEEP STUFF SAFE

16 March 2016

How to set up a Private LOCKSS Network (PLN)

Table of contents

Overview

The LOCKSS software was originally built to provide reliable, peer-based preservation and persistent access for electronic scholarly publications. PLNs leverage the same framework for the preservation of designated digital content of all types, within a community of interested and invested organizations. This documentation addresses the technical steps needed to configure and manage a PLN.

PLN components

A PLN consists of (a minimum of six) **nodes**. PLN nodes run the LOCKSS software and collectively preserve the content designated by the PLN member organizations.

PLN nodes inherit PLN-specific configuration through a **configuration server**. The configuration server may be managed either locally by the PLN or remotely by the LOCKSS team and hosts configuration for three aspects of the PLN: properties, the plug-in registry, and the title database. Nodes access this information as versioned Web content that they then preserve; their operation does not depend on the configuration server being up.

A large number of **properties** can be modified to customize PLN operation and behavior. The **plug-in registry** stores the plug-ins necessary to transform the arrangement of content on a content source into discrete information packages called **Archival Units** (AUs). The **title database** is the inventory list of AUs available to be preserved.

PLN nodes retrieve content over the Web, from an incidental interface to an existing system and/or from a staging server configured specifically for this purpose.

The specific component that harvests content is called the LOCKSS **daemon**. The daemon is also responsible for storing the harvested content in AUs, computing checksums, and regularly monitoring integrity by comparing the preserved content with that stored in other nodes through a polling mechanism.

Set up configuration server

The configuration server itself does not run the LOCKSS software; it merely serves as a web host for a set of files that collectively define three key aspects of the PLN: behaviors of the LOCKSS software on all of the PLN nodes (i.e., properties), how to retrieve and package the content made available to the PLN for preservation (i.e., plug-ins), and the location and description of the content made available to the PLN for preservation (i.e., titles). The initial implementation of the configuration server may feature default or empty values for all three aspects, to be refined later once the source content structure(s) and location(s) are more firmly defined. Each node loads the configuration information upon initialization and periodically checks for changes.

The configuration server is managed centrally by the LOCKSS team on behalf of many PLNs. To more fully realize the benefits of truly distributed preservation, the LOCKSS team encourages PLN management of their own configuration server but recognizes a shortfall of documentation on how to do this. The LOCKSS team is therefore interested in partner feedback on how to make this easier.

Properties

Many aspects of the PLN configuration, including both plug-ins and titles, are controlled through parameters in the lockss.xml file. Properties are key-value pairs that affect the behavior of the LOCKSS daemon. For reference, the LOCKSS team has created a sample properties file[1] and partially annotated a complete list of properties.[2]

Plug-in registry

The plug-in registry hosts as many plug-ins as are necessary to parse the content made available on source platforms, typically one plug-in per platform. A plug-in instructs the LOCKSS software how to navigate source platforms and organize the remote structure into local AUs. In addition to general documentation on plug-ins,[3] the LOCKSS team has created a tool to facilitate plug-in design and testing[4] and a sample plug-in registry.[5]

Though a full list and description of plug-ins that have been created by either the LOCKSS team or the PLN community has not yet been assembled, the LOCKSS team is knowledgeable about existing plug-ins and will help PLNs to re-use them whenever possible. Widely-used academic publishing platforms, content management systems, and repository platforms are well covered by existing plug-ins.

Title database

The title database enumerates the content to be preserved in the PLN. The title database and plug-in registry closely complement one another; each AU entry in the title database specifies an associated plug-in along with all of the parameter values required by that plug-in. The most important parameter value, which is ubiquitously required regardless of plug-in, is the base URL of the candidate content.

The title database itself is an XML file conventionally named titlesdb.xml. For ease of editing and to reduce the chance that updates may accidentally produce malformed XML, constituent AU entries are now modified in individual TDB text files then processed into the titlesdb.xml file.

[1] http://props.lockss.org:8001/samplepln/lockss.xml
[2] http://www.lockss.org/lockssdoc/gamma/daemon/paramdoc.html
[3] http://www.lockss.org/locksswiki/index.php/Plugins
[4] http://www.lockss.org/locksswiki/index.php/Plugin_Tool
[5] http://props.lockss.org:8001/samplepln/plugins/

The LOCKSS team provides an example of the syntax of these lightweight files[6] but has not yet produced documentation on how PLNs may maintain them for themselves. For the moment, PLNs will need to work closely with the LOCKSS team to populate the title database.

Set up nodes

Configuration of an individual PLN node involves installation and configuration of the LOCKSS software on a physical or virtual machine. The LOCKSS team has created documentation for this process,[7] noting at the appropriate points the two PLN-specific configuration details: PLN name and the URL of the lockss.xml file.

Prepare content for ingest

The LOCKSS software can only ingest content into the PLN by web harvest. The source content must therefore not only be web-accessible to the LOCKSS nodes, it should also be optimized for access by a crawler. The web access provided by many widely-used academic publishing platforms, content management systems, and repository platforms satisfies this requirement; consult with the LOCKSS team to determine whether the PLN content source system(s) are already interoperable with the LOCKSS software.

If no existing web interface is adequate and one or more dedicated staging locations are instead determined necessary, PLN operators may follow the general guidelines[8] prepared by the LOCKSS team for staging content. The simplest way to stage content for ingest is to use the Apache hierarchical directory listing,[9] with web access limited to the IP addresses of the PLN nodes. Each bundle of content to be packaged into an AU must also be accompanied by either a LOCKSS permission statement or a Creative Commons license,[10] authorized by the publisher(s). This establishes that each PLN Node has permission to preserve the designated content, through an attestation that accompanies each AU.

[6] http://documents.clockss.org/index.php/LOCKSS:_Basic_Concepts#Title_Database
[7] http://www.lockss.org/docs/LOCKSS-Linux6-Install.pdf
[8] http://www.lockss.org/support/prepare-your-content/
[9] http://httpd.apache.org/docs/trunk/mod/mod_autoindex.html
[10] http://creativecommons.org/licenses/

Implement ingest plug-in

If the precise arrangement of the source content was unknown prior to configuration of the plug-in registry and/or if new content is staged for access by the PLN, plug-ins will need to be added or updated after the content is prepared for ingest. As previously indicated, consult with the LOCKSS team to determine whether a suitable plug-in may already exist for your source content platform and structure. If you need to modify or write a new plug-in, check out the documentation on the plug-in tool.[11]

View content

Once PLN configuration is completed, plug-in(s) have been implemented, and harvesting initiated, the preservation status of the designated content can be monitored through the administrative user interface (UI). The administrative UI for any given node is accessible by default via port 8081. This can be set to an alternate port through the properties.[12]

Once authenticated, the relevant section of the administrative UI to consult is *Daemon Status*. The MetaArchive Cooperative has produced some documentation of the options available in this interface.[13] Sub-sections that are useful in assessing the status of preserved content include *Archival Units*, *Crawl Status*, *Hash Queue*, *Polls*, and *Votes*.

The administrative UI does not provide access to the preserved content itself; this function is provided by the audit proxy. The audit proxy is for small-scale testing and auditing by the node administrators. It constrains access only to the PLN contents stored on the given node; it does not pass through available source content that has not yet been retrieved. The LOCKSS team has created documentation on configuring the proxy using proxy auto-configuration (PAC) files.[14] Though the audit proxy is configured at the node level, some aspects are controlled in the PLN properties file.[15]

[11] http://www.lockss.org/locksswiki/index.php/Plugin_Tool
[12] http://www.lockss.org/lockssdoc/gamma/daemon/paramdoc.html#org.lockss.ui.port
[13] http://metaarchive.org/public/resources/Lockss_UI_Guide.pdf#page=8
[14] http://www.lockss.org/support/use-a-lockss-box/view-your-preserved-content/proxy-integration/
[15] http://www.lockss.org/lockssdoc/gamma/daemon/paramdoc.html#org.lockss.proxy.audit.bindAddrs

Audit preservation

Once a given node has harvested all of the content designated for preservation within the PLN, the node operator can perform an audit and repair test to validate the expected operation of the LOCKSS software. Intentionally removing or modifying content stored on the node simulates bit damage to the managed AUs. The filesystem location of files associated with an AU can be viewed in the administrative UI in the *Daemon Status* > *Archival Units* sub-section, by clicking on a given AU.

If a file is observed missing, it can be restored on the node by crawling (if the content source is available and accessible) or from the other nodes, through the polling mechanism (if not). Crawls and polls take place automatically and periodically, but a node administrator can manually initiate either through the administrative UI by navigating to *Debug Panel*[16] and clicking on *Start Crawl* or *Start V3 Poll*, respectively.

Secure the PLN

Poor security is a preservation risk. While a broad array of security features can be enabled and configured through the properties, the LOCKSS team recommends three as a baseline: IP filtering, encrypting network access to the administrative UI, and encrypting inter-node polling communications.

The IP address and network details gathered during the process of configuring each of the nodes[17] can be used to configure IP filtering in the properties. Consult with the LOCKSS team to have this information entered into the LOCKSS-hosted properties file or for guidance on which specific properties to update in a PLN-hosted properties file. This will ensure that PLN communications can only take place between whitelisted IP addresses and networks.

By default, the LOCKSS administrative UI is configured with basic HTTP access authentication, which means unprotected transmission of user credentials. A single property can be modified to simultaneously require HTTPS access, enable form authentication, and specify other user account-related parameters.[18] The same documentation referenced in the preceding footnote

[16] http://metaarchive.org/public/resources/Lockss_UI_Guide.pdf#page=7
[17] http://www.lockss.org/docs/LOCKSS-Linux6-Install.pdf#page=6
[18] http://www.lockss.org/locksswp/wp-content/uploads/2012/02/LOCKSS-Network-Administration-LOCKSS.pdf#page=4

provides additional detail on management of cryptographic keys and certificates, more granular user account policies, and user role privileges.

To encrypt the inter-node polling communications, the PLN manager must create and distribute Java KeyStores to each node. Each node receives two KeyStores: one containing its own private key and another containing public certificates for each of the PLN nodes, plus a password file containing the secret password for the private key. Once installed on a given node, restart the daemon and confirm that encrypted network communications are enabled through the administrative UI.

Coordinate

The strength of the community of organizations participating in the PLN is at least as important as the shared technical infrastructure in ensuring the durability of the designated content. Many PLNs foster their communities by setting up a listserv to facilitate communication among node administrators, creating documentation, scheduling regular meetings, and, eventually, determining a governance structure. A representative from the LOCKSS team may participate in some of these early community activities, if desired; they can provide perspective from the founding of other PLNs.

Additional resources

One of the advantages of the PLN model is that the PLN community itself works together to define and share best practices that are applicable across PLNs. The following resources produced and maintained by the PLN community may be helpful in gaining a deeper understanding of some of the topics covered here, as well as other aspects of PLNs:

- The LOCKSS UI Guide[19] by the MetaArchive Cooperative PLN describes the LOCKSS administrative UI pages and form fields. The administrative UI has changed somewhat since the documentation was created but much of it is the same.
- The LOCKSS Software wiki page[20] by the Alabama Digital Preservation Network provides a deeper explanation of the LOCKSS software architecture and operation.
- The PLNwiki Technical Manual[21] by the Alabama Digital Preservation Network provides additional detail on how to set up and manage a PLN.

[19] http://metaarchive.org/public/resources/Lockss_UI_Guide.pdf
[20] http://www.adpn.org/wiki/LOCKSS_Software

[21] http://plnwiki.lockss.org/wiki/index.php/LOCKSS_Technical_Manual