LOCKSS System Manual

LOCKSS Program

2023-09-28

LOCKSS 2.0-ALPHA4 SYSTEM MANUAL

Welcome to the LOCKSS 2.0-alpha4 System Manual.

Latest release: 2.0.43-alpha4 (2021-12-15)
First release: 2.0.41-alpha4 (2021-06-28)
System manual last built: 2023-09-28.

Attention: Security advisories: CVE-2021-44228 ("Log4Shell"), CVE-2021-45046, CVE-2021-4104, CVE-2021-45105, CVE-2021-44832

LOCKSS 2.0-alpha4 and the custom Solr and OpenWayback containers it includes are affected.

See *Security Advisories*.

CHAPTER

ONE

INTRODUCTION

The LOCKSS system is a distributed digital preservation software system developed by the LOCKSS Program, a division of the Digital Library Systems and Services department at Stanford University Libraries.

The 2.x series of the LOCKSS system stems from the LAAWS (LOCKSS Architected As Web Services) initiative, an ambitious modernization project that includes rewriting the classic LOCKSS daemon as a suite of containerized components, funded in part by a grant from the Andrew W. Mellon Foundation.

This version, LOCKSS 2.0-alpha4, is the fourth preview release on the road to LOCKSS 2.0.

1.1 System Prerequisites

1.1.1 Host

The LOCKSS system runs in a **64-bit Linux** host (physical or virtual).

See the next section (*Operating System*) for operating system choices.

1.1.2 CPU

The CPU requirements depend on which components of the LOCKSS system you choose to run. We recommend at least **4 CPU cores**, preferably 8.

1.1.3 Memory

Likewise, the memory requirements also depend on which components of the LOCKSS system you choose to run. We recommend at least **8 GB** of memory, preferably 16 GB.

1.1.4 Storage

LOCKSS makes use of several storage areas. During configuration, the administrator must specify the location of these storage areas by supplying one or more directory paths. The default is to put all storage under a single directory, but different types of storage have different size and performance requirements and on a large system, if different types of storage are available it may be advantageous to place the storage areas on different devices:

- **Content data storage:** This is where all the preserved content is stored, along with an index and several databases[1]. Many LOCKSS systems preserve a large amount of content and it has become common to use network-attached storage for this. LOCKSS' audit activities result in nearly continuous reading of content from

[1] Unless using external Solr and/or PostgreSQL servers.

storage; this may impact the storage server's performance, and a busy or non-performant storage server may impact LOCKSS' performance. For the repository service, multiple storage areas may be specified, and more can be added later.

The amount of space required depends on the amount of content that will be preserved. The content is efficiently stored in large, compressed WARC files. Unlike LOCKSS 1.x, inode usage is very low.

- **Log data storage:** Service logs will be written to subdirectories of this path. There is usually no reason to enter a different path unless you have specific logging requirements.
- **Temporary data storage:** The LOCKSS software makes heavy use of temporary storage, and we recommend that temporary directories be placed on a filesystem with relatively low latency. If the content storage directories are on network storage (for example NFS), system performance may be improved by using a local directory.

  Caution: Depending on the characteristics of the preservation activities undertaken by the system, in some circumstances content processing may require a substantial amount of temporary space, up to tens of gigabytes. Do not use a RAM-based `tmpfs` volume, or a directory in a space-constrained partition, for temporary data storage.

What's the Minimum for Experimentation?

To review the installation instructions and test the installation of K3s in various operating systems, we routinely install and bring up minimal LOCKSS 2.0-alpha4 systems, with no metadata services or Web replay engines, and with empty embedded Postgres and Solr databases, in Vagrant virtual machines with Virtualbox using 2 CPU cores and 3 GB of memory. These minimal VMs would not support a production load, but it can be a useful tool to try out the installation instructions or evaluate the system.

1.2 Operating System

The LOCKSS system requires a **64-bit Linux** host (physical or virtual) compatible with K3s, a lightweight Kubernetes distribution by Rancher.

Rancher states[1] that "K3s is expected to work on most modern Linux systems", and that "Some OSs have specific requirements" (which are documented here and integrated into the `lockss-installer` scripts).

Flavors of Linux we have successfully tested include:

- AlmaLinux 8.4, 8.3.
- Arch Linux (rolling release).
- CentOS 8.3, 8.2, 8.1, 8.0, 7.9, 7.8, 7.7, 7.6, 7.5, 7.4, 7.3.
- Debian 10.9, 10.8, 10.7, 10.6, 10.5, 10.4, 10.3, 10.2, 10.1, 10.0, 9.13, 9.12, 9.11, 9.9, 9.8, 9.7, 9.6, 9.5, 9.4, 9.3, 9.2, 9.1, 9.0.
- Fedora 34, 33, 32, 31, 30, 29, 28.
- Linux Mint 20.1, 20.0, 19.3, 19.2, 19.1, 19.0.
- OpenSUSE Leap 15.2, 15.1, 15.0.

[1] Reference: https://rancher.com/docs/k3s/latest/en/installation/installation-requirements/#operating-systems

- Oracle Linux 8.3, 8.2, 8.1, 7.9, 7.8, 7.7, 7.6.
- RHEL 8.3.
- Rocky Linux 8.4.
- Ubuntu 21.04, 20.10, 20.04 LTS, 19.10, 19.04, 18.10, 18.04 LTS.

but the LOCKSS system can likely be installed successfully on slightly different versions of the Linux flavors above, as well as other Linux flavors altogether.

We currently recommend CentOS 7.

CHAPTER

TWO

UPGRADING FROM LOCKSS 2.0-ALPHA3

This chapter describes how to upgrade the LOCKSS system from 2.0-alpha3 to 2.0-alpha4. If you have been running LOCKSS 2.0-alpha3 (or an earlier 2.x version), we thank you for helping us bring LOCKSS 2.0 closer to fruition through your testing and feedback.

- **If you are installing the LOCKSS 2.0 system for the first time**, skip this chapter and proceed to *Installing the LOCKSS System*.
- **Before you begin upgrading from 2.0-alpha3**, we strongly recommend you first bring your operating system up to date by applying security updates and upgrading installed packages.
- Please note that the upgrade process includes re-running the LOCKSS configuration tool, which **will require you to re-enter the PostgreSQL database password**.

Chapter Overview

2.1 Stopping LOCKSS 2.0-alpha3

The first step is to stop the LOCKSS 2.0-alpha3 system.

Log in as the `lockss` user and run this command in the `lockss-installer` directory:

```
scripts/stop-lockss
```

2.2 Updating the LOCKSS Installer

The next step is to update the LOCKSS installer to its latest version from GitHub.

While still logged in as `lockss` and in the `lockss-installer` directory, run these commands:

```
git checkout master

git pull
```

2.3 Adjusting File Permissions

Some files stored on disk in 2.0-alpha3 and prior are owned by `root`.

While still logged in as `lockss` and in the `lockss-installer` directory, run this command:

```
scripts/upgrades/fix-permissions
```

You may be prompted for the `lockss` user's **`sudo`** password.

2.4 Uninstalling MicroK8s and Snap

After 2.0-alpha3, Microk8s, and therefore Snap (except on Ubuntu), are no longer needed.

While still logged in as `lockss` and in the `lockss-installer` directory, run this command:

```
scripts/upgrades/uninstall-microk8s
```

You may be prompted for the `lockss` user's **`sudo`** password.

The **`uninstall-microk8s`** script will ask you to confirm before uninstalling Snap (**`snapd`**). Enter `Y` for "yes" and `N` for "no", or simply hit `Enter` to accept the suggested answer in square brackets.

Caution: On Ubuntu, Snap is used natively by the operating system and should not be uninstalled.

2.5 Restoring Packet Filters

A short-term requirement of 2.0-alpha3 was that frontends to **`iptables`** like **`firewalld`** or **`ufw`** be disabled, to work more smoothly with MicroK8s. This is no longer necessary in most cases, and you should return your system's **`firewalld`** or **`ufw`** to its original state.

If you had disabled **`firewalld`** or **`ufw`** to run 2.0-alpha3, select your operating system below and follow the corresponding instructions while still logged in as `lockss`:

CentOS

To check if **`firewalld`** is running, run this command:

```
sudo firewall-cmd --state
```

If it is not running, enable and start it by running this command:

```
sudo systemctl enable --now firewalld
```

Debian

This operating system does not have a firewall on by default, so no particular action is required.

Linux Mint

This operating system does not have a firewall on by default, so no particular action is required.

OpenSUSE

To check if **firewalld** is running, run this command:

```
sudo firewall-cmd --state
```

If it is not running, enable and start it by running this command:

```
sudo systemctl enable --now firewalld
```

RHEL

To check if **firewalld** is running, run this command:

```
sudo firewall-cmd --state
```

If it is not running, enable and start it by running this command:

```
sudo systemctl enable --now firewalld
```

Ubuntu

To check if **ufw** is running, run this command:

```
sudo ufw status
```

If it is not active, enable and start it by running this command:

```
sudo ufw enable
```

2.6 Revoking the Extra Privileges of the `lockss` User

Another short-term requirement of 2.0-alpha3 was that the `lockss` user have a login password set and be allowed access to **sudo**. This is no longer needed and we strongly recommend you revoke these extra privileges for better security.

Follow the following steps:

1. Log out of the `lockss` user account. You can do this by typing `exit` or `logout`, or hitting `Ctrl + D` on the keyboard.
2. Log in as a privileged user other than `lockss` privileged user who can become root via **sudo**[1].
3. To invalidate the login password of the `lockss` user, run this command:

   ```
   sudo usermod --lock lockss
   ```

[1] See *Running Commands as a Privileged User*.

4. To revoke the `lockss` user's access to **sudo**, select your operating system below and follow the corresponding instructions.

   CentOS

   To revoke the `lockss` user's access to **sudo**, run this command:

   ```
   sudo gpasswd --delete=lockss wheel
   ```

   Debian

   To revoke the `lockss` user's access to **sudo**, run this command:

   ```
   sudo gpasswd --delete=lockss sudo
   ```

   Linux Mint

   To revoke the `lockss` user's access to **sudo**, run this command:

   ```
   sudo gpasswd --delete=lockss sudo
   ```

   OpenSUSE

   To revoke the `lockss` user's access to **sudo**, run this command:

   ```
   sudo gpasswd --delete=lockss wheel
   ```

   RHEL

   To revoke the `lockss` user's access to **sudo**, run this command:

   ```
   sudo gpasswd --delete=lockss wheel
   ```

   Ubuntu

   To revoke the `lockss` user's access to **sudo**, run this command:

   ```
   sudo gpasswd --delete=lockss sudo
   ```

2.7 Next Steps

Next, you will need to install K3s, a lightweight Kubernetes environment to replace MicroK8s.

Proceed to the *Installing K3s* section of the *Installing the LOCKSS System* chapter, skipping over the earlier sections of the chapter that are not required in an upgrade situation (*Creating the lockss User*, *Installing Git*, *Downloading the LOCKSS Installer*).

Then simply continue following the manual from the *Installing K3s* section forward. In particular, you will need to re-run the configuration script (see *Configuring the LOCKSS System*).

CHAPTER

THREE

INSTALLING THE LOCKSS SYSTEM

This chapter describes how to install LOCKSS 2.0-alpha4 for the first time.

- **If you are upgrading from LOCKSS 2.0-alpha3**, see *Upgrading From LOCKSS 2.0-alpha3*.
- **If you are currently running the LOCKSS 1.x system**, for example version 1.75.7 of the LOCKSS daemon, you can use the instructions in this chapter to test the LOCKSS 2.0-alpha4 system on another machine, but it is not yet possible to migrate your classic system.
- **Before you begin installing the LOCKSS 2.x system**, we strongly recommend you first bring your operating system up to date by applying security updates and upgrading installed packages.

3.1 Creating the `lockss` User

The LOCKSS system runs under a system user named `lockss`, which is in a system group named `lockss`.

To create the `lockss` user and group, run this **`useradd`** command as `root`[1] :

```
useradd --system --user-group --create-home --shell=/bin/bash lockss
```

3.2 Installing Git

Git is a version control system, used to interact with code repositories. The LOCKSS Installer is available from GitHub, and you will need a Git client to download it.

Follow these instructions to install Git:

1. Run this command (as any user):

   ```
   git --version
   ```

2. If the output is a version number (for example `git version 2.31.1`), Git is already installed; however if you see an error message (for example `bash: git: command not found`), follow the instructions corresponding to your operating system below to install Git as `root`[1]:

   AlmaLinux

   To install Git, run this Dnf command (as `root`):

[1] See *Running Commands as root*.
[1] See *Running Commands as root*.

```
dnf --assumeyes install git
```

Arch Linux

To install Git, run this Pacman command (as `root`):

```
pacman -Sy --noconfirm git
```

CentOS

CentOS 7

To install Git, run this Yum command (as `root`):

```
yum --assumeyes install git
```

To install Git, run this Yum command (as `root`):

```
yum --assumeyes install git


.. group-tab:: CentOS 8

   .. include:: git-dnf.rst
```

To install Git, run this Dnf command (as `root`):

```
dnf --assumeyes install git
```

Debian

To install Git, run these Apt commands (as `root`):

```
apt update

apt install --assume-yes git
```

Fedora

To install Git, run this Dnf command (as `root`):

```
dnf --assumeyes install git
```

Linux Mint

To install Git, run these Apt commands (as `root`):

```
apt update

apt install --assume-yes git
```

OpenSUSE

To install Git, run these Zypper (as `root`):

```
zypper refresh

zypper --non-interactive install git
```

Oracle Linux

Oracle Linux 7

To install Git, run this Yum command (as `root`):

```
yum --assumeyes install git
```

To install Git, run this Yum command (as `root`):

```
yum --assumeyes install git


.. group-tab:: Oracle Linux 8

   .. include:: git-dnf.rst
```

To install Git, run this Dnf command (as `root`):

```
dnf --assumeyes install git
```

RHEL

RHEL 7

To install Git, run this Yum command (as `root`):

```
yum --assumeyes install git
```

To install Git, run this Yum command (as `root`):

```
yum --assumeyes install git


.. group-tab:: RHEL 8

   .. include:: git-dnf.rst
```

To install Git, run this Dnf command (as `root`):

```
dnf --assumeyes install git
```

Rocky Linux

To install Git, run this Dnf command (as `root`):

```
dnf --assumeyes install git
```

Ubuntu

To install Git, run these Apt commands (as `root`):

```
apt update

apt install --assume-yes git
```

3.3 Downloading the LOCKSS Installer

You will need to download the LOCKSS Installer from GitHub using Git.

Follow these instructions as the `lockss` user[1]:

1. Run this command:

```
git clone https://github.com/lockss/lockss-installer
```

Troubleshooting

On early CentOS 7 systems (for example CentOS 7.1), you may receive the error message `fatal: unable to access 'https://github.com/lockss/lockss-installer/': Peer reports incompatible or unsupported protocol version`. This is due to outdated network security libraries. Run the command `yum update -y curl nss nss-util nspr` as `root` to update them, and retry the **`git clone`** command.

2. Go into the `lockss-installer` directory created by the **`git clone`** command:

```
cd lockss-installer
```

Important: Many commands in this manual are run from this `lockss-installer` directory. Run the command `pwd` to display its full path (typically `/home/lockss/lockss-installer`) and make a note of it.

3. To avoid a harmless Git warning when updating the LOCKSS Installer from GitHub in the future, run this command:

```
git config --local pull.rebase true
```

3.4 Installing K3s

This section describes how to install K3s.

K3s is a lightweight Kubernetes distribution by Kubernetes vendor Rancher. K3s is compatible with most Linux flavors and comes with a convenient multi-OS installer.

The LOCKSS Installer provides **`install-k3s`**, a script that streamlines the installation and configuration of K3s for the purposes of running the LOCKSS system. To install, configure and check K3s:

1. In the `lockss` user's `lockss-installer` directory, run the following command[1] as `root`[2]:

```
scripts/install-k3s
```

(Typically, this is `/home/lockss/lockss-installer/scripts/install-k3s`.)

[1] See *Running Commands as the lockss User*

[1] If you invoke **`install-k3s`** with the option `--k3s-data-dir=`*`DATADIRPATH`*, the directory path *`DATADIRPATH`* will be used as your answer to the K3s state data directory question without an interactive prompt.

If you invoke **`install-k3s`** with the option `--assume-yes`, **`install-k3s`** will assume that the answer to every interactive yes/no question is `Y` for "yes", and that the answer to the K3s state data directory question is the default `/var/lib/rancher/k3s` (unless you also used the `--k3s-data-dir`, which takes precedence).

[2] See *Running Commands as root*.

2. The **`install-k3s`** script is capable of detecting a variety of problematic situations with **`iptables`**, **`firewalld`**, and **`ufw`** (firewall). If applicable, **`install-k3s`** will display warning messages and prompt you to confirm before taking corrective action. Enter `Y` for "yes" and `N` for "no", or simply hit `Enter` to accept the proposed answer (displayed in square brackets).

   Caution: If you opt out of the proposed remediations, **`install-k3s`** will proceed, but K3s is likely to malfunction without external intervention.

   Troubleshooting

   For details, see:

   - *Troubleshooting iptables*
   - *Troubleshooting firewalld*
   - *Troubleshooting ufw*

3. The **`install-k3s`** script is also capable of detecting a problematic situation with DNS resolution. If applicable, **`install-k3s`** will display a warning message and the following prompt:

   IP address(es) of DNS resolvers, separated by ';'

   Enter a semicolon-separated list of DNS server IP addresses that are *not* loopback addresses. A suggested default will be offered to you in square brackets, consisting of non-loopback addresses collected from your machine's `resolv.conf` files; you can simply hit `Enter` to accept the suggested default.

   Troubleshooting

   For details, see *Troubleshooting CoreDNS*.

4. You will be prompted for a K3s state data directory:

   K3s state data directory: [/var/lib/rancher/k3s]

   K3s stores state data in `/var/lib/rancher/k3s` by default, but if `/var` is space-limited, you should specify a different directory as the K3s state data directory will grow to at least 5-10GB. Enter a directory path of your choice followed by `Enter`, or simply hit `Enter` to accept the default.

5. If Rancher's K3s install script (which is invoked by **`install-k3s`**) cannot recover from an error condition, it may display an error message with a suggested remediation before exiting. If applicable, perform the recommended action and re-run **`install-k3s`**.

   Troubleshooting

   For details, see *Troubleshooting the K3s Installer*.

6. The LOCKSS Installer provides **`check-k3s`**, a tool to check that K3s is running and resolving DNS names properly. In the `lockss` user's `lockss-installer` directory, run this command as the `lockss` user[3]:

```
scripts/check-k3s
```

[3] See *Running Commands as the lockss User*.

If all tests succeed, the last line of output will be `STATUS: pass`.

Troubleshooting

If **`check-k3s`** fails (for example `STATUS: fail` or `STATUS: fail (3 errors)`) or keeps retrying the same step many times without succeeding, see *Troubleshooting K3s*.

7. K3s comes with **`k3s check-config`**, a configuration and system checker. Run the following command as `root`Page 14, 2:

```
k3s check-config
```

If all tests succeed, the last line of output will be `STATUS: pass`.

Important: On some operating systems, this checker may report an **`iptables`** error message similar to `iptables v1.8.4 (nf_tables): should be older than v1.8.0 or in legacy mode (fail)`, even though nothing is wrong. This is a known bug in the version of K3s used by LOCKSS 2.0-alpha4. If **`check-k3s`** ran successfully previously, you may ignore this spurious error message. For details, see *iptables should be older than v1.8.0 or in legacy mode*.

Troubleshooting

If this checker fails (for example `STATUS: 1 (fail)`), see *Troubleshooting the K3s Configuration Checker*.

3.5 Checking the System

After *installing the LOCKSS system*, you can confirm the status of installed components by running this command as the `lockss` user[1] in the `lockss` user's `lockss-installer` directory:

```
scripts/check-sys
```

The script run a few pre-flight checks.

[1] See *Running Commands as the lockss User*

CHAPTER

FOUR

CONFIGURING THE LOCKSS SYSTEM

After installing the LOCKSS system, configure the system with the **configure-lockss** script by running this command in the lockss user's lockss-installer directory as lockss[1]:

```
scripts/configure-lockss
```

If you have experience with classic LOCKSS daemon version 1.x, this is the equivalent of **hostconfig**.

Some notes about using **configure-lockss**:

- When run the first time, some of the questions asked by the script will have a suggested or default value, displayed in square brackets; hit `Enter` to accept the suggested value, or type the correct value and hit `Enter`.
- Any subsequent runs will use the previous values as the default value; review and hit `Enter` to leave unchanged.
- Password prompts will not display the previous value but can still be left unchanged with `Enter`.

Chapter Overview

[1] See *Running Commands as the lockss User*.

4.1 Network Settings

4.1.1 Hostname

Prompt: *Fully qualified hostname (FQDN) of this machine*

Enter the machine's fully-qualified hostname (meaning with its domain name), for example `locksstest.myuniversity.edu`.

4.1.2 IP Address

Prompt: *IP address of this machine*

If the machine is publicly routable, meaning it has an IP address that can be used to identify it over the Internet, enter the publicly routable IP address. Otherwise, if the machine is accessible via network address translation (NAT), meaning it has an IP address that is valid only on your local network but it can be reached from the Internet via a NAT router, enter the internal IP address.

4.1.3 Network Address Translation

1. Prompt: *Is this machine behind NAT?*

   If the machine is publicly routable, enter `N`; otherwise, if the machine is not publicly routable but will be accessible via network address translation (NAT), enter `Y`.

2. If you answered `Y`, you will be asked an additional configuration question:

   External IP address for NAT

   Enter the publicly routable IP address of the NAT router.

4.1.4 Initial UI Subnet

Prompt: *Initial subnet(s) for admin UI access*

Enter a semicolon-separated list of subnets in CIDR or mask notation that should initially have access to the Web user interfaces (UI) of the system. The access list can be modified later via the UI.

4.1.5 Container Subnet

1. If **`configure-lockss`** detects a discrepancy between a previously used subnet for inter-container communication in the system and the subnet it would choose now, you may either see the warning:

   Container subnet has changed from <former_subnet> to <new_subnet>

   or be asked the question:

   Container subnet was <former_subnet>, we think it should now be <new_subnet>. Do you want to change it?

   in which case you should enter `Y` (recommended) or `N`.

2. Prompt: *LOCKSS subnet for inter-service access control*

   Enter the subnet used for inter-container communication. We recommend accepting the proposed value by hitting `Enter`.

4.1.6 LCAP Port

Prompt: *LCAP V3 protocol port*

Enter the port on the publicly routable IP address that will be used to receive LCAP (LOCKSS polling and repair) traffic. Historically, most LOCKSS nodes use `9729`.

4.2 Mail Settings

4.2.1 Mail Relay

Prompt: *Mail relay for this machine*

Enter the hostname of this machine's outgoing mail server, for example `smtp.myuniversity.edu`.

4.2.2 Mail Relay Credentials

1. Prompt: *Does the mail relay <mailhost> need a username and password?*

   If the outgoing mail server does not require password authentication, enter `N`; otherwise, enter `Y`.

2. If you answered `Y`, you will be asked additional configuration questions:

   1. Prompt: *User for <mailhost>*

      Enter a username for the mail server.

   2. Prompt: *Password for <mailuser>@<mailhost>*

      Enter the password for the username on the mail server.

   3. Prompt: *Password for <mailuser>@<mailhost> (again)*

      Re-enter the password for the username on the mail server. If the two passwords do not match, the password will be asked again.

4.2.3 Administrator Email

Prompt: *E-mail address for administrator*

Enter the e-mail address of the person or team who will administer the LOCKSS system on this machine.

4.3 Preservation Network Settings

4.3.1 Configuration URL

1. Prompt: *Configuration URL*

   Accept the default (`http://props.lockss.org:8001/demo/lockss.xml`) if you are not running your own LOCKSS network; otherwise, enter the URL of the LOCKSS network configuration file provided by your LOCKSS network administrator.

2. If the configuration URL begins with `https:`, you will be asked additional configuration questions:

1. Prompt: *Verify configuration server authenticity?*

   Enter `Y` if you would like to check the authenticity of the configuration server using a custom keystore; otherwise enter `N`.

2. If you answered `Y`, you will be asked an additional configuration question:

   Server certificate keystore

   Enter the path of a Java keystore used to vverify the authenticity of the configuration server.

4.3.2 Configuration Proxy

Prompt: *Configuration proxy (host:port)*

If the configuration URL can be reached directly, leave this blank; otherwise, if a proxy server is required to reach the configuration URL, enter its host and port in *`host:port`* format (for example `proxy.myuniversity.edu:8080`).

4.3.3 Preservation Groups

Prompt: *Preservation group(s)*

Accept the default (`demo`) if you are not running your own LOCKSS network; otherwise, enter a semicolon-separated list of LOCKSS network identifiers as provided by your LOCKSS network administrator, for example `ournetwork` or `prod;usdocspln`.

4.4 Storage Areas

The LOCKSS system needs storage areas to store data:

- One or more **content data storage areas** to store preserved content, as well as several databases.
- A **log data storage area** to store log files.
- A **temporary data storage area** to store temporary files.

Depending on your host system's layout, these storage areas may all be the same, or all be different mount points or paths.

Subdirectories will be created in each storage area to fit the needs of a system component; for example `lockss-stack-cfg-data` is the LOCKSS configuration service's content data directory in the content data storage areas, and `lockss-stack-repo-logs` is the LOCKSS repository service's log data directory in the log data storage area.

4.4.1 Content Data Storage Areas

1. Prompt: *Root path for primary content data storage*

   Enter the full path of a directory to use as the root of the main storage area of the LOCKSS system, where preserved content will be stored along with several databases. It is the analog of `/cache0` in the classic LOCKSS system.

2. Prompt: *Use additional directories for content data storage?*

   If you want to use more than one filesystem to store preserved content, enter `Y`; otherwise, enter `N`.

3. If you answered `Y`, you will be asked an additional configuration question:

   Root path for additional content data storage <count> (q to quit)

   On each line, enter the full path of a directory to use as the root of an additional storage area, and enter `q` when done.

4.4.2 Log Data Storage Area

Prompt: *Root path for log data storage*

This directory is used as the root of the storage area for log files in the LOCKSS system. Accept the default (same directory as the content data storage directory root) by hitting `Enter`, or enter a custom path.

4.4.3 Temporary Data Storage Area

Prompt: *Root path for temporary data storage (local storage preferred)*

This directory is used as the root of the storage area for temporary files in the LOCKSS system. Accept the default (same directory as the content data storage directory root) by hitting `Enter`, or enter a custom path.

Tip: The LOCKSS software makes heavy use of temporary storage, and we recommend that temporary directories be placed on a filesystem with relatively low latency. If the content data storage directories are on network storage (for example NFS), system performance may be improved by supplying a local directory for temporary data storage.

Caution: Depending on the characteristics of the preservation activities undertaken by the system, in some circumstances content processing may require a substantial amount of temporary space, up to tens of gigabytes. Do not use a RAM-based `tmpfs` volume, or a directory in a space-constrained partition.

4.5 Web User Interface Settings

1. Prompt: *User name for web UI administration*

   Enter a username for the primary administrative user in the LOCKSS system's Web user interfaces.

2. Prompt: *Password for web UI administration user <uiuser>*

   Enter a password for the primary administrative user.

3. Prompt: *Password for web UI administration user <uiuser> (again)*

   Re-enter the password for the primary administrative user. If the two passwords do not match, the password will be asked again.

4.6 Database Settings

4.6.1 PostgreSQL

Prompt: *Use embedded LOCKSS PostgreSQL DB Service?*

Select **either** option A **or** option B:

A. Enter `Y` to use the **embedded PostgreSQL database**. This is recommended in most cases; a PostgreSQL database will be run and managed by the LOCKSS system internally. If you choose this option, see *Embedded PostgreSQL Database*.

B. Enter `N` to use an **external PostgreSQL database**. Select this option if you wish to use an existing PostgreSQL database at your institution or one that you run and manage yourself. If you choose this option, see *External PostgreSQL Database*.

4.6.1.1 Embedded PostgreSQL Database

If you select this option, you will be asked additional configuration questions:

1. Prompt: *Password for PostgreSQL database*

   Enter the password for the embedded PostgreSQL database.

   Warning: This prompt is used to record the PostgreSQL database password in the LOCKSS system's configuration. If you change the value of the PostgreSQL database password here without actually changing the PostgreSQL database password, the LOCKSS system components will no longer be able to connect to the PostgreSQL database. See *Working with PostgreSQL* for details.

2. Prompt: *Password for PostgreSQL database (again)*

   Re-enter the password for the embedded PostgreSQL database. If the two passwords do not match, the password will be asked again.

3. Complete the *Solr* section next.

4.6.1.2 External PostgreSQL Database

If you select this option, you will be asked additional configuration questions:

1. Prompt: *Fully qualified hostname (FQDN) of PostgreSQL host*

   Enter the hostname of the external PostgreSQL database, for example `postgres.myuniversity.edu`.

2. Prompt: *Port used by PostgreSQL host*

   Enter the port where the external PostgreSQL database can be reached, for example `5432`.

3. Prompt: *Schema for PostgreSQL service*

   Enter the schema name to be used by the LOCKSS system. The schema name used in the embedded PostgreSQL database is `LOCKSS`, but your database administrator may assign a different schema name to you.

4. Prompt: *Database name prefix for PostgreSQL service*

   Enter the prefix to use for any LOCKSS-related database names in the schema. The database name prefix in the embedded PostgreSQL databse is `Lockss` (note the uppercase/lowercase), but your database administrator may assign a different database name prefix.

5. Prompt: *Login name for PostgreSQL service*

   Enter the username for the external PostgreSQL database. The username in the embedded PostgreSQL database is `LOCKSS`, but your database administrator may assign a different username to you.

6. Prompt: *Password for PostgreSQL database*

   Enter the password for the username in the external PostgreSQL database.

   Warning: This prompt is used to record the PostgreSQL database password in the LOCKSS system's configuration. If you change the value of the PostgreSQL database password here without actually changing the PostgreSQL database password, the LOCKSS system components will no longer be able to connect to the PostgreSQL database. Contact your PostgreSQL database administrator for details.

7. Prompt: *Password for PostgreSQL database (again)*

   Re-enter the password for the username in the external PostgreSQL database. If the two passwords do not match, the password will be asked again.

8. Complete the *Solr* section next.

4.6.2 Solr

Prompt: *Use embedded LOCKSS Solr Service?*

Select **either** option A **or** option B:

A. Enter `Y` to use the **embedded Solr database**. This is recommended in most cases; a Solr database will be run and managed by the LOCKSS system internally. If you choose this option, see *Embedded Solr Database*.

B. Enter `N` to use an **external Solr database**. Select this option if you wish to use an existing Solr database at your institution or one that you run and manage yourself. If you choose this option, see *External Solr Database*.

4.6.2.1 Embedded Solr Database

If you select this option, you will be asked additional configuration questions:

1. Prompt: *User name for LOCKSS Solr access*

   Enter the username for the embedded Solr database.

2. Prompt: *Password for LOCKSS Solr access*

   Enter the password for the username in the embedded Solr database.

3. Prompt: *Password for LOCKSS Solr access (again)*

   Re-enter the password for the username in the embedded Solr database. If the two passwords do not match, the password will be asked again.

4. Complete the *Metadata Query Service* section next.

4.6.2.2 External Solr Database

If you select this option, you will be asked additional configuration questions:

1. Prompt: *Fully qualified hostname (FQDN) of Solr host*

   Enter the hostname of the external Solr database server, for example `solr.myuniversity.edu`.
2. Prompt: *Port used by Solr host:*

   Enter the port used by the database server on the Solr host, for example `8983`.
3. Prompt: *Solr core repo name:*

   Enter name of the Solr core for the LOCKSS repository. The Solr core name used in the embedded Solr database is `lockss-repo`, but your database administrator may assign a different Solr core name.
4. Prompt: *User name for LOCKSS Solr access*

   Enter the username for the external Solr database.
5. Prompt: *Password for LOCKSS Solr access*

   Enter the password for the username in the external Solr database.
6. Prompt: *Password for LOCKSS Solr access (again)*

   Re-enter the password for the username in the external Solr database. If the two passwords do not match, the password will be asked again.
7. Complete the *Metadata Query Service* section next.

4.7 LOCKSS Services

4.7.1 Metadata Query Service

Prompt: *Use LOCKSS Metadata Query Service?*

Enter `Y` if you want the metadata query service to be run, otherwise `N`.

4.7.2 Metadata Extraction Service

Prompt: *Use LOCKSS Metadata Extraction Service?*

Enter `Y` if you want the metadata extraction service to be run, otherwise `N`.

4.8 Web Replay Settings

4.8.1 Pywb

Prompt: *Use LOCKSS Pywb Service?*

Enter `Y` to run an embedded Pywb engine for Web replay; otherwise, enter `N`.

4.8.2 OpenWayback

1. Prompt: *Use LOCKSS OpenWayback Service?*

   Enter `Y` to use an embedded OpenWayback engine for Web replay; otherwise, enter `N`.

2. If you answered `Y`, you will be asked an additional configuration question:

   Okay to turn off authentication for read-only requests for LOCKSS Repository Service?

   OpenWayback currently does not supply user credentials when reading content from the LOCKSS repository, so the repository must be configured to respond to unauthenticated read requests. Enter `Y` to accept this, otherwise you will see the warning:

   Not enabling OpenWayback Service

   and OpenWayback will not be run.

4.9 Final Steps

1. Prompt: *OK to store this configuration?*

   Enter `Y` if the configuration values are to your liking; otherwise, enter `N` to make edits.

2. If you answer `Y`, **`configure-lockss`** will perform the final configuration steps. You may be asked to confirm before directories are created for the first time:

   <directory> does not exist; shall I create it?

   or before directory permissions are changed:

   <directory> is not writable; shall I chown it?

   In each case, enter `Y` for "yes" and `N` for "no".

CHAPTER

FIVE

RUNNING THE LOCKSS SYSTEM

The commands in this section are all run as the `lockss` user[1] in the `lockss` user's `lockss-installer` directory.

5.1 Starting the LOCKSS System

Run `scripts/start-lockss`. This script will call in turn:

- `scripts/generate-lockss`: This script takes your configuration data and turns it into a set of configuration files containing the right values.
- `scripts/assemble-lockss`: This script puts the configuration files and puts them in the right places, and ensures that all storage volumes are ready for use (creating them if necessary).
- `scripts/deploy-lockss`: This script deploys your LOCKSS stack by invoking Kubernetes.

The **`start-lockss`** accepts some options:

`--update` (**`-u`**)
: Force the system to check for newer container images of the system's components (LOCKSS services, embedded databases, embedded Web replay engines...) before deploying the system to Kubernetes.

`--wait` (**`-w`**)
: After deploying the system to Kubernetes and waiting for the system's containers to come up, additionally wait for an internal signal from the system that the system's components are fully initialized. (Currently this internal signal comes from the poller service.)

5.2 Shutting down the LOCKSS System

Run `scripts/stop-lockss`.

[1] See *Running Commands as the lockss User*

5.3 Restarting a Running LOCKSS System

Run `scripts/restart-lockss`.

The **`restart-lockss`** accepts the same options as **`start-lockss`**.

5.4 Removing a Configured LOCKSS System

To remove all configurations, volumes and networks configured by the LOCKSS system in Kubernetes, run `scripts/uninstall-lockss`. This will **not** remove files from the persistent store.

CHAPTER

SIX

USING THE LOCKSS SYSTEM

This chapter describes how to use the LOCKSS system.

6.1 Using the LOCKSS Configuration Service

Note: This page is under construction.

6.1.1 Accessing the Web User Interface

If you are already connected to the Web user interface (UI) of another component of the LOCKSS System, click *Config Service* in the top-left menu.

Alternatively, if your primary hostname is <*HOST*>, you can use your browser to connect to the LOCKSS Configuration Service Web user interface (UI) at `http://<HOST>:24621`.

Enter your Web UI username and password to login if prompted.

6.1.2 Adding Archival Units

To add AUs to the system for preservation:

1. In the top-right menu, click *Journal Configuration*.
2. In the center menu, click *Add AUs*.
3. Select one or more collections of AUs by selecting the checkbox next to the appropriate collection.
4. Click the *Select AUs* button. It may take a bit of time (60+ seconds) for the next screen to appear, while the list of AUs is built.
5. Select one or more AUs from the AU list. You may click *Select All* if you would like to select all AUs. If you choose to use select all AUs, please note that the next step may take some time to load.
6. Click the *Add Selected AUs* button. The time it takes for the page to refresh depends on the number of AUs added. Give the LOCKSS system some time to load the AUs and reload the page before moving on.
7. A screen will show a list of added AUs. Crawling of these new AUs will start automatically -- no further action is necessary unless prompted by a footnote next to an AU's name.

6.1.3 Configuring a Crawl Proxy

If Web crawls must be routed through a Web proxy:

1. In the top-right menu, click *Content Access Options*.
2. In the center menu, click *Proxy Client Options*.
3. Select the *Proxy crawls* checkbox.
4. Enter the hostname and port of the Web proxy in the *HTTP Proxy host* and *Port* text areas, respectively.
5. Click the *Update Proxy Client* button.

6.1.4 Managing Access to the Web User Interfaces

This section is under construction.

6.2 Using the LOCKSS Crawler Service

Note: This page is under construction.

6.2.1 Accessing the Web User Interface

Note: Currently the crawler service is run as part of the poller service.

If you are already connected to the Web user interface (UI) of another component of the LOCKSS System, click *Crawler Service* in the top-left menu.

Alternatively, if your primary hostname is *<HOST>*, you can use your browser to connect to the LOCKSS Configuration Service Web user interface (UI) at `http://<HOST>:24631`.

Enter your Web UI username and password to login if prompted.

6.2.2 Monitoring Crawl Status in the System

The Crawl status of all configured AUs is available in the Archival Unit table

1. In the top-right menu, click *Daemon Status*.
2. Open the control in the middle of the screen that says *Overview* and select *Archival Units:guilabel:* from the drop down menu.
   - If prompted, enter your Username and Password again.
   - It will take a bit of time for the next screen to appear while the AU list is being built.
3. The Archival Units screen lists statistics for each configured AU

- the *Last Successful Crawl* column provides a timestamp of the most recent sucessful crawl.
- the *Last Crawl Start* column provides a timestamp of the last attempted crawl.
- the *Last Crawl Result* column provides the exit status of the last attempted crawl.

6.2.3 Causing an Archival Unit to Crawl

Archival units (AUs) that have been added to the system for preservation crawl periodically, but you can cause an AU to crawl on demand:

1. In the top-right menu, click *Debug Panel*.
2. Select an AU in the *AU Actions: select AU* drop-down list.
3. Click the *Start Crawl* button.
4. If the AU has crawled recently, you will be prompted to confirm that you wish to override the usual recrawl interval by clicking on the *Force Start Crawl* button.

6.2.4 Crawl Status Screen

To inspect the state of crawls, access the *Crawl Status* screen:

1. In the top-right menu, click *Daemon Status*.
2. In the center drop-down list, select *Crawl Status*. Alternatively, in the center overview, click on the second line, which says "*N* active crawls".

6.2.4.1 Top-Level Crawl Information

The top left of the Crawl Status table contains the number of active, successful or failed crawls, and a countdown until the next time the system will look at the AUs being preserved and pick some that are ready to crawl or recrawl.

6.2.4.2 Crawl Status Entry

Each line in the Crawl Status table contains:

- The name of the AU
- The type of crawl
- The start time of the crawl
- The duration of a finished or in-progress crawl
- The status of the crawl
- The number of bytes fetched over the network as part of the crawl
- The number of URLs fetched as part of the crawl
- The number of URLs parsed for more links
- The number of URLs remaining to be fetched as part of this crawl
- The number of URLs encountered as part of this crawl but excluded from being fetched
- The number of URLs fetched as part of the crawl, that received an HTTP Not Modified response
- The number of URLs that caused errors as part of this crawl
- The number of different content types encountered as part of the crawl

Most of these values can be clicked to see a list of the corresponding objects.

6.3 Using the LOCKSS Poller Service

Note: This page is under construction.

6.3.1 Accessing the Web User Interface

If you are already connected to the Web user interface (UI) of another component of the LOCKSS System, click *Poller Service* in the top-left menu.

Alternatively, if your primary hostname is *<HOST>*, you can use your browser to connect to the LOCKSS Configuration Service Web user interface (UI) at `http://<HOST>:24631`.

Enter your Web UI username and password to login if prompted.

6.3.2 Requesting Polls

This section is under construction.

6.3.3 Monitoring Polling and Voting

This section is under construction.

6.4 Using the LOCKSS Metadata Extraction Service

Note: This page is under construction.

6.4.1 Accessing the Web User Interface

If you are already connected to the Web user interface (UI) of another component of the LOCKSS System, click *Metadata Extraction Service* in the top-left menu.

Alternatively, if your primary hostname is *<HOST>*, you can use your browser to connect to the LOCKSS Configuration Service Web user interface (UI) at `http://<HOST>:24641`.

Enter your Web UI username and password to login if prompted.

6.4.2 Requesting Metadata Extraction

This section is under construction.

6.5 Using the LOCKSS Metadata Service

Note: This page is under construction.

6.5.1 Accessing the Web User Interface

If you are already connected to the Web user interface (UI) of another component of the LOCKSS System, click *Metadata Service* in the top-left menu.

Alternatively, if your primary hostname is <*HOST*>, you can use your browser to connect to the LOCKSS Configuration Service Web user interface (UI) at `http://<HOST>:24651`.

Enter your Web UI username and password to login if prompted.

6.5.2 Requesting Metadata Information

This section is under construction.

6.6 Replaying Web Content with Pywb

6.6.1 Accessing the Pywb User Interface

Given that your primary hostname is samp:*{<HOST>}*, you can use your browser to connect to the Pywb user interface (UI) at `http://<HOST>:8080`.

6.6.2 Replaying a URL

To view a URL from Pywb:

1. The Pywb screen provides a list of links to available collections. Click on the top-most collection which should be `/lockss`.
2. Enter the URL you want to replay in the URL search box.
3. Click the *Search* button.
4. Replay the most recent URL by clicking on the topmost entry of the third column.

6.6.3 Finding a URL From an AU to Replay

There are multiple ways to discover URLs belonging to an AU in the Configuration Service UI:

1. Obtaining a URL by clicking on "pages fetched" inside of crawl status
   - In the top-right menu, click *Daemon Status*.
   - Open the control in the middle of the screen that says *Overview* and select *Crawl Status* from the drop down menu.
   - Picking an AU from the active crawls, click on the number associated with *Pages Fetched* to bring up a list of URLs that have been crawled.

   - Copy one of the URLs and paste it in the Pywb interface as described previously.

2. Obtaining a Substance URL
   - In the top-right menu, click *Daemon Status*.
   - Open the control in the middle of the screen that says *Overview* and select *Archival Units* from the drop down menu. If prompted, enter your Username and Password again. It will take a bit of time for the next screen to appear while the AU list is being built.
   - Select an AU by clicking on the AU title in the first column.
   - Open the *Substance URLs* link
   - Copy one of the URLs and paste it in the Pywb interface as described previously.

6.7 Replaying Web Content with OpenWayback

6.7.1 Accessing the OpenWayback User Interface

Given that your primary hostname is samp:*{<HOST>}*, you can use your browser to connect to the Pywb user interface (UI) at `http://<HOST>:8080/wayback`.

6.7.2 Replaying a URL

To view a URL from OpenWayback:

1. Enter the URL you want to replay in the URL search box.
2. Click the *Search* button.
3. Select the *Year* or leave as :guilabel:`All*
4. Click *Take Me Back*.

6.7.3 Finding a URL From an AU to Replay

There are multiple ways to discover URLs belonging to an AU in the Configuration Service UI:

1. Obtaining a URL by clicking on "pages fetched" inside of crawl status
   - In the top-right menu, click *Daemon Status*.
   - Open the control in the middle of the screen that says *Overview* and select *Crawl Status* from the drop down menu.
   - Picking an AU from the active crawls, click on the number associated with *Pages Fetched* to bring up a list of URLs that have been crawled.
   - Copy one of the URLs and paste it in the OpenWayback interface as described previously.
2. Obtaining a Substance URL
   - In the top-right menu, click *Daemon Status*.
   - Open the control in the middle of the screen that says *Overview* and select *Archival Units* from the drop down menu. If prompted, enter your Username and Password again. It will take a bit of time for the next screen to appear while the AU list is being built.
   - Select an AU by clicking on the AU title in the first column.

- Open the *Substance URLs* link
- Copy one of the URLs and paste it in the OpenWayback interface as described previously.

6.8 Using the Kubernetes Dashboard

Kubernetes comes with the Kubernetes Dashboard, a web-based user interface (UI).

To facilitate installing and interacting with the Kubernetes Dashboard, the LOCKSS Installer offers the **`dashboard-util`** script.

6.8.1 Installing the Kubernetes Dashboard

To install the Kubernetes Dashboard, run this command[1]:

```
scripts/dashboard-util --install
```

If the installation succeeds, the program will also display the login URL and the bearer token.

6.8.2 Accessing the Kubernetes Dashboard

To access the Kubernetes Dashboard:

1. Create a secure channel to your K3s cluster with the following command:

   ```
   k3s kubectl proxy &
   ```

   Note: This command runs in the background "forever".

2. Obtain the login URL with the following command:

   ```
   scripts/dashboard-util --url
   ```

3. Obtain the login token with the following command:

   ```
   scripts/dashboard-util --token
   ```

4. Open a browser and go to the login URL.
5. Make sure the *Token* radio button is selected.
6. Copy and paste the login token into the *Enter token* text field.

[1] This command is relative to the `lockss` user's `lockss-installer` directory.

6.8.3 Using the Kubernetes Dashboard UI

When the dashboard comes up, it will be in the default namespace. Click on the namespace pull-down menu near the top and select the `lockss` namespace to see the LOCKSS components. If all of your deployments are running and ready, the three circles at the top should be green. In the left hand panel you can select the components you are interested in:

- Click on *Services* to see the cluster IP for each of the running services. You can click on a specific service to see more detailed information.
- Click on *Deployments* to see a list of services and their CPU and memory usage. You can access specific services and deployments from here.
- Click on *Pods*. This will give you information about all the pods running. Click on a pod of interest to obtain more granular information:

  View logs
  Since LOCKSS output logs are persisted to a local directory, there will be very little in the Kubernetes logs if the container came up without errors.

  Exec into pods
  This will open a terminal window into the container.

  Edit the pod resource
  This will allow you to view and edit the YAML file which was used to start the pod. The edit will not persist on restart.

  Delete the pod
  While this will delete the current pod, a new pod will be spawned by the deployment with a new pod ID.

6.8.4 Updating the Kubernetes Dashboard

To update the Kubernetes Dashboard to the most recent release, run this command[Page 35, 1]:

```
scripts/dashboard-util --update
```

6.8.5 Removing the Kubernetes Dashboard

To remove the Kubernetes Dashboard from the `kubernetes-dashboard` namespace, run this command[Page 35, 1]:

```
scripts/dashboard-util --remove
```

See Also

- Web UI (Dashboard) on the Kubernetes website.

CHAPTER

SEVEN

TROUBLESHOOTING THE LOCKSS SYSTEM

This chapter contains pages of additional information about troubleshooting the LOCKSS system.

7.1 Known Issues

This page was last updated: 2021-07-07.

- **Security**
  - In the "alpha" phase of development of LOCKSS 2.0, there are no access controls on Kubernetes' API. It is not accessible from outside the machine, but any local user can access the API, so they can stop the LOCKSS containers, change their contents, read secrets, etc. We plan to enable access controls in the "beta" phase.
  - In the Classic LOCKSS system (1.x), the LCAP SSL key could only be read by `root`, but now it can also be read by `lockss`.
- **DNS Resolution**

  K3s' default DNS cache timeout is 30 seconds, which results in enough repetitive upstream queries to trigger alarms at some institutions. One remediation is to change the CoreDNS configuration by editing its configmap.

  With K3s, changes made to CoreDNS's configmap with **`kubectl apply`** do not persist, because the configmap is constantly reloaded from `/var/lib/rancher/k3s/server/manifests/coredns.yaml`. Additionally, K3s overwrites the file with the defaults at startup, so changes there are not really persistent either.

  The LOCKSS Installer offers the script `scripts/coredns-cron-hack`, which sets the CoreDNS cache timeout to 30 minutes. It should be run once, as `root`, after each time K3s starts. Absent a good way to do that, it is harmless to run it periodically from `root`'s crontab. The recommended use is to copy it to a `root`-owned file in `/etc/cron.hourly`.
- **Harmless PID File Errors**

  The `stdout` log files of the various LOCKSS service containers contain the following error messages at startup:

  ```
  /usr/local/bin/docker-entrypoint: line 374: can't create /var/run/docker-entrypoint.
  ↪pid: Permission denied
  ```

  This is harmless and will be addressed in the next release of the system.

7.2 Troubleshooting `iptables`

K3s, the Kubernetes environment recommended for the LOCKSS system, does not currently work with **`iptables`** version 1.8.0 or later in `nf_tables` mode via Alternatives, for instance in some Debian or Ubuntu systems[1]. If **`configure-firewall`** (a script called by **`install-k3s`**) detects this situation, you will see a warning message and the following prompt[2]:

Switch iptables to legacy mode via Alternatives?

Enter `Y` for "yes" and `N` for "no", or simply hit `Enter` to accept the proposed answer (displayed in square brackets).

Caution: If you opt out of the proposed remediation, K3s may malfunction.

The remediation attempted by **`configure-firewall`** is equivalent to:

```
# Needed if ufw is installed and active
ufw disable

# Required
update-alternatives --set iptables /usr/sbin/iptables-legacy

# Required
update-alternatives --set ip6tables /usr/sbin/ip6tables-legacy

# Optional
update-alternatives --set arptables /usr/sbin/arptables-legacy

# Optional
update-alternatives --set ebtables /usr/sbin/ebtables-legacy

# Required
iptables --flush

# Needed if ufw is installed and was active
ufw enable
```

Tip: If your system did not initially need an adjustment for **`iptables`** at the time K3s was installed, but later does (for example because **`iptables`** is upgraded from a pre-1.8.0 version to version 1.8.0 or later), re-run this command in the `lockss` user's `lockss-installer` directory as a privileged user who can become root via **`sudo`**[3]:

```
scripts/configure-firewall
```

[1] References:

- https://rancher.com/docs/k3s/latest/en/known-issues/
- https://github.com/kubernetes/kubernetes/issues/71305
- https://github.com/k3s-io/k3s/issues/116
  - https://github.com/k3s-io/k3s/issues/116#issuecomment-624770403
- https://github.com/k3s-io/k3s/issues/703

[2] See *Installing K3s*.

[3] See *Running Commands as a Privileged User*.

7.3 Troubleshooting `firewalld`

If your system is running the **`firewalld`** firewall, it is necessary to add K3s' pod subnet (by default 10.42.0.0/16) and service subnet (by default 10.43.0.0/16) to **`firewalld`**'s `trusted` zone for K3s to work properly[1]. If **`configure-firewall`** (a script called by **`install-k3s`**) detects this situation, you will see a warning message and the following prompt[2]:

Add 10.42.0.0/16 and 10.43.0.0/16 to firewalld's trusted zone?

Enter `Y` for "yes" and `N` for "no", or simply hit `Enter` to accept the proposed answer (displayed in square brackets).

Caution: If you opt out of the proposed remediation, K3s may malfunction.

The remediation attempted by **`configure-firewall`** is equivalent to[3]:

```
firewall-cmd --permanent --zone=trusted --add-source=10.42.0.0/16

firewall-cmd --permanent --zone=trusted --add-source=10.43.0.0/16

firewall-cmd --reload
```

Tip: If your system did not initially use **`firewalld`** at the time K3s was installed, but later does (for example because **`firewalld`** becomes enabled), run this command in the `lockss` user's `lockss-installer` directory as a privileged user who can become root via **`sudo`**[4]:

```
scripts/configure-firewall
```

7.4 Troubleshooting `ufw`

If your system is running the **`ufw`** firewall, it is necessary to allow traffic from K3s' pod subnet (by default 10.42.0.0/16) and service subnet (by default 10.43.0.0/16) via **`ufw`** for K3s to work properly[1]. If **`configure-firewall`** (a script called by **`install-k3s`**) detects this situation, you will see a warning message and the following prompt[2]:

Allow traffic from 10.42.0.0/16 and 10.43.0.0/16 via ufw?

[1] For operating systems in the RHEL family (CentOS, Rocky Linux, AlmaLinux...), the action recommended by the K3s manual is to disable **`firewalld`** entirely (see https://rancher.com/docs/k3s/latest/en/advanced/#additional-preparation-for-red-hat-centos-enterprise-linux), but **`install-k3s`** takes a lighter approach commonly used in the K3s community.

References:

- https://github.com/k3s-io/k3s/issues/1556
  - https://github.com/k3s-io/k3s/issues/1556#issuecomment-604112415

[2] See *Installing K3s*.

[3] By default, K3s' pod subnet is 10.42.0.0/16 and service subnet is 10.43.0.0/16.

[4] See *Running Commands as a Privileged User*.

[1] References:

- https://github.com/k3s-io/k3s/issues/1280
  - https://github.com/k3s-io/k3s/issues/1280#issuecomment-663269728

[2] See *Installing K3s*.

Enter `Y` for "yes" and `N` for "no", or simply hit `Enter` to accept the proposed answer (displayed in square brackets).

Caution: If you opt out of the proposed remediation, K3s may malfunction.

The remediation attempted by **`configure-firewall`** is equivalent to[3]:

```
ufw allow from 10.42.0.0/16 to any

ufw allow from 10.43.0.0/16 to any

ufw reload
```

By default, K3s' pod subnet is 10.42.0.0/16 and service subnet is 10.43.0.0/16, but if you customized your K3s installation to use other subnets, you should substitute them here.

Tip: If your system did not initially use **`ufw`** at the time K3s was installed, but later does (for example because **`ufw`** becomes enabled), run this command in the `lockss` user's `lockss-installer` directory as a privileged user who can become root via **`sudo`**[4]:

```
scripts/configure-firewall
```

7.5 Troubleshooting CoreDNS

This section offers troubleshooting information related to DNS resolution in the K3s cluster.

If both `/etc/resolv.conf` and `/run/systemd/resolve/resolv.conf` (files used to list the IP address of DNS servers) contain loopback addresses, CoreDNS (a component of the K3s Kubernetes cluster that handles DNS resolution) will not work properly[1]. If **`configure-dns`** (a script called by **`install-k3s`**) detects this situation, you will see a warning message and the following prompt[2]:

IP address(es) of DNS resolvers, separated by ';'

Enter a semicolon-separated list of DNS server IP addresses that are *not* loopback addresses. A suggested default will be offered to you in square brackets, consisting of non-loopback addresses collected from your machine's `resolv.conf` files; you can simply hit `Enter` to accept the suggested default.

Tip: If the DNS settings of your system change after K3s is initially installed (for example if DNS servers are added or removed), run this command in the `lockss` user's `lockss-installer` directory as a privileged user who can become root via **`sudo`**[3]:

```
scripts/configure-dns
```

[3] By default, K3s' pod subnet is 10.42.0.0/16 and service subnet is 10.43.0.0/16.

[4] See *Running Commands as a Privileged User*.

[1] References:

- https://coredns.io/plugins/loop/#troubleshooting-loops-in-kubernetes-clusters

[2] See *Installing K3s*.

[3] See *Running Commands as a Privileged User*.

7.6 Troubleshooting K3s

This section offers troubleshooting information when the K3s installer or the K3s configuration checker fail.

7.6.1 Troubleshooting the K3s Installer

The LOCKSS Installer's **install-k3s** script installs K3s by executing Rancher's official K3s installer after making sure many firewall and DNS issues are resolved[1]. However, the installer can still run into issues and fail. Some of the error messages you might encounter are documented below, but you may need to refer to the official K3s documentation or use a search engine to look up the specific error message.

7.6.1.1 Failed to apply container_runtime_exec_t

In some Fedora systems, you may see an error message similar to the following:

```
[ERROR]  Failed to apply container_runtime_exec_t to /usr/local/bin/k3s, please install:
    yum install -y container-selinux selinux-policy-base
    yum install -y https://rpm.rancher.io/k3s/stable/common/centos/7/noarch/k3s-selinux-
↪0.2-1.el7_8.noarch.rpm
```

The specific commands and version numbers may vary from the example above.

To resolve this problem:

1. Run the recommended commands as `root`[6].
2. Re-run **install-k3s**[Page 41, 1].

7.6.1.2 k3s-selinux requires container-selinux

In some Oracle Linux 7 systems, you may see an error message similar to the following:

```
Error: Package: k3s-selinux-0.3-0.el7.noarch (rancher-k3s-common-stable)
           Requires: container-selinux >= 2.107-3
 You could try using --skip-broken to work around the problem
 You could try running: rpm -Va --nofiles --nodigest
```

The specific commands and version numbers may vary from the example above.

This can occur in environments where the Oracle Linux 7 Addons Yum repository is not enabled by default, so Rancher's official K3s installer is unable to install the package `container-selinux` automatically.

To resolve this problem:

1. Run the following command as `root`[6]:

   ```
   yum-config-manager --enable ol7_addons
   ```

2. Re-run **install-k3s**[1].

[1] See *install-k3s*.
[6] See *Running Commands as root*.

7.6.2 Troubleshooting the K3s Configuration Checker

After installing K3s with **install-k3s**[1] and successfully running **check-k3s**[2], you can run the following command as root[Page 41, 6]:

```
k3s check-config
```

This configuration checker runs through a more extensive series of tests, covering "required", "generally necessary", and "optional" aspects for K3s to operate.

As a rule of thumb, if **k3s check-config** ends successfully with `STATUS: pass`, there is a good chance the K3s cluster is configured correctly.

Some failures, especially in "optional" aspects, may not actually prevent the cluster from working normally in the limited ways the LOCKSS system uses Kubernetes, but if possible they should be addressed. Some of the error messages you might encounter are documented below, but you may need to refer to the official K3s documentation or use a search engine to look up the specific error message.

7.6.2.1 iptables should be older than v1.8.0 or in legacy mode

In some instances, you may encounter an error message similar to the following:

```
iptables v1.8.4 (nf_tables): should be older than v1.8.0 or in legacy mode (fail)
```

This error message is generally spurious, because the LOCKSS Installer should have previously detected and offered to correct this issue in the circumstances where it applies, and Rancher has a documented bug report that the K3s configuration checker keeps reporting this issue even in circumstances where it does not apply[4].

- If **check-k3s** ran successfully[2], your K3s cluster is probably running normally and you can ignore this error message even if you receive it.
- If your system is running **iptables** version 1.8.0 or later in `nf_tables` mode via Alternatives, as can be the case in some Debian or Ubuntu systems, **iptables** needs to be switched to `legacy` mode via Alternatives. The **configure-firewall** script called by **install-k3s** is supposed to detect this condition and offer to fix it for you[Page 41, 1]. See *Troubleshooting iptables*.

7.6.2.2 User namespaces disabled

In the RHEL/CentOS family of operating systems, you may receive the following error message:

```
RHEL7/CentOS7: User namespaces disabled; add 'user_namespace.enable=1' to boot command␣
↪line
```

To resolve this issue[5]:

1. Edit the file `/etc/default/grub` as root[Page 41, 6].
   1. Look for the line beginning with `GRUB_CMDLINE_LINUX=`, for example:

[2] See *check-k3s*.

[4] References:

- https://github.com/k3s-io/k3s/issues/2946

[5] References:

- https://fortuitousengineer.com/installing-kubernetes-k3s-on-centos-rhel-hosts/

```
GRUB_CMDLINE_LINUX="no_timer_check console=tty0 console=ttyS0,115200n8 net.
↪ifnames=0 biosdevname=0 elevator=noop crashkernel=auto"
```

2. Add `user_namespace.enable=1` to the space-separated list of boot arguments, for instance:

```
GRUB_CMDLINE_LINUX="user_namespace.enable=1 no_timer_check console=tty0␣
↪console=ttyS0,115200n8 net.ifnames=0 biosdevname=0 elevator=noop␣
↪crashkernel=auto"
```

2. Run the following command as `root`:

```
grub2-mkconfig -o /boot/grub2/grub.cfg
```

3. Reboot the system.
4. Re-run **`k3s check-config`**[3].

7.6.2.3 swap should be disabled

This warning can be ignored:

```
swap: should be disabled
```

7.6.2.4 CONFIG_INET_XFRM_MODE_TRANSPORT missing

This warning can be ignored:

```
CONFIG_INET_XFRM_MODE_TRANSPORT: missing
```

---

[3] See *k3s-check-config*.

CHAPTER EIGHT

APPENDIX

This appendix contains pages of additional information about the LOCKSS system.

8.1 Security Advisories

Topic Overview

8.1.1 CVE-2021-45105 and CVE-2021-44832

First published: 2021-01-02

Attention: The LOCKSS 2.x system up to and including 2.0.51-alpha5 (originally released 2021-12-17), and the custom Solr and OpenWayback containers it includes, are affected by CVE-2021-45105 and CVE-2021-44832.

The recommended remediation is to upgrade LOCKSS 2.0.51-alpha5 and earlier to LOCKSS 2.0.52-alpha5 or later.

See CVE-2021-45105 and CVE-2021-44832 in our Security pages.

8.1.2 CVE-2021-44228, CVE-2021-45046 and CVE-2021-4104

First published: 2021-12-13
Last updated: 2021-01-02

Attention: The LOCKSS 2.x system up to and including version 2.0.42-alpha4, and the custom Solr and OpenWayback containers it includes, are affected by CVE-2021-44228 ("Log4Shell"), CVE-2021-45046 and CVE-2021-4104.

Because additional vulnerabilities in Log4j 2.x have been discovered, the recommended remediation is to upgrade to LOCKSS version 2.0.42-alpha4 and earlier to LOCKSS 2.0.52-alpha5 immediately.

See CVE-2021-44228, CVE-2021-45046 and CVE-2021-4104 in our Security pages.

8.2 Release Notes

8.2.1 LOCKSS 2.0.43-alpha4

Released: 2021-12-15
Also known as: LOCKSS 2.0-alpha4c

LOCKSS 2.0.43-alpha4 (also known as LOCKSS 2.0-alpha4c) is a security release and the latest release of the LOCKSS 2.0-alpha4 system. It addresses security vulnerabilities in Apache Log4j 2.x. See *CVE-2021-45105 and CVE-2021-44832* in our *Security Advisories*.

Release Notes

- Custom versions of embedded Solr and OpenWayback with Log4j 2.16.0 (CVE-2021-45105).

Component Versions

LOCKSS 2.0.43-alpha4 consists of a configurable set of the following components:

- LOCKSS Installer version 2.0.43-alpha4
- LOCKSS Repository Service version 2.11.0
- LOCKSS Configuration Service version 2.5.0
- LOCKSS Poller Service version 2.3.0
- LOCKSS Metadata Extraction Service version 2.4.0
- LOCKSS Metadata Service version 2.3.0
- PostgreSQL version 9.6.12
- Apache Solr version 7.2.1
- Pywb version 2.4.2
- OpenWayback version 2.4.0 (custom version 2.4.0-3)

8.2.2 LOCKSS 2.0.42-alpha4

Released: 2021-12-13
Also known as: LOCKSS 2.0-alpha4b

LOCKSS 2.0.42-alpha4 (also known as LOCKSS 2.0-alpha4b) is a security release of the LOCKSS 2.0-alpha4 system. It addresses security vulnerabilities in Apache Log4j 2.x. See *CVE-2021-44228, CVE-2021-45046 and CVE-2021-4104* in our *Security Advisories*.

Release Notes

- Run LOCKSS services, Solr and OpenWayback with Log4j2 mitigation (CVE-2021-44228).

Component Versions

LOCKSS 2.0.42-alpha4 consists of a configurable set of the following components:

- LOCKSS Installer version 2.0.42-alpha4
- LOCKSS Repository Service version 2.11.0
- LOCKSS Configuration Service version 2.5.0
- LOCKSS Poller Service version 2.3.0
- LOCKSS Metadata Extraction Service version 2.4.0
- LOCKSS Metadata Service version 2.3.0
- PostgreSQL version 9.6.12
- Apache Solr version 7.2.1
- Pywb version 2.4.2
- OpenWayback version 2.4.0

8.2.3 LOCKSS 2.0.41-alpha4

Released: 2021-06-28
Also known as: LOCKSS 2.0-alpha4a

LOCKSS 2.0.41-alpha4 (also known as LOCKSS 2.0-alpha4a) is the first release of the LOCKSS 2.0-alpha4 system.

Release Notes

- Containers are now orchestrated by **K3s**, a lightweight Kubernetes distribution by Kubernetes vendor Rancher. K3s is compatible with most Linux flavors and comes with a convenient multi-OS installer.
- Content is now stored in compressed WARC files by default.
- Numerous bug fixes and enhancements in the repository service and underlying componentry, delivering more reliability and performance in error handling, scalability, and content replay.
- Many security enhancements, including support for **`firewalld`** and **`ufw`**, containers not running as `root` internally, the `lockss` user no longer needing **`sudo`** privileges, Solr authentication, and LCAP SSL support with TLSv1.2 by default.
- Web user interface enhancements, including LCAP SSL key generation and support for the Kubernetes dashboard.

Component Versions

LOCKSS 2.0.41-alpha4 consists of a configurable set of the following components:

- LOCKSS Installer version 2.0.41-alpha4
- LOCKSS Repository Service version 2.11.0
- LOCKSS Configuration Service version 2.5.0
- LOCKSS Poller Service version 2.3.0
- LOCKSS Metadata Extraction Service version 2.4.0
- LOCKSS Metadata Service version 2.3.0
- PostgreSQL version 9.6.12
- Apache Solr version 7.2.1
- Pywb version 2.4.2
- OpenWayback version 2.4.0

8.3 Past Versions

- 2.0-alpha3 (2020-10-29)
- 2.0-alpha2 (2020-02-06)
- 2.0-alpha1 (2019-05-13)
- 2.0-alpha Technology Preview (2019-04-05)

8.4 Running Commands as `root`

Some commands or scripts in this manual are intended to be run as `root`. This section describes two methods for doing so.

Topic Overview

- *Running Commands as `root` With **sudo***
- *Running Commands Directly as `root`*

8.4.1 Running Commands as `root` With sudo

If you are logged in as a user who can run commands as `root` via **sudo**, simply add the following in front of the command listed in the manual[1]:

```
sudo ...
```

For example, if the command listed in the manual is `iptables -F`, you would type `sudo iptables -F`.

8.4.2 Running Commands Directly as `root`

If you are logged in as `root` directly, you can simply run the command as listed in the manual, for example `iptables -F`.

See Also

- *Running Commands as a Privileged User*
- *Running Commands as the lockss User*

8.5 Running Commands as a Privileged User

Some commands or scripts in this manual are intended to be run as a privileged user who can become `root` via **sudo**. Compared to running an entire command or script directly as `root` (see *Running Commands as root*), this approach has the security advantage of being granular -- only those portions of the command or script that require `root` privileges will have `root` privileges.

To run a command in this context, simply type the command listed in the manual. Depending on your system's **sudo** configuration, you may be prompted for the user's **sudo** password.

---

[1] Depending on your system's **sudo** configuration, you may be prompted for the user's **sudo** password.

See Also

- *Running Commands as root*
- *Running Commands as the lockss User*

8.6 Running Commands as the `lockss` User

Unless otherwise noted, most commands in this manual are intended to be run as the `lockss` user (oftentimes in the `lockss` user's `lockss-installer` directory). This section describes two methods for doing so.

Topic Overview

- *Running Commands as `lockss` With **`sudo`***
- *Running Commands as `lockss` With **`su`***

8.6.1 Running Commands as `lockss` With `sudo`

If you are logged in as a user who can run commands as `lockss` via **`sudo`**:

- You can start a Bash shell session as the `lockss` user and run any number of commands in it:
  1. Run this command[1]:

     ```
     sudo -i -u lockss
     ```

     Tip: You can also use the slightly shorter version `sudo -iu lockss`.

  2. Run commands as they are listed in the manual, for example `scripts/start-lockss --wait`.
  3. When you are done, exit the `lockss` shell session by typing `exit` or `logout` or hitting `Ctrl + D`.
- Alternatively, you can use **`sudo`** to run a single command as the `lockss` user.

  Add the following in front of the command listed in the manual[Page 50, 1]:

  ```
  sudo -u lockss ...
  ```

  For example, if the command listed in the manual is `scripts/start-lockss --wait`, you would type `sudo -u lockss scripts/start-lockss --wait`.

[1] Depending on your system's **`sudo`** configuration, you may be prompted for the user's **`sudo`** password.

8.6.2 Running Commands as `lockss` With `su`

If you are logged in as `root` but your system does not have **sudo** (or does not let `root` use **sudo**), you can use **su** instead:

- You can use **su** to start a Bash shell session as the `lockss` user and run any number of commands in it:
  1. Type this command:

     ```
     su lockss
     ```

  2. Run commands as they are listed in the manual, for example `scripts/start-lockss --wait`.
  3. When you are done, exit the `lockss` shell session by typing `exit` or `logout` or hitting `Ctrl + D`.
- Alternatively, you can use **su** to run a single command as the `lockss` user:

  Put the command listed in the manual in quotation marks in the following way:

  ```
  su -c '...' lockss
  ```

  For example, if the command to be run as the `lockss` user is `scripts/start-lockss --wait`, you would type `su -c 'scripts/start-lockss --wait' lockss`.

  Caution: You will need to take care if the command itself contains quotation marks[2] .

See Also

- *Running Commands as root*
- *Running Commands as a Privileged User*

8.7 Working with PostgreSQL

This section of the appendix documents administrative tasks for the embedded PostgreSQL database configured by the LOCKSS 2.x system.

8.7.1 Changing the PostgreSQL Database Password

To change the password of the embedded PostgreSQL database, perform the following steps as the `lockss` user[1] in the `lockss` user's `lockss-installer` directory:

1. Ensure the Kubernetes service definitions reflect the current state of the LOCKSS configuration by running:

   ```
   scripts/assemble-lockss
   ```

[2] If the command contains quotation marks, use `-c "..."` instead of `-c '...'`, and add a backslash in front of each double quotation mark in the command (`\"` instead of `"`); single quotation marks in the command are unchanged.

[1] See *Running Commands as the lockss User*.

2. Start the PostgreSQL database container by running:

   ```
   k3s kubectl apply -n lockss --filename=config/configs/lockss-stack/svcs/lockss-
   ↪postgres-service.yaml
   ```

3. Run the following command to store the name of the PostgreSQL database container into the variable `postgres_pod`:

   ```
   postgres_pod=$(k3s kubectl get pod -n lockss --selector=io.kompose.service=lockss-
   ↪postgres-service --output=jsonpath="{.items[0].metadata.name}")
   ```

4. Run the following command to store the IP of the PostgreSQL database container into the variable `postgres_ip`:

   ```
   postgres_ip=$(k3s kubectl get pod -n lockss --selector=io.kompose.service=lockss-
   ↪postgres-service --output=jsonpath="{.items[0].status.podIP}")
   ```

5. Execute the following command to alter the LOCKSS database user's password, taking care to replace *`newpassword`* with your new embedded PostgreSQL database password:

   ```
   echo "ALTER USER \"LOCKSS\" WITH PASSWORD 'newpassword'" | k3s kubectl exec
   ↪$postgres_pod -n lockss -i -- psql --username=LOCKSS --dbname=postgres
   ```

   Successful execution of the command results in the output `ALTER ROLE`.

6. To verify that the password change worked, run the following command:

   ```
   k3s kubectl exec $postgres_pod -n lockss -it -- psql --username=LOCKSS --
   ↪dbname=postgres --host=$postgres_ip
   ```

   and enter *`newpassword`* at the *Password for user LOCKSS* prompt. If the password change was successful and you enter *`newpassword`* correctly, you will see a PostgreSQL prompt similar to:

   ```
   psql (9.6.12)
   Type "help" for help.

   postgres=#
   ```

   which you can exit by entering `q` or hitting `Ctrl + D`. If the password change was unsuccessful or you do not enter *`newpassword`* correctly, you will see output similar to:

   ```
   psql: FATAL:  password authentication failed for user "LOCKSS"
   command terminated with exit code 2
   ```

7. Stop the PostgreSQL database container by running this command:

   ```
   k3s kubectl -n lockss delete service,deployment lockss-postgres-service &&
       k3s kubectl -n lockss wait --for=delete pod $postgres_pod --timeout=60s
   ```

8. Re-run **`configure-lockss`** so that you can record the new embedded PostgreSQL database password into the configuration of the LOCKSS stack:

   ```
   scripts/configure-lockss
   ```

   See the *PostgreSQL* and *Embedded PostgreSQL Database* sections of *Configuring the LOCKSS System* for details.

8.8 LCAP Over SSL

The section explains how to configure secure communication between LOCKSS boxes in a network.

Some LOCKSS networks, such as the Global LOCKSS Network (GLN), are open, in the sense that anyone may join and set up a LOCKSS box to participate in that network. The LOCKSS polling protocol (LCAP) includes several security measures to prevent rogue players from disrupting the network, but it is also possible to create a closed network, where only authorized nodes are allowed to participate. This document describes the steps needed to set up such a network.

In order to ensure that only authorized nodes may participate, each node is issued a private key, and all nodes are provided the set of corresponding public keys. This allows all inter-node communication to be both encrypted and authenticated, using SSL.

Note: The Classic LOCKSS system (version 1.x) does not support PKCS12, so if building keystores for a network that includes classic LOCKSS nodes, JCEKS should be selected.

8.8.1 Generating Keystores

The authority in charge of the private LOCKSS network (PLN) must create and distribute Java keystores to all participants. Each box receives two keystores: one containing its own private key (along with a password file containing the secret password for the private key) and another containing the public certificates for each of the boxes in the network. There are two methods available to create these keystores:

- A *Command Line Tool* run in the LOCKSS development environment.
- An *Interactive Tool* invoked in a running LOCKSS node.

In both cases, the admin creating the keystores must know the complete set of hostnames of boxes in the network. More hosts can be added at any time, but a new public keystore must be created and distributed to each box.

8.8.1.1 Command Line Tool

To use the command line tool:

1. Clone the lockss-core and laaws-dev-scripts projects from GitHub, in sibling directories.
2. Build `lockss-core`.
3. In the root directory of `lockss-core`, run this command:

   ```
   ../laaws-dev-scripts/bin/runclass org.lockss.keystore.EditKeyStores -s pubkeystore.
   ↪pkcs12 -o keydir box1.pln.org ... boxN.pln.org
   ```

   This will create, in the directory ***`keydir`***, a public keystore named `pubkeystore.pkcs12`, and a pair of files `boxK.pln.org.pkcs12` and `boxK.pln.org.pass` for each one of the *N* host names `box1.pln.org` through `boxN.pln.org`.

4. To add additional hosts, provide the existing public keystore as the value of the `-s` argument, and list the new hosts. The new public keys will be added to the existing public keystore.

8.8.1.2 Interactive Tool

1. Bring up a LOCKSS stack, either in the production environment or `runcluster`. In the UI, select *DebugPanel → Generate LCAP Keys*.
2. Enter the hostname of each of the LOCKSS boxes in the *Hostnames* text box, then click the *Generate Keystores* button. A .tgz or a .zip file will be generated and offered for download. This file will contain the private keystore and password file for each host, as well as the shared public keystore.
3. To add additional hosts, use the *Browse* button to supply the existing public keystore, and enter the new hosts in the *Hostnames* text box. The downloaded file will contain the private keystore and password files for each new host, as well as the updated shared public keystore, which must be installed on all hosts.

8.8.2 Installing the Keystores

1. **Securely** transmit to each box its two files and the public keystore. Put them in `~lockss/lockss-installer/config/keys`, and set the owner and group to `lockss:lockss` and the permissions to `600`.
2. Restart the stack and check that it is now using SSL. In the UI, select *Daemon Status → Comm Channels*. The page should show *SSL: TLSv1.2, Client Auth*.
3. After a few hours, select *Daemon Status → Comm Peer Data* to ensure that each box is successfully originating and accepting connections from all the other boxes.

8.9 Network Ports

This section describes the default network ports used by the LOCKSS system.

Unless otherwise noted, all ports are **TCP**.

All ports in the 24600-24699 range should be considered reserved. The LCAP (LOCKSS polling and repair) port retains its historical value of 9729.

- 8080: OpenWayback replay engine[1]
- 9729: LCAP (LOCKSS polling and repair)
- 24600: *reserved*
- 24602: PostgreSQL
- 24603: Solr
- 24606: ActiveMQ
- 24610: LOCKSS Repository Service - REST port
- 24619: *reserved* (HDFS FS port)
- 24620: LOCKSS Configuration Service - REST port
- 24621: LOCKSS Configuration Service - UI port
- 24630: LOCKSS Poller Service - REST port
- 24631: LOCKSS Poller Service - UI port
- 24640: LOCKSS Metadata Extraction Service - REST port
- 24641: LOCKSS Metadata Extraction Service - UI port

[1] This is a known issue. In a future version of the system, the intended port will be 24682.

- 24650: LOCKSS Metadata Service - REST port
- 24651: LOCKSS Metadata Service - UI port
- 24670: LOCKSS Proxy
- 24671: *reserved*
- 24672: LOCKSS Audit Proxy
- 24673: *reserved*
- 24674: ICP server **(UDP)**
- 24680: LOCKSS Content Server (ServeContent)
- 24681: Pywb replay engine
- 24682: *reserved*[Page 54, 1]