LOCKSS System Manual

LOCKSS Program

2023-09-28

LOCKSS 2.0-ALPHA6 SYSTEM MANUAL

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-alpha6, is the sixth 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 **16 GB** of memory, preferably more.

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-alpha5 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. The K3s documentation states[1] that "K3s is expected to work on most modern Linux systems", and that "Some OSs have specific requirements" (which are documented in this manual and integrated into the **`lockss-installer`** scripts).

The LOCKSS team has successfully tested the LOCKSS system installation process on many flavors of Linux, some of which are listed below:

AlmaLinux OS

The LOCKSS system is compatible with AlmaLinux OS:

[1] Reference: https://docs.k3s.io/installation/requirements#operating-systems

Operating System	Version	Active Support	Security Support	Notes
AlmaLinux OS	9.1	2027-05-31	2032-05-31	
AlmaLinux OS	9.0	2027-05-31	2032-05-31	
AlmaLinux OS	8.7	2024-05-31	2029-05-31	
AlmaLinux OS	8.6	2024-05-31	2029-05-31	
AlmaLinux OS	8.5	2024-05-31	2029-05-31	
AlmaLinux OS	8.4	2024-05-31	2029-05-31	
AlmaLinux OS	8.3	2024-05-31	2029-05-31	

Arch Linux

The LOCKSS system is compatible with Arch Linux:

Operating System	Version	Active Support	Security Support	Notes
Arch Linux				rolling updates

CentOS

Caution: We no longer recommend CentOS for new installations; we recommend Rocky Linux instead.

The LOCKSS system is compatible with CentOS:

Operating System	Version	Active Support	Security Support	Notes
CentOS Stream				rolling updates
CentOS Linux	7.9	2020-08-06	2024-06-30	end of life
CentOS Linux	7.8	2020-08-06	2024-06-30	end of life
CentOS Linux	7.7	2020-08-06	2024-06-30	end of life
CentOS Linux	7.6	2020-08-06	2024-06-30	end of life
CentOS Linux	7.5	2020-08-06	2024-06-30	end of life
CentOS Linux	7.4	2020-08-06	2024-06-30	end of life
CentOS Linux	7.3	2020-08-06	2024-06-30	end of life

Debian

The LOCKSS system is compatible with Debian:

Operating System	Version	Active Support	Security Support	Notes
Debian	11.6	2024-07-01	2026-06-30	
Debian	11.5	2024-07-01	2026-06-30	
Debian	11.4	2024-07-01	2026-06-30	
Debian	11.3	2024-07-01	2026-06-30	
Debian	11.2	2024-07-01	2026-06-30	
Debian	11.1	2024-07-01	2026-06-30	
Debian	11.0	2024-07-01	2026-06-30	
Debian	10.13	2022-09-10	2024-06-30	
Debian	10.12	2022-09-10	2024-06-30	
Debian	10.11	2022-09-10	2024-06-30	
Debian	10.10	2022-09-10	2024-06-30	
Debian	10.9	2022-09-10	2024-06-30	
Debian	10.8	2022-09-10	2024-06-30	
Debian	10.7	2022-09-10	2024-06-30	
Debian	10.6	2022-09-10	2024-06-30	
Debian	10.5	2022-09-10	2024-06-30	
Debian	10.4	2022-09-10	2024-06-30	
Debian	10.3	2022-09-10	2024-06-30	
Debian	10.2	2022-09-10	2024-06-30	
Debian	10.1	2022-09-10	2024-06-30	
Debian	10.0	2022-09-10	2024-06-30	

EuroLinux

The LOCKSS system is compatible with EuroLinux:

Operating System	Version	Active Support	Security Support	Notes
EuroLinux	9.1	2032-05-31	2032-06-30	
EuroLinux	9.0	2032-05-31	2032-06-30	
EuroLinux	8.6	2029-03-01	2029-06-30	
EuroLinux	8.5	2029-03-01	2029-06-30	
EuroLinux	8.4	2029-03-01	2029-06-30	
EuroLinux	8.3	2029-03-01	2029-06-30	
EuroLinux	7.9	2024-07-31	2024-07-31	
EuroLinux	7.8	2024-07-31	2024-07-31	
EuroLinux	7.7	2024-07-31	2024-07-31	
EuroLinux	7.6	2024-07-31	2024-07-31	

Fedora Linux

The LOCKSS system is compatible with Fedora Linux:

Operating System	Version	Active Support	Security Support	Notes
Fedora Linux	37	2023-12-15	2023-12-15	
Fedora Linux	36	2023-05-16	2023-05-16	

Linux Mint

The LOCKSS system is compatible with Linux Mint:

Operating System	Version	Active Support	Security Support	Notes
Linux Mint	21	Yes	2027-04-01	
Linux Mint	20.3	Yes	2025-04-01	
Linux Mint	20.2	Yes	2025-04-01	
Linux Mint	20.1	No	2025-04-01	end of life
Linux Mint	20	No	2025-04-01	end of life

OpenSUSE

The LOCKSS system is compatible with OpenSUSE:

Operating System	Version	Active Support	Security Support	Notes
OpenSUSE Tumbleweed				rolling updates
OpenSUSE Leap	15.4	2023-12-01	2023-12-01	

Oracle Linux

The LOCKSS system is compatible with Oracle Linux:

Operating System	Version	Active Support	Security Support	Notes
Oracle Linux	9.1	2032-07-01	2034-06-01	
Oracle Linux	9.0	2032-07-01	2034-06-01	
Oracle Linux	8.7	2029-07-01	2029-07-01	
Oracle Linux	8.6	2029-07-01	2029-07-01	
Oracle Linux	8.5	2029-07-01	2029-07-01	
Oracle Linux	8.4	2029-07-01	2029-07-01	
Oracle Linux	8.3	2029-07-01	2029-07-01	
Oracle Linux	8.2	2029-07-01	2029-07-01	
Oracle Linux	8.1	2029-07-01	2029-07-01	
Oracle Linux	7.9	2024-07-01	2026-06-01	
Oracle Linux	7.8	2024-07-01	2026-06-01	
Oracle Linux	7.7	2024-07-01	2026-06-01	
Oracle Linux	7.6	2024-07-01	2026-06-01	

RHEL

The LOCKSS system is compatible with RHEL:

Operating System	Version	Active Support	Security Support	Notes
RHEL	8.3	2024-05-31	2029-05-31	

Rocky Linux

Tip: Rocky Linux is the operating system we currently recommend for new installations, and for existing installations based on CentOS or Scientific Linux.

The LOCKSS system is compatible with Rocky Linux:

Operating System	Version	Active Support	Security Support	Notes
Rocky Linux	9.1	2025-05-31	2032-05-31	
Rocky Linux	9.0	2025-05-31	2032-05-31	
Rocky Linux	8.7	2024-05-31	2029-05-31	
Rocky Linux	8.6	2024-05-31	2029-05-31	
Rocky Linux	8.5	2024-05-31	2029-05-31	
Rocky Linux	8.4	2024-05-31	2029-05-31	

Scientific Linux

Caution: We no longer recommend Scientific Linux for new installations; we recommend Rocky Linux instead.

The LOCKSS system is compatible with Scientific Linux:

Operating System	Version	Active Support	Security Support	Notes
Scientific Linux	7.9	2024-06-30	2024-06-30	obsolescent
Scientific Linux	7.8	2024-06-30	2024-06-30	obsolescent
Scientific Linux	7.7	2024-06-30	2024-06-30	obsolescent
Scientific Linux	7.6	2024-06-30	2024-06-30	obsolescent

Ubuntu

The LOCKSS system is compatible with Ubuntu:

Operating System	Version	Active Support	Security Support	Notes
Ubuntu	22.10	2023-07-20	2023-07-20	
Ubuntu	22.04 LTS	2027-04-21	2032-04-01	
Ubuntu	20.04 LTS	2025-04-02	2030-04-01	
Ubuntu	18.04 LTS	2023-04-02	2028-04-01	

The LOCKSS system can likely be installed successfully on slightly different versions of the Linux flavors above, as well as other Linux flavors altogether, including commercial variants like RHEL or SLES. We welcome reports of successful installations from the community so they can be added to the list above.

Tip: Rocky Linux is the operating system we currently recommend for new installations, and for existing installations based on CentOS or Scientific Linux.

CHAPTER

TWO

UPGRADING FROM LOCKSS 2.0-ALPHA5

Note: This chapter describes how to upgrade an existing LOCKSS 2.0-alpha5 system to 2.0-alpha6. If you are installing the LOCKSS 2.x system for the first time, please see the installation instructions in the next chapter:

Installing the LOCKSS System

Tip: Before you begin the upgrade, we strongly recommend you first bring your operating system up to date by applying security updates and upgrading installed packages. Ask your system administrator or see *Operating System Updates* in the appendix.

2.1 Stop LOCKSS 2.0-alpha5

The first step is to stop the LOCKSS 2.0-alpha5 system. Log in as the `lockss` user and run the following command in the *LOCKSS Installer Directory* (by default *$HOME*`/lockss-installer`, typically `/home/lockss/lockss-installer`).

```
scripts/stop-lockss
```

You may verify all LOCKSS components have stopped by running the following command:

```
k3s kubectl get deployments -n lockss
```

which should return:

```
No resources found in lockss namespace.
```

2.2 Update the LOCKSS Installer

As of 2.0-alpha5, the official way to install and upgrade the LOCKSS Installer is using the LOCKSS Downloader, rather than cloning the LOCKSS Installer as a Git project. The instructions below detail the use of the LOCKSS Downloader. (Advanced users may continue to use **`git`** if they wish; please see *Downloading the LOCKSS Installer using git* for instructions.)

As the `lockss` user, run either this **`curl`** command:

```
curl -sSfL https://lockss.org/downloader | sh -s -
```

Or this **wget** command:

```
wget -qO- https://lockss.org/downloader | sh -s -
```

This will download and invoke the LOCKSS Downloader, which in turn will install the latest version of the LOCKSS Installer into the default LOCKSS Installer Directory (`$HOME/lockss-installer`). If you are using a custom LOCKSS Installer Directory *DIR*, remember to use `--download-dir=DIR`; see *Running the LOCKSS Downloader* for details.

2.3 Run the Upgrade Script

The next step is to update archived content from the previous release version. As the `lockss` user, run the following command in the *LOCKSS Installer Directory*:

```
scripts/upgrades/upgrade-to-alpha6
```

Hint: If it takes more than a few seconds for `upgrade-to-alpha6` above to run, the reindexing of all previously archived content which occurs the first time you start 2.0-alpha6 after upgrading from 2.0-alpha5 may take prohibitively long. This performance issue will be addressed in the next release. If you do not need the previously stored content during alpha testing, you could delete it and skip this reindexing step; see *Resetting the System to a Blank State*.

2.4 Re-run the Configure Script

Re-run the configuration script by running the command below and follow the instructions in *Configuring the LOCKSS System* to ensure all existing configuration parameters are still correct and to configure any new parameters:

```
scripts/configure-lockss
```

2.5 Start LOCKSS 2.0-alpha6

Follow the instructions in *Running the LOCKSS System* to start your LOCKSS 2.0-alpha6 instance:

```
scripts/start-lockss
```

Hint: If it takes more than a few seconds for `upgrade-to-alpha6` above to run, the reindexing of all previously archived content which occurs the first time you start 2.0-alpha6 after upgrading from 2.0-alpha5 may take prohibitively long. This performance issue will be addressed in the next release. If you do not need the previously stored content during alpha testing, you could delete it and skip this reindexing step; see *Resetting the System to a Blank State*.

CHAPTER

THREE

INSTALLING THE LOCKSS SYSTEM

Note: This chapter describes how to install the LOCKSS 2.0-alpha6 system from scratch. If you are upgrading an existing LOCKSS 2.0-alpha5 system to 2.0-alpha6, please see the upgrade instructions in the previous chapter:

Upgrading From LOCKSS 2.0-alpha5

Tip: Before you begin installing the LOCKSS system, we strongly recommend you first bring your operating system up to date by applying security updates and upgrading installed packages. Ask your system administrator or see *Operating System Updates*.

3.1 Creating the `lockss` User

Note: Commands in this section are run as `root`[1].

The first task is to create a system user named `lockss`, under which the LOCKSS system will run.

Run this **`useradd`** command as `root`[1] :

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

or equivalently:

```
useradd -rUms /bin/bash lockss
```

This will create a `lockss` system user, a `lockss` system group, and a home directory in `/home/lockss`.

[1] See *Running Commands as root*.

3.2 Downloading the LOCKSS Installer

Note: Commands in this section are run as the `lockss` user[1].

The next task is to download the LOCKSS Installer.

3.2.1 LOCKSS Installer Directory

The LOCKSS Installer will be installed into a directory that will simply be known as the **LOCKSS Installer Directory**. Many commands in upcoming sections of this manual, such as those to install, configure, start and stop the LOCKSS system, will be listed relative to the LOCKSS Installer Directory.

By default, the LOCKSS Installer Directory is a directory named `lockss-installer` in the `lockss` user's home directory.

3.2.2 Running the LOCKSS Downloader

To install the LOCKSS Installer, you will use **`curl`** or **`wget`**[2] to invoke the LOCKSS Downloader[3]. (Alternatively, for security purposes, you can download and inspect the LOCKSS Downloader before executing it manually[5].)

As the `lockss` user[Page 12, 1], run either this **`curl`** command:

```
curl -sSfL https://lockss.org/downloader | sh -s -
```

or this **`wget`** command:

```
wget -qO- https://lockss.org/downloader | sh -s -
```

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

[2] Most typical Linux systems have at least one of **`curl`** or **`wget`** installed by default. You can check by typing `curl --version` or `wget --version` and verifying that the output is not an error message. If you need to install **`curl`**, see *Installing curl*. If you prefer to install **`wget`**, see *Installing wget*.

[3] See https://github.com/lockss/lockss-downloader.

[5] For security purposes, you may wish to inspect the LOCKSS Downloader before executing it.

One option is to review the contents of the script directly on GitHub to your satisfaction, then execute it as described above. The URL https://lockss.org/downloader redirects to https://github.com/lockss/lockss-downloader/raw/main/lockss-downloader.

Another option is to download a copy of the LOCKSS Downloader, review the **`lockss-downloader`** script, then execute it, all locally. To do so, follow this procedure:

1. Run either:

   ```
   curl -Lo /tmp/lockss-downloader https://lockss.org/downloader
   ```

   or:

   ```
   wget -O /tmp/lockss-downloader https://lockss.org/downloader
   ```

   to download the **`lockss-downloader`** script to `/tmp/lockss-downloader`.

2. Inspect `/tmp/lockss-downloader` to your satisfaction.

3. Run this command:

   ```
   chmod +x /tmp/lockss-downloader
   ```

   to make `/tmp/lockss-downloader` executable.

4. Type:

   ```
   /tmp/lockss-downloader
   ```

   to run the **`lockss-downloader`** script, appending options like `--download-dir=`*`DIR`* to the end as desired.

You can add options after `| sh -s -`:

- If you need your LOCKSS Installer Directory to be a directory *DIR* other than *the default*, add `--download-dir=DIR` (or alternatively `-d DIR`) after `| sh -s -`, like so:

  ```
  ... | sh -s - --download-dir=DIR
  ```

  and the LOCKSS Installer will be installed into the custom LOCKSS Installer Directory *DIR*.

- (Advanced uses only.) If you have a reason to install a version of the LOCKSS Installer other than the latest stable release, you can do so by making references to the `lockss-installer` Git repository on GitHub[4]:
  - You can install a version labeled by the Git tag *TAG* (e.g. `version-2.0.61-alpha6`) by adding `--git-tag=TAG` (or `-t TAG`).
  - You can install a version from the tip of a Git branch *BRA* (e.g. `develop`) by adding `--git-branch=BRA` (or `-b BRA`).
  - You can even install a version as of a specific Git commit identifier *COM* by adding `--git-commit=COM` (or `-c COM`).
- The LOCKSS Downloader accepts other options after `| sh -s -`; you can list them by adding `--help` (or `-h`) after `| sh -s -`.

3.3 Running the LOCKSS Installer

Note: Commands in this section are run as `root`[1].

The next task is to run the LOCKSS Installer.

3.3.1 Overview of the LOCKSS Installer

When you invoke the LOCKSS Installer, the installation process goes through various phases:

- Checking that some prerequisites to install K3s are met. No user interaction is expected.
- Checking that the `lockss` system user and group exist. No user interaction is expected.
- Configuring **`iptables`**, **`firewalld`** and **`ufw`** for K3s. If applicable, you will be prompted to confirm before your system configuration is modified. You may incidentally be prompted for your **`sudo`** password.
- Configuring CoreDNS for K3s. If applicable, you will be prompted to enter non-loopback IP addresses of DNS servers.
- Installing K3s. If applicable, you will be prompted for a Kubernetes state data storage directory.
- Testing the K3s node. No user interaction is expected.

After the LOCKSS Installer succeeds, you can also optionally run the K3s Configuration Checker.

[4] See https://github.com/lockss/lockss-installer.
[1] See *Running Commands as root*.

3.3.2 Invoking the LOCKSS Installer

To start the installation process, run this command (relative to the lockss-installer-directory) as `root`[Page 13, 1]:

```
scripts/install-lockss
```

The installer will run through its phases, each of which is described in its own section below. The first phase is *Checking K3s Prerequisites* (Section 3.3.3).

3.3.3 Checking K3s Prerequisites

Heading

This phase begins with the heading *Checking K3s prerequisites...*.

Description

During this phase, **install-lockss** will check that certain prerequisites to installing K3s are met.

Steps

1. If **install-lockss** was invoked with the `--skip-check-prerequisites` option (implied by `--skip-install-k3s`), you will see one of these messages:

   ```
   [success] Skipping (--skip-install-k3s)

   [success] Skipping (--skip-check-prerequisites)
   ```

   and **install-lockss** will successfully proceed to the next phase, *Checking the System User and Group* (Section 3.3.4).

2. Next, **install-lockss** will check that user namespaces are enabled. In some RHEL 7 and CentOS 7 systems, user namespaces are not enabled by default; if this is the case, you will see the error message:

   ```
   [ERROR] User namespaces must be enabled in RHEL/CentOS 7; see manual
   ```

   and **install-lockss** will fail.

   Troubleshooting

   See *Enabling User Namespaces in RHEL 7 and CentOS 7* for troubleshooting, then go back to *Invoking the LOCKSS Installer* to try again.

3. Then **install-lockss** will check that **apparmor_parser** is installed if Apparmor is enabled. If Apparmor is enabled but **apparmor_parser** is not installed, you will see the error message:

   ```
   [ERROR] apparmor enabled but apparmor_parser missing; see manual
   ```

   and **install-lockss** will fail.

   Troubleshooting

See *Installing apparmor_parser* for troubleshooting, then go back to *Invoking the LOCKSS Installer* to try again.

4. Finally, you will see the message:

   ```
   [success] K3s prerequisites checked
   ```

   and **install-lockss** will successfully proceed to the next phase, *Checking the System User and Group* (Section 3.3.4).

3.3.4 Checking the System User and Group

Heading

This phase begins with the heading *Checking the system user and group...*.

Description

During this phase, **install-lockss** will check that the `lockss` user and group exist on the host system.

Steps

1. If **install-lockss** was invoked with the `--skip-check-system-user` option, you will see the message:

   ```
   [success] Skipping (--skip-check-system-user)
   ```

   and **install-lockss** will successfully proceed to the next phase, *Configuring iptables for K3s* (Section 3.3.5).

2. If the `lockss` user or group does not exist on the host system, you will see one of these error messages:

   ```
   [ERROR] The lockss user does not exist

   [ERROR] The lockss group does not exist
   ```

   and **install-lockss** will fail.

   Troubleshooting

   See the *Creating the lockss User* section to create the `lockss` user and group, then go back to *Invoking the LOCKSS Installer* to try again.

3. Finally, you will see the message:

   ```
   [success] System user and group present
   ```

   and **install-lockss** will successfully proceed to the next phase, *Configuring iptables for K3s* (Section 3.3.5).

3.3.5 Configuring `iptables` for K3s

Heading

This phase begins with the heading *Configuring iptables for K3s…*.

Description

During this phase, **install-lockss** will configure **iptables** to work with K3s, if applicable.

Steps

1. If **install-lockss** was invoked with the `--skip-configure-iptables` option (implied by `--skip-install-k3s`), or if no changes to the configuration of **iptables** are necessary, you will see one of these messages:

   ```
   [success] Skipping (--skip-install-k3s)

   [success] Skipping (--skip-configure-iptables)

   [success] Skipping (iptables is not on the PATH nor run via Alternatives)

   [success] Skipping (iptables version is older than 1.8.0)

   [success] Skipping (iptables version is newer than 1.8.3)

   [success] Skipping (iptables is in legacy mode)

   [success] Skipping (iptables is not run via Alternatives)
   ```

   and **install-lockss** will successfully proceed to the next phase, *Configuring firewalld for K3s* (Section 3.3.6).

2. Otherwise, you will receive the following prompt:

   Switch iptables to legacy mode via Alternatives?

   Enter `Y` to accept the proposed **iptables** configuration or `N` to bypass (or hit `Enter` to accept the default in square brackets).

   - If **install-lockss** was invoked with the `--assume-yes` option, `Y` is automatically entered for you.
   - You may be prompted for your **sudo** password.

   Warning: If you bypass the proposed **iptables** configuration, you will see the warning:

   ```
   [Warning] Leaving iptables unchanged; see manual for details
   ```

   and **install-lockss** will immediately proceed to the next phase, *Configuring firewalld for K3s* (Section 3.3.6). But K3s may malfunction without further intervention; see *Troubleshooting iptables* for details.

3. If the **iptables** configuration attempt fails, you will see one of these error messages:

```
[ERROR] Error deactivating ufw

[ERROR] Error applying update-alternatives to iptables

[ERROR] Error applying update-alternatives to ip6tables

[ERROR] Error flushing iptables

[ERROR] Error reactivating ufw
```

and **install-lockss** will fail.

Troubleshooting

See *Troubleshooting iptables* for remediation details.

---

4. Finally, you will see the message:

```
[success] Configured iptables for K3s
```

and **install-lockss** will successfully proceed to the next phase, *Configuring firewalld for K3s* (Section 3.3.6).

3.3.6 Configuring `firewalld` for K3s

Heading

This phase begins with the heading *Configuring firewalld for K3s...*.

Description

During this phase, **install-lockss** will configure **firewalld** to work with K3s, if applicable.

Steps

1. If **install-lockss** was invoked with the `--skip-configure-firewalld` option (implied by `--skip-install-k3s`), or if **firewalld** is not present or is not running, you will see one of these messages:

```
[success] Skipping (--skip-install-k3s)

[success] Skipping (--skip-configure-firewalld)

[success] Skipping (firewall-cmd is not on the PATH)

[success] Skipping (firewalld is not running)
```

and **install-lockss** will successfully proceed to the next phase, *Configuring ufw for K3s* (Section 3.3.7).

2. If **firewalld** is running, you will receive the following prompt:

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

Enter `Y` to accept the proposed **`firewalld`** configuration or `N` to bypass (or hit `Enter` to accept the default in square brackets).

- If **`install-lockss`** was invoked with the `--assume-yes` option, `Y` is automatically entered for you.
- You may be prompted for your **`sudo`** password.

Warning: If you bypass the proposed **`firewalld`** configuration, you will see the warning:

```
[Warning] Leaving firewalld unchanged; see manual for details
```

and **`install-lockss`** will immediately proceed to the next phase, *Configuring ufw for K3s* (Section 3.3.7). But K3s may malfunction without further intervention; see *Troubleshooting firewalld* for details.

3. If the **`firewalld`** configuration attempt fails, you will see one of these error messages:

```
[ERROR] Could not add 10.42.0.0/16 to firewalld's trusted zone

[ERROR] Could not add 10.43.0.0/16 to firewalld's trusted zone

[ERROR] Could not reload firewalld
```

and **`install-lockss`** will fail.

Troubleshooting

See *Troubleshooting firewalld* for remediation details.

4. Finally, you will see the message:

```
[success] Configured firewalld for K3s
```

and **`install-lockss`** will successfully proceed to the next phase *Configuring ufw for K3s* (Section 3.3.7).

3.3.7 Configuring `ufw` for K3s

Heading

This phase begins with the heading *Configuring firewalld for ufw...*.

Description

During this phase, **`install-lockss`** will configure **`ufw`** to work with K3s, if necessary.

Steps

1. If **install-lockss** was invoked with the `--skip-configure-ufw` option (implied by `--skip-install-k3s`), or if **ufw** is not present or is not active, you will see one of these messages:

   ```
   [success] Skipping (--skip-install-k3s)

   [success] Skipping (--skip-configure-ufw)

   [success] Skipping (ufw is not on the PATH)

   [success] Skipping (ufw is not active)
   ```

   and **install-lockss** will successfully proceed to the next phase, *Configuring CoreDNS for K3s* (Section 3.3.8).

2. If **ufw** is active, you will receive the following prompt:

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

   Enter `Y` to accept the proposed **ufw** configuration or `N` to bypass (or hit `Enter` to accept the default in square brackets).

   - If **install-lockss** was invoked with the `--assume-yes` option, `Y` is automatically entered for you.
   - You may be prompted for your **sudo** password.

   Warning: If you bypass the proposed **ufw** configuration, you will see the warning:

   ```
   [Warning] Leaving ufw unchanged; see manual for details
   ```

   and **install-lockss** will immediately proceed to the next phase, *Configuring CoreDNS for K3s* (Section 3.3.8). But K3s may malfunction without further intervention. See *Troubleshooting ufw* for details.

3. If the **ufw** configuration attempt fails, you will see one of these error messages:

   ```
   [ERROR] Could not allow traffic from 10.42.0.0/16 via ufw

   [ERROR] Could not allow traffic from 10.43.0.0/16 via ufw

   [ERROR] Could not reload ufw
   ```

   and **install-lockss** will fail.

   Troubleshooting

   See *Troubleshooting ufw* for remediation details.

4. Finally, you will see the message:

   ```
   [success] Configured ufw for K3s
   ```

   and **install-lockss** will successfully proceed to the next phase, *Configuring CoreDNS for K3s* (Section 3.3.8).

3.3.8 Configuring CoreDNS for K3s

Heading

This phase begins with the heading *Configuring CoreDNS for K3s...*.

Description

During this phase, **`install-lockss`** will configure CoreDNS to work with K3s, if necessary.

Steps

1. If **`install-lockss`** was invoked with the `--skip-configure-coredns` option (implied by `--skip-install-k3s`), or if your system's DNS configuration will simply work with CoreDNS, you will see one of these messages:

   ```
   [success] Skipping (--skip-install-k3s)

   [success] Skipping (--skip-configure-dns)

   [success] Using system resolv.conf files
   ```

   and **`install-lockss`** will successfully proceed to the next phase, *Installing K3s* (Section 3.3.9).

2. If your system's DNS configuration will not work with CoreDNS, or if **`install-lockss`** was invoked with the `--force-dns-prompt` option, you will receive a message including `CoreDNS does not allow a loopback address to be given to Kubernetes pods as an upstream DNS server`, 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 value will be offered to you in square brackets, consisting of non-loopback IP addresses collected from your machine's DNS configuration; you can simply hit `Enter` to accept the suggested value.

   - If **`install-lockss`** was invoked with the `--assume-yes` option, the suggested value is automatically accepted witout the prompt.

3. If the creation of the CoreDNS configuration file fails, you will see error messages similar to these:

   ```
   [ERROR] Could not create /etc/lockss

   [ERROR] Error rendering config/templates/k3s/resolv.conf.mustache to config/resolv.
   ↪conf

   [ERROR] Could not copy config/resolv.conf to /etc/lockss/resolv.conf
   ```

   and **`install-lockss`** will fail.

   Troubleshooting

   See *Troubleshooting CoreDNS* for remediation details.

4. Finally, you will see the message:

```
[success] Configured CoreDNS for K3s
```

and **install-lockss** will successfully proceed to the next phase, *Installing K3s* (Section 3.3.9).

3.3.9 Installing K3s

Heading

This phase begins with the heading *Installing K3s...*.

Description

During this phase, **install-lockss** will install K3s.

Steps

1. If **install-lockss** was invoked with the `--skip-install-k3s` option, you will see the message:

   ```
   [success] Skipping (--skip-install-k3s)
   ```

   and **install-lockss** will successfully proceed to the next phase, *Testing the K3s Node* (Section 3.3.10).

2. Next, **install-lockss** will determine if K3s needs to be installed or upgraded. There are five cases:

   - Case 1: If K3s is not present, **install-lockss** will display `K3s is not present`, and will install K3s.
   - Case 2: If the expected version of K3s is already present, **install-lockss** will display `K3s version` *`installed_version`* `is already installed; skipping`, and will *not* install K3s.
   - Case 3: If a more recent version of K3s is present, **install-lockss** will display `Detected K3s version` *`installed_version`* `is more recent than expected version` *`expected_version`*, and will *not* install K3s.
   - Case 4: If an older version of K3s is present, **install-lockss** will display `Detected K3s version` *`installed_version`* `is older than expected version` *`expected_version`* and you will receive the following prompt:

     Upgrade K3s from {installed_version} to {expected_version}?

     Enter `Y` and **install-lockss** will install the newer K3s version, or `N` and **install-lockss** will *not* install the newer K3s version (or hit `Enter` to accept the default in square brackets).

     – If **install-lockss** was invoked with the `--assume-yes` option, `Y` is automatically entered for you.
   - Case 5: If K3s is detected but the installed and expected version numbers cannot be compared automatically, you will see the following warning:

     `[Warning] Detected K3s version` *`installed_version`*`, expected version` *`expected_version`*`, comparison failure, skipping`

     and **install-lockss** will *not* install K3s.

   If **install-lockss** determines that it will *not* install K3s, it will confirm `Not installing K3s`, then will *skip the next 3 steps*.

   Otherwise, **install-lockss** will confirm `Installing K3s version` *`expected_version`* and will simply proceed to the next step.

3. First, **install-lockss** will warn you that if the directory K3s uses to store state data (by default `/var/lib/rancher/k3s`) is space-limited, you should specify a different directory. You will see the following prompt:

   K3s state data directory

   Enter a suitable directory path for the K3s state data directory, or simply hit `Enter` to accept the default in square brackets.

   - If **install-lockss** was invoked with the `--k3s-data-dir=DIR` option, *DIR* will automatically be used without the prompt.
   - If **install-lockss** was invoked with the `--assume-yes` option, the default is automatically used without the prompt.

4. Next, **install-lockss** will detect the filesystem type backing the K3s state data directory, because some filesystem types (like NFS, or XFS with legacy `ftype=0`) are not suitable or require special handling:

   a. If the filesystem type backing the K3s state data directory cannot be inferred automatically, you will see the warning:

      `[Warning] Filesystem type of k3s_dir unknown (findmnt not present); proceeding`

      and **install-lockss** will *proceed to the next step*.

   b. If the filesystem type backing the K3s state data directory is NFS, you will see the error message:

      `[ERROR] Filesystem type of k3s_dir (k3s_mountpoint) is NFS; see manual`

      and **install-lockss** will fail.

   c. If the filesystem type backing the K3s state data directory is XFS, **install-lockss** will determine its `ftype` (an internal characteristic):

      (i) If the XFS filesystem backing the K3s state data directory has legacy `ftype=0`, you will see the error message:

      ```
      [ERROR] Filesystem type of k3s_dir (k3s_mountpoint) is XFS with legacy
      ftype=0; see manual for workaround
      ```

      and **install-lockss** will fail.

      (ii) If the XFS filesystem backing the K3s state data directory does not have legacy `ftype=0`, you will see one of these messages:

      ```
      Filesystem type of k3s_dir (k3s_mountpoint) is XFS with ftype=1;
      proceeding
      ```

      ```
      Filesystem type of k3s_dir (k3s_mountpoint) is XFS but not with legacy
      ftype=0; proceeding
      ```

      and **install-lockss** will *proceed to the next step*.

      (iii) If the `ftype` of the XFS filesystem backing the K3s state data directory cannot be inferred automatically, you will see the warning:

      ```
      [Warning] Filesystem type of k3s_dir (k3s_mountpoint) is XFS but ftype
      unknown (xfs_info not present); proceeding
      ```

      and **install-lockss** will *proceed to the next step*.

      (iv) Otherwise, **install-lockss** will display the filesystem type backing the K3s state data directory in a message similar to this:

      `Filesystem type of k3s_dir (k3s_mountpoint) is fs_type; proceeding`

      (for some filesystem type *fs_type*, for example `ext4`), and will *proceed to the next step*.

5. Finally, the K3s Installer will be downloaded from https://get.k3s.io/ and invoked with suitable options.

   Depending on your operating system and other factors, the K3s Installer may install additional software packages or configure system components, using **sudo** if necessary (which may prompt for the user's **sudo** password).

   If the K3s Installer does not succeed, it will display its own error messages, then **install-lockss** will fail.

   Troubleshooting

   Error messages that the K3s Installer may display include:

   ```
   [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/8/noarch/k3s-
   ↪selinux-0.3-0.el8.noarch.rpm

   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
   ```

   See *Troubleshooting the K3s Installer* for remediation details.

6. Whether or not K3s was installed, **install-lockss** will store Kubernetes configuration data as the `lockss` user in the file `config/k8s.cfg`, relative to the LOCKSS Installer home directory. If the creation of the file fails, you will see one of these error messages:

   ```
   [ERROR] Could not write k8s.cfg

   [ERROR] Could not append to k8s.cfg
   ```

   and **install-lockss** will fail.

   Troubleshooting

   Check file permission mismatches between the user running **install-lockss** and the `lockss-installer/config` directory, then try again.

7. Finally, you will see the message:

   ```
   [success] Installed K3s
   ```

   and **install-lockss** will successfully proceed to the next phase (*Testing the K3s Node*).

3.3.10 Testing the K3s Node

Heading

This phase begins with the heading *Testing the K3s node...*.

Description

During this phase, **install-lockss** runs a series of tests to verify that the K3s node is operational.

Steps

1. If **install-lockss** was invoked with the `--skip-test-k3s` option (implied by `--skip-install-k3s`), you will see one of these messages:

```
[success] Skipping (--skip-install-k3s)

[success] Skipping (--skip-test-k3s)
```

   and **install-lockss** will successfully proceed to the next phase, *Completion of the LOCKSS Installation Process* (Section 3.3.11).

2. Next, **install-lockss** will run a series of tests. If a test fails, you will see one of these error messages:

```
[ERROR] k8s.cfg not found

[ERROR] Error reading K8S_FLAVOR

[ERROR] K8S_FLAVOR is not set

[ERROR] K8S_FLAVOR is not k3s

[ERROR] Error reading KUBECTL_CMD

[ERROR] KUBECTL_CMD is not set

[ERROR] k3s command of KUBECTL_CMD is not on the PATH

[ERROR] Command failed (kubectl version)

[ERROR] Timeout waiting for the K3s node to be ready

[ERROR] Command failed (kubectl get node)

[ERROR] Unexpected number of K3s nodes

[ERROR] Timeout waiting for the CoreDNS pod to be running and ready

[ERROR] Command failed (kubectl get pod)

[ERROR] Unexpected number of CoreDNS pods
```

(continues on next page)

(continued from previous page)

```
[ERROR] Timeout waiting for the DNS service to be present

[ERROR] Command failed (kubectl get service)

[ERROR] Unexpected number of kube-dns services

[ERROR] Unexpected kube-dns service type

[ERROR] Timeout waiting for DNS resolution

[ERROR] Unexpected Cluster-IP
```

and **install-lockss** will fail.

Troubleshooting

The reasons for some of these tests failing vary. Some wait for K3s to start up and retry a number of times but eventually give up, even though K3s will eventually come up fully. You can invoke just this portion of **lockss-install** by invoking:

```
install-lockss --test-k3s
```

or equivalently:

```
install-lockss -T
```

You can also alter the number of retries and the number of seconds between retries with `--retries=`*N* and `--wait=`*S* respectively.

Other problems may require reaching out to the LOCKSS support team at for assistance.

---

3. Finally, you will see the message:

```
[success] Tested the K3s node
```

and **install-lockss** will successfully proceed to the next phase, *Completion of the LOCKSS Installation Process* (Section 3.3.11).

3.3.11 Completion of the LOCKSS Installation Process

If all phases completed successfully, you will see the message:

```
[success] Successful completion of the LOCKSS installation process
```

and **install-lockss** will terminate.

3.3.12 Checking the K3s Configuration

Tip: This section is optional.

K3s comes with **k3s check-config**, a configuration checker tool. The K3s configuration checker is capable of detecting complex underlying system situations that definitely require fixing (or applications running in the K3s cluster will not be able to function properly). On the other hand, the versions of the K3s configuration checker available at the time LOCKSS 2.0-alpha4 and LOCKSS 2.0-alpha5 were released contained bugs that reported spurious issues that are either inaccurate or moot. As a result, we have decided against running **k3s check-config** as part of **install-lockss** at this time, to avoid unnecessary interruptions in the installation of the LOCKSS system in many cases where there is no particular cause for concern.

That being said, we still recommend running **k3s check-config** and interpreting the results using the *Troubleshooting the K3s Configuration Checker* section of the manual:

1. Run this command:

```
k3s check-config
```

2. The following error messages in the output are indicative of system situations that require attention:

```
/usr/sbin iptables v1.8.2 (nf_tables): should be older than v1.8.0, newer than v1.8.
↪3, or in legacy mode (fail)

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

apparmor: enabled, but apparmor_parser missing (fail)
```

Troubleshooting

See *Troubleshooting the K3s Configuration Checker* for details.

3. The following error messages in the output can be ignored:

```
cgroup hierarchy: nonexistent?? (fail)

links: aux/ip6tables should link to iptables-detect.sh (fail)
links: aux/ip6tables-restore should link to iptables-detect.sh (fail)
links: aux/ip6tables-save should link to iptables-detect.sh (fail)
links: aux/iptables should link to iptables-detect.sh (fail)
links: aux/iptables-restore should link to iptables-detect.sh (fail)
links: aux/iptables-save should link to iptables-detect.sh (fail)

swap: should be disabled

CONFIG_INET_XFRM_MODE_TRANSPORT: missing
```

Troubleshooting

See *Troubleshooting the K3s Configuration Checker* for details.

4. For other error messages, check the official K3s documentation, search for K3s issues database on GitHub or the Web for resources matching your error message or operating system, and/or contact us so we can help investigate and document for future reference.

CHAPTER

FOUR

CONFIGURING THE LOCKSS SYSTEM

After installing the LOCKSS system, you will configure it with the **`configure-lockss`** script. If you have experience with classic LOCKSS daemon version 1.x, this is the equivalent of **`hostconfig`**.

4.1 Before Invoking `configure-lockss`

You will need to gather information to answer configuration questions asked by **`configure-lockss`**, including:

- The name (FQDN) of the host.
- The IP address of the host, and if behind NAT, the external IP address for NAT.
- The mail relay host, and optionally mail credentials, for sending e-mail from the host.
- The e-mail address for the administrator of the system.
- The configuration URL and preservation group or groups corresponding to the LOCKSS network your system is joining.
- The path for the primary content data storage area, any additional content data storage areas, log data storage area and temporary data storage area.
- Username and password for the Web user interfaces.
- A password for the PostgreSQL database.
  - Alternatively, if using an existing PostgreSQL database, the host name, port, schema, username and password for the external PostgreSQL database, as well as a prefix for database names.
- A username and password for the Solr database.
  - Alternatively, if using an existing Solr database, the host name, port, username and password for the external Solr database, as well as the core name for the LOCKSS repository.
- Whether you wish to use the LOCKSS Metadata Extraction Serice, LOCKSS metadata Service, LOCKSS SOAP Compatibility Service, OpenWayback Web replay engine, and Pywb Web replay engine.

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`.

4.2 Invoking `configure-lockss`

To invoke **`configure-lockss`**, simply run this command in the `lockss` user's `lockss-installer` directory as `lockss`[1]:

```
scripts/configure-lockss
```

The script will begin with the first series of configuration questions, about *Network Settings*.

4.3 Network Settings

4.3.1 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`.

4.3.2 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.

4.3.3 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.

4.3.4 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.

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

4.3.5 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`.

4.3.6 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`.

4.4 Mail Settings

4.4.1 Mail Relay

Prompt: *Mail relay for this machine*

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

4.4.2 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.

4.4.3 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.

4.5 Preservation Network Settings

4.5.1 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 verify the authenticity of the configuration server.

4.5.2 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`).

4.5.3 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`.

4.6 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.

4.6.1 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.

4.6.2 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.

4.6.3 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.

4.7 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.

4.8 Database Settings

4.8.1 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*.

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.

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.

4.8.2 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*.

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.

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.

4.9 LOCKSS Services

4.9.1 Metadata Query Service

Prompt: *Use LOCKSS Metadata Query Service?*

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

4.9.2 Metadata Extraction Service

Prompt: *Use LOCKSS Metadata Extraction Service?*

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

4.9.3 SOAP Compatibility Service

Prompt: *Use LOCKSS SOAP Compatibility Service?*

Enter `Y` if you want the SOAP compatibility servvice to be run, otherwise `N`.

4.10 Web Replay Settings

4.10.1 Pywb

Prompt: *Use LOCKSS Pywb Service?*

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

4.10.2 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.

4.11 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.

These sections are under construction. The user interface of the LOCKSS 2.x is currently largely similar to that of the classic LOCKSS system (1.x); you can refer to the classic LOCKSS system manual in the meantime for similar information.

6.1 Using the LOCKSS Configuration Service

Note: This section 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 section 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".

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.

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 section 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 section 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 section 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 47, 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 47, 1]:

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

See Also

- Web UI (Dashboard) on the Kubernetes website.

CHAPTER

SEVEN

TROUBLESHOOTING THE LOCKSS SYSTEM

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

Sections include guidance for troubleshooting networking-related components like **iptables**, **firewalld**, and **ufw**, and K3s-related components like CoreDNS, the K3s Installer, the K3s Configuration Checker.

7.1 Known Issues

This section was last updated: 2023-01-23.

Expired K3s Certificate

LOCKSS-provided scripts like `scripts/stop-lockss` or `scripts/restart-lockss` that interact with the K3s server may display error messages that include:

```
error: You must be logged in to the server (Unauthorized)
```

In parallel, the host system log might contain messages similar to:

```
client_builder_dynamic.go:197: get or create service account failed: Get "https://127.0.
↪0.1:6444/api/v1/namespaces/kube-system/serviceaccounts/generic-garbage-collector":␣
↪x509: certificate has expired or is not yet valid: current time 2023-01-22T03:32:05-
↪08:00 is after 2023-01-21T02:13:28Z
```

This is caused by a bug in the version of K3s that currently ships with the LOCKSS system, whereby the auto-rotation of a certificate after the K3s server runs continuously for one year does not take place correctly. Future releases of the LOCKSS system will use a newer version of K3s without this bug. In the meantime, to work around this problem, simply restart the K3s service with **systemctl** as `root`:

```
systemctl restart k3s
```

Scripts that interact with the K3s server should then again work as expected.

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 a future release of the system.

7.2 Troubleshooting `iptables`

This section provides troubleshooting information for the *Configuring iptables for K3s* phase of *Running the LOCKSS Installer*.

7.2.1 Switch iptables to legacy mode via Alternatives

K3s 1.21.5+k3s1 (the version used by LOCKSS 2.0-alpha6) does not always work with **`iptables`** version 1.8.0-1.8.3 when run via Alternatives but not in `legacy` mode, for instance in some Debian or Ubuntu systems[1]. If **`install-lockss`** detects this situation, you will see a warning message and the following prompt[2]:

Switch iptables to legacy mode via Alternatives?

[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 *Configuring iptables for K3s*.

Enter `Y` to accept the proposed **`iptables`** configuration. **If you bypass the proposed configuration, K3s may malfunction.**

The remediation attempted by **`install-lockss`** is equivalent to:

```
# Required only if ufw is 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 only if ufw was active
ufw enable
```

7.2.2 Post-Installation Changes to `iptables`

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), run this command (relative to the lockss-installer-directory) as a privileged user who can become `root` via **`sudo`**[3]:

```
scripts/install-lockss --configure-iptables
```

This will run only the *Configuring iptables for K3s* phase of **`install-lockss`**.

7.3 Troubleshooting `firewalld`

This section provides troubleshooting information for the *Configuring firewalld for K3s* phase of *Running the LOCKSS Installer*.

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

If your system is running the **`firewalld`** firewall, it is necessary to add K3s' pod and service subnets[1] to **`firewalld`**'s `trusted` zone for K3s to work properly[2]. If **`install-lockss`** detects this situation, you will see a warning message

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

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

[2] For operating systems in the RHEL family (CentOS, Rocky Linux, AlmaLinux OS...), 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-lockss`** 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

and the following prompt[3]:

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

Enter `Y` to accept the proposed **firewalld** configuration. **If you bypass the proposed configuration, K3s may malfunction.**

The **firewalld** configuration attempted by **install-lockss** is equivalent to[Page 51, 1]:

```
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
```

7.3.2 Post-Installation Changes to `firewalld`

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 (relative to the lockss-installer-directory) as a privileged user who can become `root` via **sudo**[4]:

```
scripts/install-lockss --configure-firewalld
```

This will run only the *Configuring firewalld for K3s* phase of **install-lockss**.

7.4 Troubleshooting `ufw`

This section provides troubleshooting information for the *Configuring ufw for K3s* phase of *Running the LOCKSS Installer*.

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

If your system is running the **ufw** firewall, it is necessary to allow traffic from K3s' pod and service subnets[1] via **ufw** for K3s to work properly[2]. If **install-lockss** detects this situation, you will see a warning message and the following prompt[3]:

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

Enter `Y` to accept the proposed **ufw** configuration. **If you bypass the proposed configuration, K3s may malfunction.**

The **firewalld** configuration attempted by **install-lockss** is equivalent to[1]:

[3] See *Configuring firewalld for K3s*.
[4] See *Running Commands as a Privileged User*.
[1] By default, K3s' pod subnet is 10.42.0.0/16 and service subnet is 10.43.0.0/16.
[2] References:

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

[3] See *Configuring ufw for K3s*.

```
ufw allow from 10.42.0.0/16 to any

ufw allow from 10.43.0.0/16 to any

ufw reload
```

7.4.2 Post-Installation Changes to `ufw`

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 (which is relative to the lockss-installer-directory) as a privileged user who can become `root` via **`sudo`**[4]:

```
scripts/install-lockss --configure-ufw
```

This will run only the *Configuring ufw for K3s* phase of **`install-lockss`**.

7.5 Troubleshooting CoreDNS

This section provides troubleshooting information for the *Configuring CoreDNS for K3s* phase of *Running the LOCKSS Installer*.

7.5.1 CoreDNS does not allow a loopback address to be given to Kubernetes pods as an upstream DNS server

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 **`install-lockss`** detects this situation, you will see a warning message including `CoreDNS does not allow a loopback address to be given to Kubernetes pods as an upstream DNS server`, 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 or hit `Enter` to accept the proposed value.

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

[1] References:

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

[2] See *Configuring CoreDNS for K3s*.

7.5.2 Post-Installation Changes to DNS

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 (which is relative to the lockss-installer-directory) as a privileged user who can become `root` via **sudo**[3]:

```
scripts/install-lockss --configure-coredns
```

This will run only the *Configuring CoreDNS for K3s* phase of **install-lockss**.

7.6 Troubleshooting the K3s Installer

The LOCKSS Installer's **install-lockss** script installs K3s by executing Rancher's official K3s Installer from https://get.k3s.io/, after checking that various system, firewall and DNS prerequisites are addressed (see *Running the LOCKSS Installer*). However, the installation 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 a specific error message.

7.6.1 Enabling User Namespaces in RHEL 7 and CentOS 7

K3s requires user namespaces, a feature generally available and enabled in many Linux flavors. However, some RHEL 7 and CentOS 7 systems do not have user namespace enabled by default. This can cause the *Checking K3s Prerequisites* or *Testing the K3s Node* phases of **install-lockss** or the optional *Checking the K3s Configuration* phase to fail.

To resolve this issue in RHEL 7 or CentOS 7[1]:

1. Edit the file `/etc/default/grub` as `root`[2].
   1. Look for the line beginning with `GRUB_CMDLINE_LINUX=`, for example:

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

      This line defines a space-separated list of boot arguments.

   2. Add `user_namespace.enable=1` to the beginning of the space-separated list, 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] See *Running Commands as a Privileged User*.

[1] References:

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

[2] See *Running Commands as root*.

Note: If EFI is in use, the file to use for the `-o` option will not be `/boot/grub2/grub.cfg` exactly. For example on CentOS, it may be `/boot/efi/EFI/centos/grub.cfg`.

3. Reboot the system.
4. Check that the change took effect. To do so, look at the contents of `/proc/cmdline` (for example type `cat /proc/cmdline`) and verify that it now contains `user_namespace.enable=1`.

Note: If the change did not take effect, it could be an indication that a different `grub.cfg` file is needed for the `-o` option of **`grub2-mkconfig`** command above.

7.6.2 Installing `apparmor_parser`

K3s uses Apparmor in systems where it is enabled. However, some systems, especially OpenSUSE systems, have Apparmor enabled but **`apparmor_parser`** is not installed by default. This can cause the *Checking K3s Prerequisites* or *Installing K3s* phases of **`install-lockss`** or the optional *Checking the K3s Configuration* phase to fail.

To resolve this issue in OpenSUSE, run these **`zypper`** commands as `root`[Page 54, 2]:

```
zypper refresh

zypper --non-interactive install apparmor-parser
```

or equivalently:

```
zypper refresh

zypper -n install apparmor-parser
```

Tip: In other Linux flavors, use similar package installation commands.

7.6.3 Failed to apply container_runtime_exec_t to /usr/local/bin/k3s

In some Fedora systems, the K3s installer may fail with 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/8/noarch/k3s-selinux-
↪0.3-0.el8.noarch.rpm
```

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

To resolve this problem, run the recommended commands as `root`[Page 54, 2].

7.6.4 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 in Oracle Linux 7, run the following command as `root`[Page 54, 2]:

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

7.7 Troubleshooting the K3s Configuration Checker

After installing K3s[2], you may optionally run the K3s configuration checker **`k3s check-config`**[3] (see *Checking the K3s Configuration*). This configuration checker runs through a more extensive series of tests, covering "required", "generally necessary", and "optional" system aspects needed by K3s.

Some failures, especially in "optional" aspects, may not actually prevent the cluster from working normally, in the limited ways the LOCKSS system uses Kubernetes. 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 a specific error message.

7.7.1 iptables should be older than v1.8.0, newer than v1.8.3, or in legacy mode

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

```
/usr/sbin iptables v1.8.2 (nf_tables): should be older than v1.8.0, newer than v1.8.3,␣
↪or in legacy mode (fail)
```

In previous versions of K3s, this error message was also sometimes phrased as `should be older than v1.8.0 or in legacy mode`.

The **`install-lockss`** script should detect this situation and offer to switch **`iptables`** to legacy mode via Alternatives (see *Troubleshooting iptables*). If the error above occurs:

- Verify that the *Configuring iptables for K3s* phase of **`install-lockss`** was not skipped.
- Verify that, if applicable, the proposed **`iptables`** configuration changes in the *Configuring iptables for K3s* phase of **`install-lockss`** were not bypassed.
- Using the *Troubleshooting iptables* section as reference, verify that the remediation attempted by **`install-lockss`** has taken effect.
- Search the K3s issues database for issues related to **`k3s check-config`**, **`iptables`** and your operating system.

[2] See *Installing K3s*.
[3] See *Checking the K3s Configuration*.

7.7.2 User namespaces disabled

In some RHEL 7 and CentOS 7 systems, you may receive the following error message:

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

To resolve this issue, see *Enabling User Namespaces in RHEL 7 and CentOS 7*.

7.7.3 apparmor enabled but apparmor_parser missing

In some systems where Apparmor is enabled, you may receive the following error:

```
apparmor: enabled, but apparmor_parser missing (fail)
```

To resolve this issue, see *Installing apparmor_parser*.

7.7.4 cgroup hierarchy nonexistent

In some Arch Linux, Debian and Fedora systems, you may see the following error message:

```
cgroup hierarchy: nonexistent?? (fail)
```

K3s supports `cgroup2` but **`k3s check-config`** version 1.21.5+k3s1 (used in LOCKSS 2.0-alpha5) does not process this condition correctly. **This warning can be ignored.**

7.7.5 links: aux/iptables should link to iptables-detect.sh

In some Fedora and OpenSUSE systems, you may encounter six related error messages like the following:

```
links: aux/ip6tables should link to iptables-detect.sh (fail)
links: aux/ip6tables-restore should link to iptables-detect.sh (fail)
links: aux/ip6tables-save should link to iptables-detect.sh (fail)
links: aux/iptables should link to iptables-detect.sh (fail)
links: aux/iptables-restore should link to iptables-detect.sh (fail)
links: aux/iptables-save should link to iptables-detect.sh (fail)
```

This is due to a bug in **`k3s check-config`**[6], triggered in environments where there is no **`iptables`** system package installed. **This warning can be ignored.**

[6] Reference:

- https://github.com/k3s-io/k3s/issues/4066
  - https://github.com/k3s-io/k3s/issues/4066#issuecomment-925137706

7.7.6 swap should be disabled

This warning can be ignored:

```
swap: should be disabled
```

7.7.7 CONFIG_INET_XFRM_MODE_TRANSPORT missing

This warning can be ignored:

```
CONFIG_INET_XFRM_MODE_TRANSPORT: missing
```

7.8 Troubleshooting OverlayFS with XFS

This section provides troubleshooting information for the check_xfs phase of *Running the LOCKSS Installer*.

7.8.1 Filesystem backing `/var/lib/rancher` is an XFS filesystem with `ftype=0`

The OverlayFS driver used by K3s will fail and prevent containers from starting, if the filesystem backing `/var/lib/rancher` is an XFS filesystem configured with the legacy `ftype=0` setting that some older Linux systems defaulted to when creating XFS filesystems. To view or verify the setting of `ftype` from an XFS filesystem, use `xfs_info` (see step 1 below).

The recommended way fix to this issue is to reformat the XFS filesystem with `ftype=1`, or with another filesystem such as ext4. If this is not possible (e.g., because you are performing an in-place migration from LOCKSS 1.x to 2.x and the filesystem backing `/var/lib/rancher` is shared with other parts of the system), the workaround is to create a new XFS filesystem on another partition or a loopback device backed by a file on an existing filesystem.

Setup of the latter is described below. Commands should be run as the `root` user[1].

1. (Optional) Verify the existing XFS filesystem has `ftype=0`:

   ```
   xfs_info BLOCK_DEVICE
   ```

   ...where `BLOCK_DEVICE` is the block device of the filesystem backing `/var/lib/rancher`.

   Tip: Use `findmnt` to determine the block device of the filesystem backing `/var/lib/rancher`:

   ```
   findmnt --target /var/lib/rancher
   ```

   The output should look similar to the following (some parameters may be different):

   ```
   TARGET SOURCE           FSTYPE OPTIONS
   /      /dev/mapper/root xfs    rw,relatime,attr2,inode64,logbufs=8,logbsize=32k,
   ↪noquota
   ```

   In the example above, the block device is `/dev/mapper/root` (i.e., the `SOURCE` column).

   The output from `xfs_info` should look similar to the following (some parameters may be different):

[1] See *Running Commands as root*.

```
meta-data=/dev/mapper/root isize=512    agcount=4, agsize=4194304 blks
         =                       sectsz=4096  attr=2, projid32bit=1
         =                       crc=1        finobt=1, sparse=1, rmapbt=0
         =                       reflink=1    bigtime=1 inobtcount=1 nrext64=0
data     =                       bsize=4096   blocks=16777216, imaxpct=25
         =                       sunit=0      swidth=0 blks
naming   =version 2              bsize=4096   ascii-ci=0, ftype=0
log      =internal log           bsize=4096   blocks=16384, version=2
         =                       sectsz=4096  sunit=1 blks, lazy-count=1
realtime =none                   extsz=4096   blocks=0, rtextents=0
```

Note: In the example above, `ftype=0`. If your output contains `ftype=1`, there is nothing wrong with your XFS filesystem and you should not proceed with the instructions on this page!

2. Stop the K3s service:

```
systemctl stop k3s.service
```

3. Rename the existing `/var/lib/rancher` directory:

```
mv /var/lib/rancher /var/lib/rancher.backup
```

4. Allocate a file for the loopback device: This file will contain the XFS filesystem that will hold the K3s runtime environment and container images. We recommend allocating a minimum of 16GB. Allocate the file on a filesystem with sufficient space.

Note: In this and the following examples, we allocate and use the file located at `/var/lib/rancherfs.img`.

```
fallocate -l 16g /var/lib/rancherfs.img
```

5. Create a new XFS filesystem with `ftype` explicitly set to 1:

```
mkfs.xfs -n ftype=1 /var/lib/rancherfs.img
```

You may verify `ftype=1` by running:

```
xfs_info /var/lib/rancherfs.img
```

6. Mount the loopback to `/var/lib/rancher`:

```
mount --mkdir -o loop /var/lib/rancherfs.img /var/lib/rancher
```

7. Edit `/etc/fstab` to make the loopback persistent across reboots: Use your favorite text editor to open `/etc/fstab` and append the following line to it:

```
/var/lib/rancherfs.img   /var/lib/rancher   xfs   loop   0 0
```

8. Copy the existing K3s runtime environment to the new XFS filesystem:

```
cp -r /var/lib/rancher.backup/* /var/lib/rancher/.
```

9. Finally, restart the K3s service:

```
systemctl start k3s.service
```

10. (Optional) If setup of the loopback was successful and K3s is running correctly, you may reclaim space by removing `/var/lib/rancher.backup`:

```
rm -rf /var/lib/rancher.backup
```

CHAPTER

EIGHT

SYSTEM ADMINISTRATION TASKS

This chapter covers some common system administration tasks related to running a LOCKSS system.

Sections include how to run commands as the `root` user, a privileged user who can become `root` via **sudo**, or the `lockss` user; how to update a Linux operating system; and how to install various system utilities like **curl** or **wget**.

8.1 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.

8.1.1 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`.

8.1.2 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`.

8.2 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` and `lockss` via **sudo**.

Compared to running an entire command or script directly 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, and only those commands that need to read or write files as the `lockss` user will be run as the `lockss` user.

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.

8.3 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.

8.3.1 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 62, 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`.

8.3.2 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`.

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

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

8.4 Operating System Updates

You will want to update your Linux operating system at different times:

- Before *Upgrading From LOCKSS 2.0-alpha5*.
- Before *Installing the LOCKSS System* for the first time.
- When security-related updates are released for the Linux kernel (which requires rebooting the machine) or other installed software packages.

Your system may be set up for automatic updates or your system administrator may have policies for which packages can be udpated and when.

If you wish to update software packages manually, select your operating system below and follow the instructions as `root`[1]:

AlmaLinux OS

To manually update software packages, run this **`dnf`** command as `root`:

```
dnf update
```

Arch Linux

To manually update software packages, run this **`pacman`** command as `root`:

```
pacman --sync --refresh --sysupgrade
```

or equivalently:

```
pacman -Syu
```

CentOS

CentOS Stream

To manually update software packages, run this **`dnf`** command as `root`:

```
dnf update
```

[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 root*.

CentOS 7

To manually update software packages, run this **yum** command as `root`:

```
yum update
```

Debian

To manually update software packages, run these **apt** commands as `root`:

```
apt update

apt upgrade
```

EuroLinux

EuroLinux 8

To manually update software packages, run this **dnf** command as `root`:

```
dnf update
```

EuroLinux 7

To manually update software packages, run this **yum** command as `root`:

```
yum update
```

Fedora Linux

To manually update software packages, run this **dnf** command as `root`:

```
dnf update
```

Linux Mint

To manually update software packages, run these **apt** commands as `root`:

```
apt update

apt upgrade
```

OpenSUSE

OpenSUSE Tumbleweed

To manually update software packages, run these **zypper** commands as `root`:

```
zypper refresh

zypper update
```

OpenSUSE Leap

To manually update software packages, run these **zypper** commands as `root`:

```
zypper refresh

zypper update
```

Oracle Linux

Oracle Linux 8-9

To manually update software packages, run this **dnf** command as `root`:

```
dnf update
```

Oracle Linux 7

To manually update software packages, run this **yum** command as `root`:

```
yum update
```

RHEL

RHEL 8-9

To manually update software packages, run this **dnf** command as `root`:

```
dnf update
```

RHEL 7

To manually update software packages, run this **yum** command as `root`:

```
yum update
```

Rocky Linux

To manually update software packages, run this **dnf** command as `root`:

```
dnf update
```

Scientific Linux

To manually update software packages, run this **yum** command as `root`:

```
yum update
```

Ubuntu

To manually update software packages, run these **apt** commands as `root`:

```
apt update

apt upgrade
```

8.5 Installing `curl`

Downloading and running the LOCKSS Installer requires either **curl** or **wget**. Most typical Linux systems have at least one installed by default. You can check by typing `curl --version` or `wget --version` and verifying that the output is not an error message. This section describes how to install **curl** if necessary. (If you prefer to install **wget**, see *Installing wget*.)

Select your operating system below and follow the instructions as root[1]:

AlmaLinux OS

To install **curl**, run this **dnf** command (as `root`):

```
dnf --assumeyes install curl
```

or equivalently:

```
dnf -y install curl
```

[1] See *Running Commands as root*.

Arch Linux

To install **curl**, run this **pacman** command (as `root`):

```
pacman --sync --noconfirm curl
```

or equivalently:

```
pacman -S --noconfirm curl
```

CentOS

CentOS Stream

To install **curl**, run this **dnf** command (as `root`):

```
dnf --assumeyes install curl
```

or equivalently:

```
dnf -y install curl
```

CentOS 7

To install **curl**, run this **yum** command (as `root`):

```
yum --assumeyes install curl
```

or equivalently:

```
yum -y install curl
```

Debian

To install **curl**, follow these instructions (as `root`):

1. Run this **apt** command:

   ```
   apt update
   ```

2. Run this **apt** command:

   ```
   apt install --assume-yes curl
   ```

   or equivalently:

   ```
   apt -y install curl
   ```

EuroLinux

EuroLinux 8-9

To install **curl**, run this **dnf** command (as `root`):

```
dnf --assumeyes install curl
```

or equivalently:

```
dnf -y install curl
```

EuroLinux 7

To install **curl**, run this **yum** command (as `root`):

```
yum --assumeyes install curl
```

or equivalently:

```
yum -y install curl
```

Fedora Linux

To install **curl**, run this **dnf** command (as `root`):

```
dnf --assumeyes install curl
```

or equivalently:

```
dnf -y install curl
```

Linux Mint

To install **curl**, follow these instructions (as `root`):

1. Run this **apt** command:

   ```
   apt update
   ```

2. Run this **apt** command:

   ```
   apt install --assume-yes curl
   ```

   or equivalently:

   ```
   apt -y install curl
   ```

OpenSUSE

OpenSUSE Tumbleweed

To install **curl**, run these **zypper** (as `root`):

```
zypper refresh

zypper --non-interactive install curl
```

or equivalently:

```
zypper refresh

zypper -n install curl
```

OpenSUSE Leap

To install **curl**, run these **zypper** (as `root`):

```
zypper refresh

zypper --non-interactive install curl
```

or equivalently:

```
zypper refresh

zypper -n install curl
```

Oracle Linux

Oracle Linux 8-9

To install **curl**, run this **dnf** command (as `root`):

```
dnf --assumeyes install curl
```

or equivalently:

```
dnf -y install curl
```

Oracle Linux 7

To install **curl**, run this **yum** command (as `root`):

```
yum --assumeyes install curl
```

or equivalently:

```
yum -y install curl
```

RHEL

RHEL 8-9

To install **`curl`**, run this **`dnf`** command (as `root`):

```
dnf --assumeyes install curl
```

or equivalently:

```
dnf -y install curl
```

RHEL 7

To install **`curl`**, run this **`yum`** command (as `root`):

```
yum --assumeyes install curl
```

or equivalently:

```
yum -y install curl
```

Rocky Linux

To install **`curl`**, run this **`dnf`** command (as `root`):

```
dnf --assumeyes install curl
```

or equivalently:

```
dnf -y install curl
```

Scientific Linux

To install **`curl`**, run this **`yum`** command (as `root`):

```
yum --assumeyes install curl
```

or equivalently:

```
yum -y install curl
```

Ubuntu

To install **curl**, follow these instructions (as `root`):

1. Run this **apt** command:

   ```
   apt update
   ```

2. Run this **apt** command:

   ```
   apt install --assume-yes curl
   ```

   or equivalently:

   ```
   apt -y install curl
   ```

8.6 Installing wget

Downloading and running the LOCKSS Installer requires either **curl** or **wget**. Most typical Linux systems have at least one installed by default. You can check by typing `curl --version` or `wget --version` and verifying that the output is not an error message. This section describes how to install **wget** if necessary. (If you prefer to install **curl**, see *Installing curl*.)

Select your operating system below and follow the instructions as root[1]:

AlmaLinux OS

To install **wget**, run this **dnf** command (as `root`):

```
dnf --assumeyes install wget
```

or equivalently:

```
dnf -y install wget
```

Arch Linux

To install **wget**, run this **pacman** command (as `root`):

```
pacman --sync --noconfirm wget
```

or equivalently:

```
pacman -S --noconfirm wget
```

[1] See *Running Commands as root*.

CentOS

CentOS Stream

To install **wget**, run this **dnf** command (as root):

```
dnf --assumeyes install wget
```

or equivalently:

```
dnf -y install wget
```

CentOS 7

To install **wget**, run this **yum** command (as root):

```
yum --assumeyes install wget
```

or equivalently:

```
yum -y install wget
```

Debian

To install **wget**, follow these instructions (as root):

1. Run this **apt** command:

   ```
   apt update
   ```

2. Run this **apt** command:

   ```
   apt install --assume-yes wget
   ```

   or equivalently:

   ```
   apt -y install wget
   ```

EuroLinux

EuroLinux 8

To install **wget**, run this **dnf** command (as root):

```
dnf --assumeyes install wget
```

or equivalently:

```
dnf -y install wget
```

EuroLinux 7

To install **wget**, run this **yum** command (as root):

```
yum --assumeyes install wget
```

or equivalently:

```
yum -y install wget
```

Fedora Linux

To install **wget**, run this **dnf** command (as root):

```
dnf --assumeyes install wget
```

or equivalently:

```
dnf -y install wget
```

Linux Mint

To install **wget**, follow these instructions (as root):

1. Run this **apt** command:

   ```
   apt update
   ```

2. Run this **apt** command:

   ```
   apt install --assume-yes wget
   ```

   or equivalently:

   ```
   apt -y install wget
   ```

OpenSUSE

OpenSUSE Tumbleweed

To install **wget**, run these **zypper** (as root):

```
zypper refresh

zypper --non-interactive install wget
```

or equivalently:

```
zypper refresh

zypper -n install wget
```

OpenSUSE Leap

To install **`wget`**, run these **`zypper`** (as `root`):

```
zypper refresh

zypper --non-interactive install wget
```

or equivalently:

```
zypper refresh

zypper -n install wget
```

Oracle Linux

Oracle Linux 8-9

To install **`wget`**, run this **`dnf`** command (as `root`):

```
dnf --assumeyes install wget
```

or equivalently:

```
dnf -y install wget
```

Oracle Linux 7

To install **`wget`**, run this **`yum`** command (as `root`):

```
yum --assumeyes install wget
```

or equivalently:

```
yum -y install wget
```

RHEL

RHEL 8-9

To install **`wget`**, run this **`dnf`** command (as `root`):

```
dnf --assumeyes install wget
```

or equivalently:

```
dnf -y install wget
```

RHEL 7

To install **wget**, run this **yum** command (as `root`):

```
yum --assumeyes install wget
```

or equivalently:

```
yum -y install wget
```

Rocky Linux

To install **wget**, run this **dnf** command (as `root`):

```
dnf --assumeyes install wget
```

or equivalently:

```
dnf -y install wget
```

Scientific Linux

To install **wget**, run this **yum** command (as `root`):

```
yum --assumeyes install wget
```

or equivalently:

```
yum -y install wget
```

Ubuntu

To install **wget**, follow these instructions (as `root`):

1. Run this **apt** command:

   ```
   apt update
   ```

2. Run this **apt** command:

   ```
   apt install --assume-yes wget
   ```

   or equivalently:

   ```
   apt -y install wget
   ```

8.7 Resetting the System to a Blank State

During and after alpha and beta testing, you may have cause to delete all of the stored content in order to start over. This section provides guidance on how to do this without reinstalling completely from scratch.

LOCKSS 2.x stores content and metadata in several places which depend on how it was configured. We've provided a script to facilitate deleting all the right directories, but to provide flexibility and make it difficult to do this accidentally, the script generates a second script that actually performs the deletions.

The procedure consists of deleting the PostgreSQL data directory (`lockss-stack-postgres-data`) and Solr data directory (`lockss-stack-solr-data`) from the primary content data storage area, and the repository data directory (`lockss-stack-repo-data`) from each content data storage area, all of which must be done after shutting down the system.

Directions

1. Stop the stack:

   ```
   scripts/stop-lockss
   ```

2. Generate the deletion script:

   ```
   scripts/upgrades/generate-content-reset-script > /tmp/delcontent
   ```

3. Examine `/tmp/delcontent` if you wish. Note that there is one **`sudo`** command because the PostgreSQL files are not owned by the `lockss` user, which may need modifying in some environments.
4. Make the script executable:

   ```
   chmod +x /tmp/delcontent
   ```

5. Delete the content:

   ```
   /tmp/delcontent
   ```

6. We recommend deleting the script after use:

   ```
   rm /tmp/delcontent
   ```

7. Restart the system:

   ```
   scripts/start-lockss
   ```

CHAPTER

NINE

APPENDIX

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

Sections include security advisories; reference information such as list of ports or component versions; and advanced topics like running the LCAP polling and repair protocol over SSL, working with PostgreSQL, or running the LOCKSS Installer from Git.

9.1 Security Advisories

There are no security advisories currently published for this version of the LOCKSS system.

See also the LOCKSS Documentation Portal's Security pages.

9.2 Release Notes

9.2.1 LOCKSS 2.0.61-alpha6

Released: 2023-01-23

LOCKSS 2.0.61-alpha6 is the first release of the LOCKSS 2.0-alpha6 system.

Release Notes

- New SOAP Compatibility Service provides interim backward compatibility for existing SOAP clients. We intend to remove this service once all SOAP clients have been converted to be REST clients.
- Many of the REST service APIs have changed, to accommodate more usage models (i.e. more typical PLN usage) and to be more consistent. They should now be close to their final form.
- It is no longer necessary to force all content into the crawled-content model of LOCKSS 1.x.
  - Content can be directly stored in the LOCKSS repository without the involvement of a staging server or crawler.
  - The system distinguishes between storing complete HTTP responses (which result from crawls), and plain data files. The audit/repair mechanism works the same for both types of content; the Web replay mechanisms, which involve link rewriting, make sense only for crawled content.
  - Features for preserving directly-stored content:
    - No user-defined plugins are needed. The builtin `NamedPlugin` and `NamedArchicalUnit` are used to group content into archival units.

        - The names of artifacts in `NamedArchicalUnits` need not be URIs -- they can be arbitrary Unicode strings.

- LOCKSS 2.x is now binary-compatible with plugins built for LOCKSS 1.x.
- Memory requirements have been significantly reduced across most services (but are still higher than we would like).
- Improved robustness in the face of temporary network failures, service restarts, etc.
- Many of the capabilities needed to migrate content from 1.x to 2.x are present in this release (in conjunction with the 1.76 release of 1.x).
- **LOCKSS Installer changes**
    - Add SOAP Compatibility Service
    - A script is now included to facilitate deleting all content to restore a cluster to its initial state. This is a multi-step operation to make it hard to do accidentally.
    - Prevent starting new version without running upgrade script if required.
    - Prevent running upgrade script if stack is still running.
    - Fix Solr logging.
    - Adjusted some heap limits.
    - Allow service-specific JVM flags and port mappings to be specified through environment variables, to support profiling and other uses.
- **LOCKSS Configuration Service changes**
    - Added a REST endpoint to calculate AUIDs for non-crawled content.
- **LOCKSS Repository Service changes**
    - API changes
        - `HttpResponse` and plain resource artifacts are stored and fetched in different formats, natural to each. Internally they are stored in WARC Response records or Resource records, respectively.
        - Several other cleanups to resolve inconsistencies.
        - New endpoint to import the individual members of archive files (currently just WARC).
        - The confusingly-named `collection` attribute of `Artifact` objects has been renamed `namespace`.
        - `artifactId` is now `artifactUuid`.
        - The Solr schema has changed. Clusters using our embedded Solr database are updated automatically. In the event you are using an external Solr, please contact us (or start over from scratch).
        - AU size information now includes uncompressed size, compressed size and total disk usage.
    - Performance improvements
        - Reduced file copies when storing Artifacts.
        - Reduced lock contention.
        - Algorithmic improvements to temporary WARC handling to allow garbage collection to run much more frequently, also reducing startup time.
        - Reduced the number of Solr commits and queries.
        - Bulk storage mode removes most Solr overhead for migration and reindexing in case of index lossage.
    - Support for huge filesystems (> 8 EiB).

    - Bug fixes
        * Gracefully handle truncated WARCs resulting from abrupt shutdown.
        * Fixed incorrect character encoding in HTTP headers in WARC response records. (This may cause the headers of `Artifacts` stored with pre-alpha6 releases to not load correctly.)

- **LOCKSS Poller Service changes**
    - Renamed the REST endpoints that exist only for the interim SOAP Compatibility Service to a distinct path (`/ws`).
- Building plugins now performs well-formedness checks, similar to 1.x.
- All changes in LOCKSS 1.75.9 and 1.76.5.

Component Versions

LOCKSS 2.0.61-alpha6 consists of a configurable set of the following components:

- LOCKSS Installer version 2.0.61-alpha6
- LOCKSS Repository Service version 2.13.0
- LOCKSS Configuration Service version 2.7.0
- LOCKSS Poller Service version 2.5.0
- LOCKSS Metadata Extraction Service version 2.6.0
- LOCKSS Metadata Service version 2.5.0
- LOCKSS SOAP Compatibility Service version 1.3.0
- PostgreSQL version 9.6.12
- Apache Solr version 8.11.2 (custom version 8.11.2-slim-1)
- Pywb version 2.4.2 (custom version 2.4.2-3)
- OpenWayback version 2.4.0 (custom version 2.4.0-5)

9.3 Frequently Asked Questions

This section answers some common questions about the LOCKSS system.

I have an existing classic LOCKSS system (version 1.x). Can I upgrade to LOCKSS 2.0-alpha6?

The LOCKSS 2.0-alpha6 release is a technology preview which we are excited to share with the community for testing purposes. It is not yet possible to convert from a classic LOCKSS system (e.g. version 1.76.4) to a LOCKSS 2.0 system for *production* purposes.

However, version 1.76 of the classic LOCKSS system contains a prototype tool to test the migration of archival units (AUs) from a production 1.76 system to a *test* 2.0-alpha6 system, for *testing* purposes. See https://github.com/lockss/community/wiki/Migration-Tool.

To help us advance toward the final LOCKSS 2.0 release, please consider installing and running the LOCKSS 2.0-alpha6 release on a test machine and providing us with your feedback.

I have a LOCKSS system running 2.0-alpha5. Can I upgrade to LOCKSS 2.0-alpha6?

Yes. You are welcome to wipe your testing data from LOCKSS 2.0-alpha6 and start from scratch, but there is an *upgrade path* from LOCKSS 2.0-alpha5.

Can I use my own PostgreSQL database? Can I use my own Solr database?
Yes, you can configure the system to use your institution's Postgres database and/or Solr database -- or you can simply let system run included ones locally.

Can I replay Web content with my own Pywb instance? Can I replay Web content with my own OpenWayback instance?
Yes, you can configure your own Pywb instance and/or OpenWayback instance to connect directly to the LOCKSS Repository Service -- or you can let the system run included ones locally, or you can choose not to run any Web replay engine at all.

9.4 Software License

The LOCKSS system is made available under the terms of the 3-Clause BSD License, a permissive open-source license.

```
Copyright (c) 2000-2022, Board of Trustees of Leland Stanford Jr. University

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice,
this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice,
this list of conditions and the following disclaimer in the documentation
and/or other materials provided with the distribution.

3. Neither the name of the copyright holder nor the names of its contributors
may be used to endorse or promote products derived from this software without
specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE
LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF
SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN
CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE
POSSIBILITY OF SUCH DAMAGE.
```

For more information, visit the Software License page on the LOCKSS Web site.

9.5 Network Ports

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

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

All ports are in the 24600-24699 range, except the LCAP (LOCKSS polling and repair) port which retains its historical value of 9729, and the OpenWayback port which is 8080.

Port	Component
8080	OpenWayback replay engine
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
24650	LOCKSS Metadata Service - REST port
24651	LOCKSS Metadata Service - UI port
24670	LOCKSS Proxy
24671	*reserved*
24672	LOCKSS Audit Proxy
24673	*reserved*
24674	(**UDP**) ICP server
24675	LOCKSS SOAP Compatibility Service - SOAP port
24680	LOCKSS Content Server (ServeContent)
24681	Pywb replay engine
24682	*reserved* (OpenWayback)

9.6 Downloading the LOCKSS Installer using `git`

Warning: The official way to download the LOCKSS Installer is now through the LOCKSS Downloader, rather than cloning the LOCKSS Installer as a Git project as in previous releases.

Follow the instructions in *Installing the LOCKSS System* to use the LOCKSS Downloader.

Tip: Follow the instructions in /appendix/git to install **`git`**, if it is not yet available on your system.

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

1. If you have not previously cloned the LOCKSS Installer, run this command to clone it from GitHub:

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

```
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.

Tip: To avoid a harmless Git warning when updating the LOCKSS Installer from GitHub in the future, run this command within the `lockss-installer` directory created by the **`git clone`** command:

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

2. Checkout the 2.0-alpha5 branch of the LOCKSS Installer by running this command within the `lockss-installer` directory created by the **`git clone`** command earlier:

```
git checkout 2.0-alpha5
```

9.7 Working with PostgreSQL

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

9.7.1 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. 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`:

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

```
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.

9.8 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.

9.8.1 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.

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 `box`*`K`*`.pln.org.pkcs12` and `box`*`K`*`.pln.org.pass` for each one of the *N* host names `box1.pln.org` through `box`*`N`*`.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.

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.

9.8.2 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.