Secure containers

# Introduction

Creating secure containers.

There are three main issues with containers:

* The container has not updated its libraries
* The container’s main software is not hardened.
* The container will become obsolete after a while

### First point

The first point solved by extracting the container’s Dockerfile and repository and slightly modifying it to update its packages at build time.

### Second point

The second point is solved by identifying and applying the main software’s hardening settings

### Third point

The third point is and has always been the biggest problem with containers.

The container runs just fine for a while until a security flaw is discovered for the containers main software or in its libraries.

As a container per definition should not change, incorporating updates, migrations and downtimes is difficult and often forgotten.

# Goals

This document and the attached code solves these three issues.

As a side note we also want the example container to:

* run on CentOS 8
* use MySQL 8.

# Solutions

Solving the first point in the CentOS way that would be to add this to the Dockerfile and disable docker build caching using:





The second point is solved by using the steps from mysql\_secure\_installation script from Oracle added to the container build process:

We disable MySQL root login, remove the installation test database and the anonymous account.

In root-common/usr/share/container-scripts/mysql/common.sh



We remove MySQL users ability to read local files and force connection to the database to use TLS in 8.0/root-common/etc/my.cnf



To start our container we run



How can we prevent the container from becoming obsolete?

Perhaps a better way is to ask “How obsolete are we willing to allow the container to become.”

If we get an email as soon as there are new library or main software updates available, we can then manually, in a controlled fashion, update and migrate the service the container provides to a new, updated, container.

This leads to the container not becoming obsolete and no downtimes since the update migration are done manually, thus any potential migration problems out of reach of automated solutions are dealt with care.

This can easily be done by having a copy of the container which only checks for new updates run in tandem with the container.

The copy only runs this code. Since we can catch a container’s exit status we can then send an email to the operators if new updates exist.



To start the copy we would run for example:

