Sharing files with host machine

Al Forbes edited this page on Oct 21, 2015 · 4 revisions

This page gives instructions for sharing files/volumes/directories/folders between the docker container and the host machine.

Mac

The ability to share volumes with Mac hosts is built into Docker `>= 1.3`. Just link any volume (under /Users/) as you would on Linux:

```
docker run -v /Users/bob/myapp/src:/src [...]
```

Still need to test if this requires any special handling of permissions.

Windows

boot2docker provides two current workarounds for sharing directories on Windows, though more native sharing might follow. The official instructions are here but are more complicated than our preferred method of using the `-v` flag, which you can do like so:

```
docker run -d -p 8787:8787 -v /c/Users/foobar:/home/rstudio/foobar rocker/rstudio
```

In this case, `/c/Users/foobar` corresponds to an existing folder on your computer at `C:/Users/foobar`, and `foobar` can be anything. You can now connect to RStudio at http://192.168.59.103:8787 and log in with rstudio/rstusio. With this `-v` method you can read and write files both ways between Windows and RStudio.

The above approach may not always work, as described in this bug report. The workaround is to either use a double slash or $(pwd):

```
docker run -d -p 8787:8787 -v //c/Users/foobar:/home/rstudio/foobar rocker/rstudio
```

or

```
docker run -d -p 8787:8787 -v /$(pwd):/home/rstudio/foobar rocker/rstudio
```

Linux

As docker runs natively on Linux, you can always link any volume to the container with the `-v` or `--volume` flag, as described in the docker documentation. Note that this overwrites anything on the container that already exists in the target location, and will create the necessary directories if they do not already exist. We recommend running as a non-root user and linking to a subdirectory in the that user's home directory as shown below.

Avoiding permission changes when sharing volumes

By default, our docker containers run as the root user. (See managing users in docker for details.)

Files created or modified by the container will thus become owned by the root user, even after quitting the container. To avoid this problem, it is necessary to run the container using a non-root user.

RStudio-server logins require a non-root user whether or not the container is sharing any volumes with the host. A default user is created automatically when these containers are run with the default command. When sharing volumes with the host, it may be necessary to make sure that the UID of the user created (whether the it is the default user, `rstudio` or a custom username) matches the UID of the host user (the username is irrelevant, only the UID must match). If your host machine has only one non-root user, chances are this is already fine, though you can still set the UID explicitly as described below.

Anyone running our containers as a non-root user can link a directory, including ones at or above `~/`, to the corresponding user's directory on the container. The container will only ever have one user.

If the host machine user has a UID other than 1000 (or 0, for root), the user should specify their UID when running docker, e.g.

```
docker run -d -P -v $(pwd):/home/$USER/foo \
-e USER=$USER  -e USERID=$UID rocker/rstudio
```

to avoid changing the permissions in the linked volume on the host. This is designed for `rstudio`-based logins and runs only when the container is executed without a default command.

Interactive containers

For interactive containers, it is sufficient to run with the flag `--user docker`, e.g. the interactive R container:

```
docker run --rm -it --user docker -v $(pwd):/home/docker/foo -w /home/docker/foo rocker/r-ba
```

note this command links the current working directory to `/home/docker/foo` on the container, where our user `docker` can write to, and also sets this as the working directory for the container. This is analogous to just running R in the working directory.

The only limitation is that the interactive method doesn't handle alternate UIDs. If the user running docker has a different UID, they have to do this more manually. Run a terminal in docker (using rstudio image or above):

```
docker run --rm -it -v $(pwd):/home/$USER/foo \
-e USER=$USER  -e USERID=$UID rocker/rstudio bash
```

and then in the bash shell run:

```
userconf.sh
su $USER
R
```

to setup the user with correct UID, switch to the non-root user, and then run R, etc.