Guided Exercise: Volume Mounting

Inject files into a container by using a bind mount and a volume.

Outcomes

You should be able to:

- Use bind mounts with your containers.
- Use `podman unshare` to troubleshoot permission issues with bind mounts.
- Create named volumes.
- Import files into named volumes.

As the `student` user on the `workstation` machine, use the `lab` command to prepare your system for this exercise.

```
[student@workstation ~]$ lab start persisting-mounting
```

Instructions

1. Examine the Containerfile for the application.

   ```
   [student@workstation ~]$ cat \
   ~/DO188/labs/persisting-mounting/podman-python-server/Containerfile
   FROM registry.access.redhat.com/ubi9/ubi:9.0.0-1468

   RUN adduser \
     --no-create-home \
     --system \
     --shell /usr/sbin/nologin \
     python-server && \
     mkdir /server && \
     chown -R 'python-server:python-server' /server

   WORKDIR /server

   USER python-server

   CMD ["python3", "-m", "http.server"]
   ```

   The resulting container image uses the `/server` directory as the web root directory for the Python HTTP server.
   The `registry.ocp4.example.com:8443/redhattraining/podman-python-server` container image is based on this Containerfile.

2. Copy the `index.html` file to the `~/www` directory.

   The `~/www` directory serves as a bind mount that contains the HTML for the container.

   ```
   [student@workstation ~]$ cp ~/DO188/labs/persisting-mounting/index.html ~/www
   no output expected
   ```

3. Test the `podman-python-server` container with the `~/www` directory mounted as a bind mount.

   Start a container with the following parameters:

   - Bind the `~/www` directory on the host system to the `/server` directory inside the container.
     - Use the `:Z` option to set the correct SELinux label on the bind mount.
   - Name the container `podman-server`.
   - Use the `--rm` option.
   - Use the `-ti` options to display container output.
   - Use the `registry.ocp4.example.com:8443/redhattraining/podman-python-server` image.
   - Bind the port `8000` on the local machine to port `8000` inside the container.

```
[student@workstation ~]$ podman run -ti --rm --name podman-server \
  --volume ~/www:/server:Z -p 8000:8000 \
  registry.ocp4.example.com:8443/redhattraining/podman-python-server
...output omitted...
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/)
```

In a web browser, go to `http://localhost:8000`. You are presented with an error.

Check the container logs:

```
...output omitted...
127.0.0.1 - - [28/Jun/2022 13:21:08] code 404, message No permission to list directory
127.0.0.1 - - [28/Jun/2022 13:21:08] "GET / HTTP/1.1" 404 -
...output omitted...
```

The container does not have permission to access the `index.html` file. Keep the container running.

4. Correct the permission for the `~/www` directory.

   In a new terminal, verify the directory permissions in a new user namespace.

   ```
   [student@workstation ~]$ podman unshare ls -l --directory ~/www
   drwxrwx---. 1 root root 20 Jun 28 14:56 /home/student/www
   ```

   From the perspective of a new container, the directory is owned by the `root` user and group, and other users have no permissions in the directory.

   Verify the group ID inside of the `podman-server` container.

   ```
   [student@workstation ~]$ podman run --rm \
     registry.ocp4.example.com:8443/redhattraining/podman-python-server id
   uid=994(python-server) gid=994(python-server) groups=994(python-server)
   ```

   Change the group of the `~/www` directory and its content to the `python-server` group ID.

   ```
   [student@workstation ~]$ podman unshare chgrp -R 994 ~/www
   no output expected
   ```

   Verify the directory permissions in a new user namespace.

   ```
   [student@workstation ~]$ podman unshare ls -ln --directory ~/www
   drwxrwx---. 1 0 994 20 Jun 28 14:56 /home/student/www
   ```

5. Retest the `podman-server` container with the `~/www` directory mounted as a bind mount.

   In a web browser, access `http://localhost:8000`. You are presented with the `index.html` page.

   Stop the container by pressing **Ctrl**+**c**.

6. Create a named volume with the `index.html` page.

   Create a volume called `html-vol`.

   ```
   [student@workstation ~]$ podman volume create html-vol
   html-vol
   ```

   Change to the `persisting-mounting` lab directory.

   ```
   [student@workstation ~]$ cd ~/DO188/labs/persisting-mounting
   no output expected
   ```

   Import the `index.tar.gz` archive file, which contains `index.html`, into the `html-vol` volume.

   ```
   [student@workstation persisting-mounting]$ podman volume import \
     html-vol index.tar.gz
   no output expected
   ```

7. Start a new container that uses the `podman-python-server` image. Use a volume mount instead of the bind mount.

   Start the `podman-server` container.

Bind the `html-vol` volume as a read-only `/server` directory inside the container. The rest of the parameters remain the same.

```
[student@workstation ~]$ podman run -ti --rm --name podman-server -p 8000:8000 \
   --mount 'type=volume,source=html-vol,destination=/server,ro' \
   registry.ocp4.example.com:8443/redhattraining/podman-python-server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/)
```

In a web browser, access `http://localhost:8000`. You are presented with the `index.html` page.

Stop the container by pressing **Ctrl**+**c**.

Finish

On the `workstation` machine, use the `lab` command to complete this exercise. This is important to ensure that resources from previous exercises do not impact upcoming exercises.

```
[student@workstation ~]$ lab finish persisting-mounting
```