How can I change permission of mounted volumes in docker-compose.yml from the docker-compose.yml?

Asked 5 years, 5 months ago Modified 1 year, 7 months ago Viewed 174k times

74

```
version: '2'
services:
    web:
        build:
            context: ./
            dockerfile: deploy/web.docker
        volumes:
            - ./:/var/www
        ports:
            - "8080:80"
        links:
            - app
```

How can I change permission (`chmod`) /var/www automatically when `docker-compose up -d --build`?

docker docker-compose devops

Share Follow

edited May 31, 2021 at 2:03

Pang
9,633 ● 146 ● 81 ● 122

asked May 14, 2018 at 7:54

notalentgeek
5,059 ● 11 ● 34 ● 54

3 Answers

Sorted by: Highest score (default)

71

When bind-mounting a directory from the host in a container, files and directories maintain the permissions they have on the host. This is by design: when using a bind-mount, you're giving the container access to existing files from the host, and Docker won't make modifications to those files; doing so would be very dangerous (for example, bind-mounting your home-directory would change file permissions of your host's home directory, possibly leading to your machine no longer being usable).

To change permissions of those files, change their permissions on the host.

You can find more information on this in another answer I posted on StackOverflow: https://stackoverflow.com/a/29251160/1811501

Share Follow

answered May 14, 2018 at 8:16

thaJeztah
27.9k ● 9 ● 73 ● 92

21 Dude... That's exactly what Docker is currently doing to me... It changes the permissions and the paths become accessible only by root or the docker container user... The rest of the apps cannot use the directory anymore. I am looking for a way to set the permissions so that I can set the permissions back to the ones I wanted to have before they get changed... This is incredibly annoying and breaks stuff! – Arturas M Nov 23, 2020 at 19:05

How about if a `VOLUME` was created w/ `docker volume create --driver local ...`. Will it not be bind-mounted as well and if so, does one have control over file modes on the level of tools such as `docker volume` (as opposed to `docker volume inspect ... && chmod ...`)? – rookie099 May 12 at 14:32

@ArturasM docker itself does not change the permissions, but the container runs as `root` and modifies file-permissions (or creates files), those files may be owned by `root`. If those files are in a bind-mounted directory, then those files are modified. Run your container as a non-root user (which can be the UID/GID matching your user-account), or you can run the Docker Engine in "rootless mode" and set it up to re-map the `root` user to your account. docs.docker.com/go/rootless. Alternatively, use Docker Desktop, which converts permission (but comes with a performance overhead) – thaJeztah Jun 3 at 13:21

@rookie099 a `VOLUME` is effectively also a bind-mount, but the path that's mounted from is (by default) in a dedicated location on the daemon's host (inside `/var/lib/docker`); those files should not be accessed directly from other tools. You can create a volume with a custom "source" location, but the situation is mostly the same (permissions in container === permissions "on the host"). – thaJeztah Jun 3 at 13:23

1 I should add that recent versions of the Linux kernel added support for idmapped mounts, which could improve this in future (but it's not supported by all kernels, and support for it has not yet been implemented in all components used to run containers). – thaJeztah Jun 3 at 13:24

19

you can add the permissions after an extra column like:

```
    volumes:
            - ./:/var/www:ro #read only
```

Share Follow

edited Mar 29, 2022 at 19:21

SuperPoney
605 ● 7 ● 22

answered May 14, 2018 at 8:02

Edwin
2,176 ● 20 ● 26

3 Let say I want to have 777 or 750 like permission what would `ro` be? – notalentgeek May 14, 2018 at 8:08

2 check this out en.wikipedia.org/wiki/Chmod, especially the @Numerical permissions – Edwin May 14, 2018 at 8:09

4 wait a second, you want to change the permissions of the files and folders of the mounted volume? I miss-read your question then, because that you can do it only on the host – Edwin May 14, 2018 at 8:20

1 Just remember to delete the "//read only" part or you will get confusing errors about "invalid mode" when building – Vidde Jul 8, 2021 at 8:42

1 This will mount a read-only filesystem, which isn't quite the same thing as changing permissions. It doesn't impact the flags set by chmod – CantankerousBullMoose May 12, 2022 at 17:48

write in shared volumes docker

Asked 8 years, 7 months ago Modified 5 months ago Viewed 155k times

85

I have a docker with a php application on it

I have a share volume, for example

```
/home/me/dev/site <=> /var/www/site
```

I can write something in my host, it will be sync with the container

if I launch

```
sudo docker exec test touch /var/www/site/test.txt
```

It works

But if my server is trying to create a file as `www-data` this is not working because of the rights.

Is there a way to give access to my shared volumes to `www-data` ?

I am using boot2docker

docker

Share Follow

edited Mar 25, 2015 at 0:51

asked Mar 25, 2015 at 0:05

Ajouve

9,825 • 27 • 91 • 139

Does that user need write permission to the volume, or does it need the ability to execute docker (as a wrapper for managing the volume)? – scrowler Mar 25, 2015 at 0:09

the user needs to create files in the volume, not just execute files from the volume – Ajouve Mar 25, 2015 at 0:13

And does that creation and execution happen through the shell or do you have to use the `docker exec` command to do that? – scrowler Mar 25, 2015 at 0:18

No, this is a website, it can create a file when I load a page for example – Ajouve Mar 25, 2015 at 0:20

So what are the current permissions (i.e. user and group) for that folder you write to? If it's root you will probably need to `chown` it to a generic user, or create a user group that can contain both the generic/root user and the www-user, and assign that group as the owner of the folder – scrowler Mar 25, 2015 at 0:23

3 Answers

Sorted by: Highest score (default)

135

(Bind-mounted) volumes in Docker will maintain the permissions that are set on the Docker host itself. You can use this to set the permissions on those files and directories before using them in the container.

Some background;

Permissions in Linux are based on user and group *ids* (`'uid'` / `'gid'`). Even though you *see* a user- and group *name* as owner, those names aren't actually important in Linux, they are only there to make it easier for you to see who's the owner of a file (they are looked up from the `/etc/passwd` file).

You can set *any* `uid` / `gid` on a file; a user doesn't have to *exist* when setting those permissions. For example;

```
touch foobar && sudo chown 1234:5678 foobar && ls -la foobar

# UID and GID are set to 1234 / 5678, even though there's no such user
-rw-rw-r-- 1 1234 5678 0 Mar 25 19:14 foobar
```

Checking permissions (inside and outside a container)

As mentioned above, Docker maintains ownership of the host when using a volume. This example shows that permissions and ownership in the volume are the same *outside* and *inside* a container;

```
# (First create a dummy site)

mkdir -p volume-example/site && cd volume-example
echo "<html><body>hello world</body></html>" > site/index.html

# Permissions on the Docker host;
```

```
ls -n site

# total 4
# -rw-rw-r-- 1 1002 1002 38 Mar 25 19:15 index.html

# And, permissions inside a nginx container, using it as volume;

sudo docker run --rm -v $(pwd)/site:/var/www nginx ls -n /var/www

# total 4
# -rw-rw-r-- 1 1002 1002 38 Mar 25 19:15 index.html
```

Setting the permissions

As explained, a user doesn't have to exist in order to use them, so even if we don't have a `www-data` user on the Docker host, we can still set the correct permissions if we know the "uid" and "gid" of that user inside the container;

Let's see what the uid and gid of the www-data user is *inside* the container;

```
sudo docker run --rm nginx id www-data

# uid=33(www-data) gid=33(www-data) groups=33(www-data)
```

First check the state *before* changing the permissions. This time we run the nginx container as user `www-data`;

```
sudo docker run \
  --rm \
  --volume $(pwd)/site:/var/www \
  --user www-data nginx touch /var/www/can-i-write.txt

# touch: cannot touch `/var/www/can-i-write.txt': Permission denied
```

Next, set the permissions on the local directory, and see if we are able to write;

```
sudo chown -R 33:33 site

sudo docker run \
   --rm \
   --volume $(pwd)/site:/var/www \
   --user www-data nginx touch /var/www/can-i-write.txt
```

Success!

Share Follow

edited Dec 8, 2017 at 18:18

Buurman
1,914 ● 17 ● 26

answered Mar 25, 2015 at 8:54

thaJeztah
27.9k ● 9 ● 73 ● 92

4 Thanks, you saved my day, It works well on linux. But but it doesn't work on Osx with docker-machine. Any suggestion? – Hemerson Varela Mar 3, 2016 at 21:20

16 That's a great explanation, thanks. But what if we want to use different UID/GID on a host machine and inside a container? For example `33` could be a legitimate user inside a container, but could be an "evil person" on a host machine. Also, I want for all files to belong to my user account on a host machine and not to some mysterious user `33`. – Slava Fomin II Feb 11, 2019 at 22:25

1 What if the uid of the container clashes with the uid of an existing user in the host? You'll end up giving them access to it. Also, you're setting some dangling uids in your host system that only make sense for the container. For these reasons I think it's better to do it the other way around: setting the uid of the container to match the one of the host or make use of the "docker" group on linux – disklosr May 14, 2019 at 13:49

2 > I think it's better to do it the other way around: setting the uid of the container to match the one of the host or make use of the "docker" group on linux The `docker` group should not be used as it provides access to the docker socket (API access). As to matching the container to run as the current user; The scenario described here is most useful for development situations (working on files outside of the container, but giving the container access), however, – thaJeztah May 22, 2019 at 13:51

2 making the container use a different than it is designed to run on will make your development situation differ more from the production situation (you should not use bind-mounts in production), which may cause other headaches down the road. – thaJeztah May 22, 2019 at 13:51

49

Add the following lines to your dockerfile and rebuild your image

```
RUN usermod -u 1000 www-data
RUN usermod -G staff www-data
```

Share Follow

edited Mar 20, 2016 at 19:23

Tunaki
133k ● 46 ● 347 ● 426

answered Mar 20, 2016 at 19:20

István Döbrentei
948 ● 10 ● 20

4 Perfect! That is what I've been looking for. linux.die.net/man/8/usermod docs for this command. Basically www-data's user id is changed to 1000 and it's added to staff group. Why 1000 or staff? Run `docker exec -it CONTAINER_NAME_OR_ID bash` Once in run `ls -la /var/www` you will see ex. `drwxr-xr-x  1 1000 staff  238 Aug 20 20:14 html` – Krzysztof Boduch Aug 20, 2016 at 20:23

1 This works for me on my Ubuntu desktop because my uid is 1000. If you're not the first user account (which gets 1000 by default) you'll need to replace 1000 with your uid. – Ethan Hohensee Apr 8, 2018 at 14:03