Title: Adding permissions to host directory with docker-compose

Post Body:

I'm trying to set up a development environment using docker-compose and my container does not seem to have permissions to the host directory that is mounted to the container, i'm getting this error when running a grunt task that tries to modify folders inside the volume:

```
app_1                   | Warning: Unable to delete '.tmp' file (EACCES, permission denied '.tmp'). Use --force to continue.
```

here's my docker file:

```
FROM node:0.10  RUN mkdir -p /usr/src/app WORKDIR /usr/src/app  RUN apt-get update \     && apt-get install -y --no-install-re
```

and my docker-compose file:

```
app:   build: .   dockerfile: Dockerfile.dev   ports:    - '9000:9000'   env_file:    - ./server/config/env/development.env
```

- I'm running ubuntu 15.10 with docker-compose version 1.5.2, build 7240ff3
- Note that I am using the :Z permission

Accepted Answer: None

Highest Rated Answer:

It's just a file permissions thing.

The first thing to realise is that the volume you are mounting has different permissions to the folder you create and chown in the Dockerfile. The node user presumably doesn't have permissions to access this folder. You can fix this by running something like:

```
$ docker run -u root -v $(pwd):/usr/src/app:Z my_app_image chown -R node:node /usr/src/app
```

This will change the permissions of the folder on the host.

Alternatively, if you need to be root to run the `npm install && bower install`, you could leave the root user as the default user then change to the node user to run the application. Something like:

```
npm install && bower install && gosu node npm start
```

Here I've used the gosu tool, which you will need to install in the image. It's a little nicer than sudo, as it doesn't start a second process.