
gerassimos update

d499b69 on May 12

2 contributors 

145 lines (118 sloc) 5.25 KB

Section 5 - Containers lifecycle

2 - Lab: Manage Multiple Containers

Objective:

- To startup a typical three container application with nginx, mysql, and apache.
- To learn how to get help from both the cli command `--help` and the on line official Docker documentation.

Key points:

- Get help form cli (--help) and from on line docs.docker.com
- Start a three-containers application composed from *nginx*, *mysql* and a *httpd*.
- Run all of them --detach (or -d), name them with --name

- *nginx* should listen on 80:80, *httpd* on 8080:80, *mysql* on 3306:3306
- When running mysql, use the --env option (or -e) to pass in MYSQL_RANDOM_ROOT_PASSWORD=yes
- Use docker container logs on mysql to find the random password created on startup
- Clean up all with "docker container stop" and "docker container rm" (both can accept multiple names or ID's)
- Use docker container ls to ensure everything is correct before and after cleanup

Solution

Step 1

Find how to get useful information about the options that can be used with the "docker container run" command. In particular, find how to set an environment variable in the container. Use both:
a) the offline documentation available from the docker CLI (--help)
b) the official online documentation (docs.docker.com).

Step 1 case a (on-line help)

```
# docker container run --help
Usage:  docker container run [OPTIONS] IMAGE [COMMAND] [ARG...]

Run a command in a new container

Options:
...
-e, --env list                       Set environment variables
      --env-file list                  Read in a file of environment variables
...

---
```

Step 1 case b (on-line help)

Ref:

- https://docs.docker.com/engine/reference/run/
- https://docs.docker.com/engine/reference/run/#env-environment-variables

From the online documentation we can see that we can set any environment variable in the container by using one or more -e flags. If the operator names an environment variable without specifying a value, then the current value of the named variable is propagated into the container's environment. Example:

```
$ export today=Wednesday
$ docker run -e "deep=purple" -e today --rm alpine env
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
HOSTNAME=d2219b854598
deep=purple
today=Wednesday
HOME=/root
```

See also:
https://hub.docker.com/_/mysql

Step2

Start a three-containers application composed from *nginx*, *mysql* and a *httpd*. Note that there is no need to set up these three different containers with application data. Start all containers in detached mode (-d).

Set the following port mapping and custom names:

container	host port	container port	name
nginx	80	80	proxy
httpd	8080	80	webserver
mysql	3306	3306	db

Use the "--env" option to the environment variable MYSQL_RANDOM_ROOT_PASSWORD=yes in the mysql container. Do not worry, we will go through the (--env) option in next lectures.

```
# docker container run -d --name proxy -p 80:80 nginx
# docker container run -d --name webserver -p 8080:80 httpd
# docker container run -d -p 3306:3306 --name db -e MYSQL_RANDOM_ROOT_PASSWORD=ye
```

Q: why do we set different host ports for each container?
A: We need to expose different ports for each container because we cannot have 2 processes on the same host opening the same TCP port.

Q: What is the MYSQL_RANDOM_ROOT_PASSWORD environment variable actually doing?
A: The mysql image has an option for setting a random root password on bootup
Ref:
https://hub.docker.com/_/mysql

> `MYSQL_RANDOM_ROOT_PASSWORD`
> This is an optional variable. Set yes to generate a random initial password for the root user (using pwgen). The generated root password will be printed to stdout.

Step3

Verify that the containers are running:

```
# docker ps
CONTAINER ID   IMAGE   COMMAND                  CREATED         STATUS         PC
3a4a2960ec08   mysql   "docker-entrypoint.s…"   2 minutes ago   Up 2 minutes   0.
18beccb07c1f   httpd   "httpd-foreground"       7 minutes ago   Up 7 minutes   0.
4db904668f02   nginx   "nginx -g 'daemon of…"   8 minutes ago   Up 8 minutes   0.
```

Step4

Search the mysql password automatically generated from the log of the container.

```
# docker container logs db
...
GENERATED ROOT PASSWORD: ahChooXuR3ahpeew3eg5jou2Ae9aiXie
...
```

The following command uses also the `grep` to filter out the search result.
Note also that we used the "2>&1" to redirect the STDERROR to the STDOUT.

```
# docker container logs db 2>&1 | grep -i "password"
GENERATED ROOT PASSWORD: ahChooXuR3ahpeew3eg5jou2Ae9aiXie
```

Step5

Stop and remove all the containers created for this exercise.

```
# docker rm -f db webserver proxy
db
```

```
webserver
proxy
```

Step6

Verify that there is not any container running or stopped.

```
$ docker ps -a
CONTAINER ID   IMAGE   COMMAND   CREATED   STATUS   PORTS   NAMES
```