Running Containers and Troubleshooting

Troubleshooting

5/20/2022

Contents

How to troubleshoot docker containers

Troubleshooting Docker Containers

Note: At the end of this topic, you will be provided with a terminal to an environment that has all the prerequisites (such and Kubernetes) up and running. You can practice your commands in this tutorial without any need to setup your own en

1. Get environment settings.

   ```
   docker run --rm ubuntu env
   ```

2. Kill running containers.

   ```
   docker kill $(docker ps -q)
   ```

3. Delete all containers (force!! running or stopped containers).

   ```
   docker rm -f $(docker ps -qa)
   ```

4. Delete old containers.

   ```
   docker ps -a | grep 'weeks ago' | awk '{print $1}' | xargs docker rm
   ```

5. Delete stopped containers.

   ```
   docker rm -v $(docker ps -a -q -f status=exited)
   ```

6. Delete containers after stopping.

   ```
   docker stop $(docker ps -aq) && docker rm -v $(docker ps -aq)
   ```

7. Delete dangling images.

   ```
   docker rmi $(docker images -q -f dangling=true)
   ```

8. Delete all images.

   ```
   docker rmi $(docker images -q)
   ```

9. Delete dangling volumes. As of Docker 1.9.0:

   ```
   docker volume rm $(docker volume ls -q -f dangling=true)
   ```

   In 1.9.0, the filter dangling=false does not work. It is ignored and lists all volumes.

10. Show image dependencies.

```
docker images -viz | dot -Tpng -o docker.png
```

11. Slim down Docker containers.
    Cleaning APT in a RUN layer: This must be done in the same layer as that of the other APT commands. If not, the previous layers will still contain the original information and your images will still be large.

    ```
    RUN {apt commands} \

     && apt-get clean \

     && rm -rf /var/lib/apt/lists/* /tmp/* /var/tmp/*
    ```

    Flatten an image:

    ```
    ID=$(docker run -d image-name /bin/bash)

    docker export $ID | docker import – flat-image-name
    ```

    For backup:

    ```
    ID=$(docker run -d image-name /bin/bash)

    (docker export $ID | gzip -c > image.tgz)

    gzip -dc image.tgz | docker import - flat-image-name
    ```

Information on Running Docker Containers

- `docker ps` displays running containers.
- `docker logs` gets logs from the container. (You can use a custom log driver, but logs are available only for json-file and journald in 1.10).
- `docker inspect` inspects all the information of a container (including the IP address).
- `docker events` gets events from the container.
- `docker port` displays the public facing port of the container.
- `docker top` displays the running processes in container.
- `docker stats` displays the containers' resource usage statistics.
- `docker diff` displays the changed files in the container's FS.
- `docker ps -a` displays running and stopped containers.
- `docker stats --all` displays a running list of containers.
- `docker update` updates a container's resource limits.

To check the CPU, memory, and network I/O usage of a single container:

```
docker stats
```

For all containers listed by ID:

```
docker stats $(docker ps -q)
```

For all containers listed by name:

```
docker stats $(docker ps --format '{{.Names}}')
```

For all containers listed by image:

```
docker ps -a -f ancestor=ubuntu
```

To remove all untagged images:

```
docker rmi $(docker images | grep “^” | awk '{split($0,a," "); print a[3]}')
```

To remove container by a regular expression:

```
docker ps -a | grep wildfly | awk '{print $1}' | xargs docker rm -f
```

To remove all exited containers:

```
docker rm -f $(docker ps -a | grep Exit | awk '{ print $1 }')
```

You can practice the above-mentioned commands using the following widget: