

Packaging Network Automations with Docker

Greg Anuzelli

Senior Solutions Architect, Presidio

@ganuzelli





Greg Anuzelli

Senior Solutions Architect, Presidio

Greg has over 25 years of IT experience in networking, information security, collaboration, contact center, virtualization, cloud, and software development. As an evangelist of automation, Greg has delivered numerous talks on coding, NetDevOps, and software-defined everything.



What Problems Are We Trying to Solve?

- Packaging
- Ensuring code dependencies, libraries, etc. are met (and that newer versions don't break something)
- Versioning
 - E.g. Python 2 vs Python 3, features being added and deprecated in Ansible, etc.
- OS compatibility
 - Tools like Ansible and Cisco Network Services Orchestrator do not run on Windows

What Is a Container?

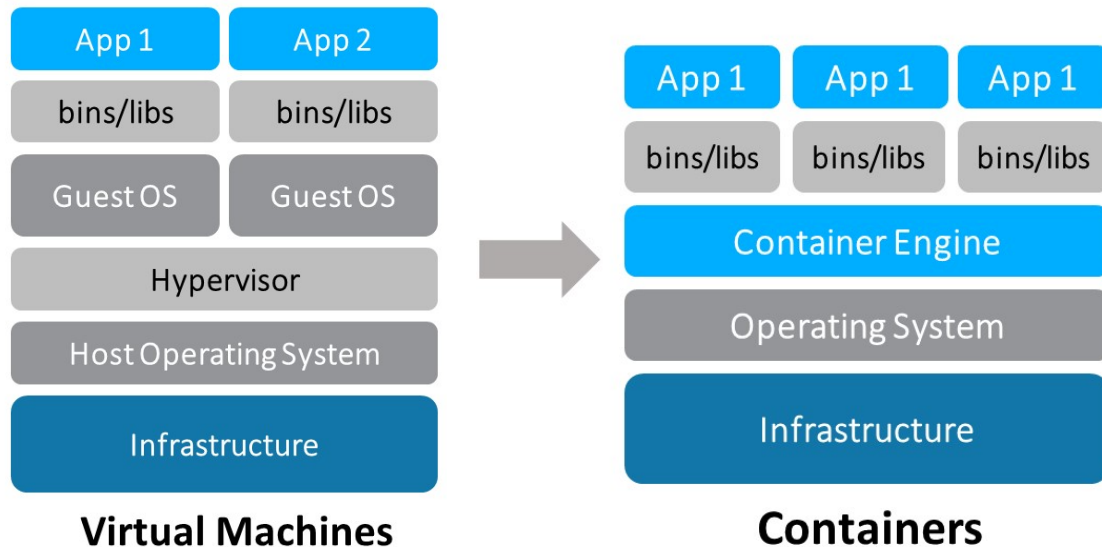


Image Source and Credits: <https://rancher.com/playing-catch-docker-containers>

Really, What Is a Container?

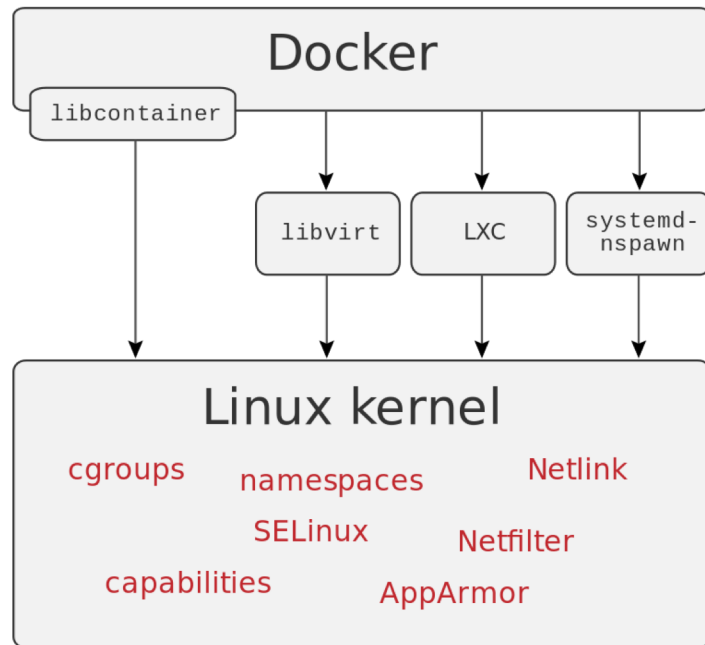


Image Source and Credits: <https://delftswa.github.io/chapters/docker/>

Docker on Windows and macOS

- Docker Desktop for Windows
 - Windows 10 64bit: Pro, Enterprise or Education
 - Linux VM runs on Hyper-V
- Docker Desktop for Mac
 - macOS Sierra 10.12 and newer
 - Linux VM runs on HyperKit
- Docker Toolbox for Windows / Mac
 - Linux VM runs on Oracle VirtualBox

Hello World

`docker run hello-world`

```
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
1b930d010525: Pull complete
Digest: sha256:2557e3c07ed1e38f26e389462d03ed943586f744621577a99efb77324b0fe535
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.
```

- What just happened?
 - Downloaded (pulled) the "hello-world:latest" image from Docker Hub
 - Created a new container from that image
 - Output from the container streamed to your terminal

Try Something More Ambitious

```
docker run -it ubuntu bash
```

```
Unable to find image 'ubuntu:latest' locally
latest: Pulling from library/ubuntu
Digest: sha256:7a47ccc3bbe8a451b500d2b53104868b46d60ee8f5b35a24b41a86077c650210
Status: Downloaded newer image for ubuntu:latest
root@95afdbfda5c1:/#
```

- What just happened?
 - Downloaded (pulled) the "ubuntu:latest" image
 - Created a new container from that image
 - "-it" means "Interactive" and "tty"
- Type "exit" to exit bash, which causes the container to stop

Common Docker Commands

Show all containers, including stopped:

```
docker ps -a
```

Remove a stopped container:

```
docker rm <container id>
```

Show downloaded images:

```
docker images
```

Remove a downloaded image:

```
docker rmi <image>
```

Dockerfile

```
FROM ubuntu:bionic
```

```
RUN DEBIAN_FRONTEND=noninteractive apt-get -y update \  
    && DEBIAN_FRONTEND=noninteractive apt-get -y install \  
fortune-mod fortunes cowsay
```

```
ENTRYPOINT /usr/games/fortune computers | \  
/usr/games/cowsay
```

Build an Image From a Dockerfile

```
$ docker build -t quotes .
```

```
$ docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
quotes	latest	40fa...	3 minutes ago	159MB

```

root@7eab654acf36: /usr/share/games/fortunes
$ docker run quotes
-----
/ Real programmers don't comment their \
| code. It was hard to write, it should |
\ be hard to understand.                /
-----
\      ^__^
 \    (oo)\_______
      (__)\       )\/\
         ||----w |
         ||     ||
$ █

```

Other Common Dockerfile Instructions

A comment

ENTRYPOINT *["executable", "param1", "param2"]*

Preferred form for ENTRYPOINT

COPY *<src> <dst>*

Copies files from the host to the container image

WORKDIR *<dir>*

Set the current working directory within the image

Layers

- Certain Dockerfile instructions (e.g. RUN and COPY) result in the creation of a layer
- Use of Layers can speed builds/rebuilds and decrease the amount of data downloaded by docker pull
- But can also make the total image size bigger
- Consolidate your RUN / COPY lines to save space
- Separate RUN / COPY lines to build layers where beneficial

Volumes

- Map directories in the container to directories on your host
- Dockerfile syntax:
`VOLUME /myvol`
- Use to expose any database storage area, configuration storage, or files/folders created by your docker container
- Do not store persistent data in a container

Docker-compose.yml

```
version: "3.5"
services:
  my-automation:
    image: my-automation
    build:
      context: .
      dockerfile: Dockerfile
    volumes:
      - ./data
    entrypoint: sh -c "python3 -u test.py"
```

Docker Compose Commands

`docker-compose build`

Builds the Docker image specified in the "build" section

`docker-compose run [--rm] my-automation`

Starts the container, and executes the "entrypoint" instruction (-rm means delete the container when done)

`docker-compose ps [-a]`

Shows running containers (-a means show stopped containers too)

`docker-compose rm`

Remove stopped containers

Docker Image Options

- Ubuntu, Centos, etc
 - Full featured
 - Ubuntu 18.04 and later images are now "slim"
- Alpine
 - Small footprint, small security surface area
- Official "purpose built" images from Docker Hub, e.g. "python", "ansible"

Exposing Network Ports

Expose TCP/UDP ports from your container with "-p"

```
docker run -p 8080:80 nginx
```

TCP port 8080 on your host maps to port 80 in the container

Network Ports in docker-compose.yml

ports:

- "8000:8000"
- "5000-5010:5000-5010"
- "8080:80"
- "127.0.0.1:10022:10022"
- "162:162/udp"

Docker Networking

Your laptop	-> Docker Linux VM	-> Container
10.1.1.1	-> 192.168.65.3	-> 172.17.0.2

Outbound traffic: NAT to host IP

Inbound traffic: SNAT through container gateway (e.g. 172.17.0.1)



Resolving Network Address Conflicts

- Docker Linux VM network
 - Change through Docker Desktop GUI



Resolving Network Address Conflicts

- Container IP network
 - E.g. 172.17.0.0/24
- Create network in docker compose file, and reference it from your service
- Carrier-grade NAT IP space is a good option (100.64.0.0/10)

```
services:
  my-automation:
    ...
    networks:
      - my-network
networks:
  my-network:
    name: my-network
    driver: bridge
    ipam:
      config:
        - subnet: 100.64.99.0/24
```

Container Distribution

- A container repository supports "pulls" (e.g. `docker-compose pull`)
- Docker Hub
- GitLab
- Other 3rd party solutions
- Roll your own: docs.docker.com/registry/deploying



Lab

Lab Overview

- Build a container with Cisco Network Services Orchestrator, and use it to simulate an IOS device
- Build a container with a Python network automation which creates a Loopback interface on the simulated IOS device



DEVNET Create