

Docker Containers

CMIS 545 - Cloud Computing Architecture

Rafael Marino, Eduardo Cassinelli

McGill University

November, 2020

- 1 Just Enough Microservices
- 2 Containerization
- 3 Containers vs Virtual Machines
- 4 Docker
- 5 Docker Primitives
- 6 Docker Demo

Section 1

Just Enough Microservices

What are Microservices?

- Microservices are

How do Microservices relate to Containers?

- Containers facilitate the modularized development and deployment of microservices.
- Using one service per container guarantees independence

Section 2

Containerization

Section 3

Containers vs Virtual Machines

Containers vs Virtual Machines

Random explanation xxx

Differences

- Virtualization
- Containerization
- third bullet
- fourth bullet

Section 4

Docker

Definition

Docker is a Platform xxx

Section 5

Docker Primitives

Just Enough Microservices

○○○

Containerization

○

Containers vs Virtual Machines

○○○

Docker

○○

Docker Primitives

●○○○

Docker Demo

○○○○○○○○

Docker Engine

Docker Image

- We can think of a Docker Image as a stopped Docker Container
- Each element within an image represents an image layer. Layers are then stacked on top of each other and ready to run.
-

Docker Container

- Containers are the central unit on top of which all Docker is built, and they are better examined practically.

Docker Compose

Section 6

Docker Demo

Graylog App

- Graylog is an open source log management solution for capturing, storing, and analyzing machine data. It needs two dependencies:
 - MongoDB: An open-source, “general purpose, document-based, distributed database”
 - Elasticsearch: An open-source, “powerful analytics engine to explore data easily”.

Docker Run

~\$ docker container run <options> <image>:<tag> <app>

Graylog Setup

```
eduardo@eduardo-L380:~$ docker container run --name mongo -d mongo:3
```

```
eduardo@eduardo-L380:~$ docker run --name elasticsearch \  
-e "http.host=0.0.0.0" \  
-e "ES_JAVA_OPTS=-Xms512m -Xmx512m" \  
-d docker.elastic.co/elasticsearch/elasticsearch-oss:6.8.10
```

```
eduardo@eduardo-L380:~$ docker run --name graylog --link mongo --link  
elasticsearch \  
-p 9000:9000 -p 12201:12201 -p 1514:1514 -p 5555:5555\  
-e GRAYLOG_HTTP_EXTERNAL_URI="http://127.0.0.1:9000/" \  
-d graylog/graylog:3.3
```

Figure 1: Graylog Setup Commands

Note

In Ubuntu 20.04 LTS stock, installing graylog requires adjusting default virtual memory settings using: *sudo sysctl -w vm.max_map_count=262144*

Testing Graylog

```
eduardo@eduardo-L380:~$ docker container ls
```

CONTAINER ID	IMAGE	CREATED	STATUS	NAMES
1de8801d699f	graylog/graylog:3.3	4 seconds ago	Up 3 seconds (health: starting)	graylog
76b0e75fb7be	./elasticsearch/	22 seconds ago	Up 21 seconds	elasticsearch
229f3aebfe56	mongo:3	27 seconds ago	Up 26 seconds	mongo

The screenshot shows the Graylog web interface. At the top, there's a navigation bar with 'graylog' logo and tabs for Search, Streams, Alerts, Dashboards, Enterprise, and System. Below the navigation bar, there's a search bar with the query 'q12_source_input:5fb2ab1a4fe9345d716b1072'. To the right of the search bar, there's a 'Not updating' button. Below the search bar, there's a 'Message Count' chart showing a single bar at a value of 1. Below the chart, there's an 'All Messages' section showing a single message with the timestamp '2020-11-16 16:39:19 +00:00' and the source '172.17.0.1'. The message content is 'Testing log message for CMIS545 Cloud Computing Architecture'.

Further Testing

```
eduardo@eduardo-L380:~$ echo 'Testing log message for CMIS545 Cloud  
Computing Architecture' | nc localhost 5555
```

All Messages



timestamp

source

2020-11-16 16:39:19 +00:00

172.17.0.1

Testing log message for CMIS545 Cloud Computing Architecture

✉ 46318500-282a-11eb-a630-0242ac11000

[Permalink](#)[Copy ID](#)[Show surrounding messages ▾](#)[Test against stream ▾](#)

Timestamp

2020-11-16 16:39:19.560

Received by

Text Input on d49f27d0 / 8759155980c7

Stored in Index

graylog_0

Routed into streams

- All messages

message

Testing log message for CMIS545 Cloud Computing Architecture

source

172.17.0.1

timestamp

2020-11-16 16:39:19 +00:00

Killing the Application // Stopping and Removing Containers??

```
eduardo@eduardo-L380:~$ docker container stop mongo
```

```
eduardo@eduardo-L380:~$ docker container ls -a
```

CONTAINER ID	IMAGE	CREATED	STATUS	NAMES
1de8801d699f	graylog/graylog:3.3	2 minutes ago	Up 2 minutes (healthy)	graylog
76b0e75fb7be	./elasticsearch/	53 seconds ago	Up 52 seconds	elasticsearch
229f3aebfe56	mongo:3	2 minutes ago	Exited(0) 5 seconds ago	mongo

Figure 2: Stopping Single Container

```
eduardo@eduardo-L380:~$ docker container stop mongo elasticsearch graylog
```

```
eduardo@eduardo-L380:~$ docker container rm mongo elasticsearch graylog
```

Figure 3: Stopping and Removing all Containers

Further Container Commands[1]

Command	Description
<i>docker container prune</i>	Remove all stopped containers
<i>docker container start</i>	Start one or more stopped containers
<i>docker container diff</i>	Inspect file or directory changes
<i>docker container exec</i>	Run a command in a running container
<i>docker container export</i>	Export a container's filesystem as a tar
<i>docker container inspect</i>	Display detailed information
<i>docker container kill</i>	Kill one or more running containers
<i>docker container logs</i>	Fetch the logs of a container

[1]

Documentation: https://docs.docker.com/engine/reference/commandline/container_run/