Cloud computing

Compose: microservices app stack

Aleix Llusà Serra

Master's degree in Machine Learning and Cybersecurity for Internet Connected Systems
Universitat Politècnica de Catalunya
https://epsem.upc.edu

22 de setembre de 2026

App stack: docker compose

Container ⟶ Service ⟶ Project/Stack

Abstract objective: Managing and provisioning computer data centers through machine-readable definition files, rather than physical hardware configuration or interactive configuration tools. The definitions may be in a version control system. Towards the culture of DevOps and Agile software development.

- Microservices
- Reproducibility
- Infrastructure as code
- Portability
- Automated environments

Compose: the basics

Let `mkdir app; app/web;` and `touch app/compose.yml`

app/compose.yml

```
services:
  db:
    image: mysql:latest
    environment:
      - MYSQL_ROOT_PASSWORD=XXX
  admindb:
    image: phpmyadmin:latest
    depends_on:
      - db
    ports:
      - "8081:80"
  web:
    image: my.registry.test/web:latest
    build:
      context: web
    ports:
      - "8080:80"
```

Compose: deploy

Deploy (foreground): `docker compose up`
Deploy (background): `docker compose up -d`
Deploy (select file): `docker compose -f compose.yml up`

Logs: `docker compose logs`
Running process: `docker compose ps`
Attach terminal: `docker compose exec web bash`

Stop: `docker compose stop`
Stop and remove: `docker compose down`

Compose: networks

By default: Creating network “app_default” with the default driver

Changing the default:

app/compose.yml

```
services:
  db:
    [...]
    networks:
       - backend
  admindb:
    [...]
    networks:
      - default
      - backend
  web:
    [...]
    depends_on:
       - admindb

networks:
  backend:
```

Compose: rollout

```
docker compose build
docker compose up -d
```

Better with versioning:

app/compose.yml

```
services:
  [...]
  web:
    image: my.registry.test/web:1
    build:
      context: web
    ports:
      - "8080:80"
```

```
docker compose up -d
```

Reverse proxy: typical web architecture

app/web/Dockerfile

```
FROM nginx:latest
[...]
COPY default.conf /etc/nginx/conf.d/
```

app/web/default.conf

```
server {
  server_name default_server;

  location /admin/ {
    proxy_pass http://admindb:80/;
  }
 root /usr/share/nginx/html;
}
```

Compose: scale

```
docker compose scale admindb=4
```

Remark!: A service with publishing ports cannot scale with compose

app/compose.yml

```
services:
  [...]
  admindb:
    image: phpmyadmin:latest
    depends_on:
      - db
    #NO PORTS
    networks:
      - default
      - backend
    environment:
      - PMA_ABSOLUTE_URI=http://localhost:8080/admin/
```

Compose: scale

Or scale permanently inside compose:

app/compose.yml

```
services:
  [...]
  admindb:
    image: phpmyadmin:latest
    depends_on:
      - db
    #NO PORTS
    networks:
      - default
      - backend
    deploy:
      replicas: 4
```

```
docker compose up -d
```

Compose: scaled rollouts or rollback

Use tags (versions) for images

app/compose.yml

```
services:
  [...]
  admindb:
    image: phpmyadmin:5.1
    depends_on:
      - db
    #NO PORTS
    networks:
      - default
      - backend
    deploy:
      replicas: 4
```

```
docker compose up -d
```

Question: do you think we can scale the database service?

Metrics

Hey, we still need hardware!

`docker stats`

Project/App name

- `docker compose -p APPNAME up -d`

`app/compose.yml`

```
name: APPNAME
services:
  [...]
```

- Default: directory that contains the compose file