Swarm deploy Yaml !

Πίνακας περιεχομένων

1. Create Swarm Service YAML

YAML

```
version: "3.4"

services:

# νομα υπηρεσίας
  master:
    image: registry.vlabs.uniwa.gr:5080/swarmlab-service-mpi2
    user: root
# ENTRYPOINT instruction allows you to configure a container that will run as an
executable.
    entrypoint: ["mpi_bootstrap", "role=master", "mpi_master_service_name=master",
"mpi_worker_service_name=worker"]
# Environment variables (declared with the ENV statement) can also be used in certain
instructions as variables to be interpreted by the Dockerfile.
# https://docs.docker.com/engine/reference/builder/#environment-replacement
    environment:
     - PASSWORD=padatest
     - PASSWORDVIEW=padatestview
     - SERVERROLE=master
     - SERVERWEB=no
# docker service inspect ondemand_mpi2_master
     - NODENAME={{.Node.Hostname}}
     - NODEID={{.Node.ID}}
     - SERVICEID={{.Service.ID}}
     - SERVICENAME={{.Service.Name}}
     - TASKID={{.Task.ID}}
     - TASKNAME={{.Task.Name}}
     - TASKREPID={{.Task.Slot}}
# Specify configuration related to the deployment and running of services.
    deploy:
# If the service is replicated (which is the default), specify the number of
containers that should be running at any given time.
# In global mode, running one replica of service per swarm node. The number of global
replicas is equal to the number of swarm nodes. In replica mode, you can run any
number of service instances.
```

```
      replicas: 9
      placement:
#        max_replicas_per_node: 1
        constraints:
          - node.role == worker
# Configures resource constraints.
#      resources:
#        limits:
#          cpus: '0.50'
#          memory: 500M
#        reservations:
#          cpus: '0.25'




#          memory: 200M
# Configures how the service should be rollbacked in case of a failing update.
#
#    parallelism: The number of containers to rollback at a time. If set to 0, all
containers rollback simultaneously.
#    delay: The time to wait between each container group's rollback (default 0s).
#    failure_action: What to do if a rollback fails. One of continue or pause (default
pause)
#    monitor: Duration after each task update to monitor for failure (ns|us|ms|s|m|h)
(default 0s).
#    max_failure_ratio: Failure rate to tolerate during a rollback (default 0).
#    order: Order of operations during rollbacks. One of stop-first (old task is
stopped before starting new one), or start-first (new task is started first, and the
running tasks briefly overlap) (default stop-first).
#
      restart_policy:
        condition: on-failure
        delay: 5s
        max_attempts: 5
        window: 120s

#Configures how the service should be updated. Useful for configuring rolling updates.
#
#    parallelism: The number of containers to update at a time.
#    delay: The time to wait between updating a group of containers.
#    failure_action: What to do if an update fails. One of continue, rollback, or
pause (default: pause).
#    monitor: Duration after each task update to monitor for failure (ns|us|ms|s|m|h)
(default 0s).
#    max_failure_ratio: Failure rate to tolerate during an update.
#    order: Order of operations during updates. One of stop-first (old task is stopped
before starting new one), or start-first (new task is started first, and the running
tasks briefly overlap) (default stop-first) Note: Only supported for v3.4 and higher.
#
#
```

```
      update_config:
        parallelism: 2
        delay: 10s
        order: stop-first
    networks:
      mpi2-net:
    volumes:
      - mpi3_vol:/var/share
    ports:
      - "55520:80"
      - "55521:8088"
      - "55522:6088"
      - "55523:6080"


  worker:
    image: registry.vlabs.uniwa.gr:5080/swarmlab-service-mpi2
    user: root
    entrypoint: ["mpi_bootstrap", "role=worker", "mpi_master_service_name=master",
"mpi_worker_service_name=worker"]
    environment:
     - SERVERROLE=worker
     - SERVERWEB=no
     - NODENAME={{.Node.Hostname}}
     - NODEID={{.Node.ID}}
     - SERVICEID={{.Service.ID}}
     - SERVICENAME={{.Service.Name}}
     - TASKID={{.Task.ID}}
     - TASKNAME={{.Task.Name}}
     - TASKREPID={{.Task.Slot}}
    deploy:
      replicas: 5
      placement:
#        max_replicas_per_node: 1
        constraints:
          - node.role == worker
          #- node.id == ${worker}
#      resources:
#        limits:
#          cpus: '0.50'
#          memory: 500M
#        reservations:
#          cpus: '0.25'
#          memory: 200M
      restart_policy:
        condition: on-failure
        delay: 5s
        max_attempts: 5
        window: 120s
      update_config:
        parallelism: 2
```

```
        delay: 10s
        order: stop-first
    networks:
      mpi2-net:
    volumes:
      - mpi3_vol:/var/share

  web:
    image: registry.vlabs.uniwa.gr:5080/swarmlab-service-sshfs
    user: root
    entrypoint: ["mpi_bootstrap", "role=worker", "mpi_master_service_name=master",
"mpi_worker_service_name=worker"]
    environment:
     - SERVERROLE=worker
     - SERVERWEB=yes
     - NODENAME={{.Node.Hostname}}
     - NODEID={{.Node.ID}}
     - SERVICEID={{.Service.ID}}
     - SERVICENAME={{.Service.Name}}
     - TASKID={{.Task.ID}}
     - TASKNAME={{.Task.Name}}
     - TASKREPID={{.Task.Slot}}
    deploy:
      replicas: 1
      placement:
        constraints:
          - node.role == worker
      resources:
        limits:
          cpus: '0.50'
          memory: 500M
        reservations:
          cpus: '0.25'
          memory: 200M
      restart_policy:
        condition: on-failure
        delay: 5s
        max_attempts: 5
        window: 120s
      update_config:
        parallelism: 2
        delay: 10s
        order: stop-first
    networks:
      mpi2-net:
    ports:
      - "55519:80"

#Creates a new network. The DRIVER accepts bridge or overlay which are the built-in
network drivers.
#Bridge networks are isolated networks on a single Engine installation. If you want to
```

```
create a network that spans multiple Docker hosts each running an Engine, you must
create an overlay network. Unlike bridge networks, overlay networks require some pre-
existing conditions before you can create one.
#
networks:
  mpi2-net:


#Mount host paths or named volumes
#Creates a new volume that containers can consume and store data in.
volumes:
  mpi3_vol:
    external: false
```