

Docker i Kubernetes w zarządzaniu projektem informatycznym

Ewa Namysł

promotor: dr hab. Serweryn Kowalski, prof. UŚ

Uniwersytet Śląski

22. listopada 2022

Spis treści

Maszyna fizyczna

Maszyna wirtualna

Kubernetes

Kolejny etap

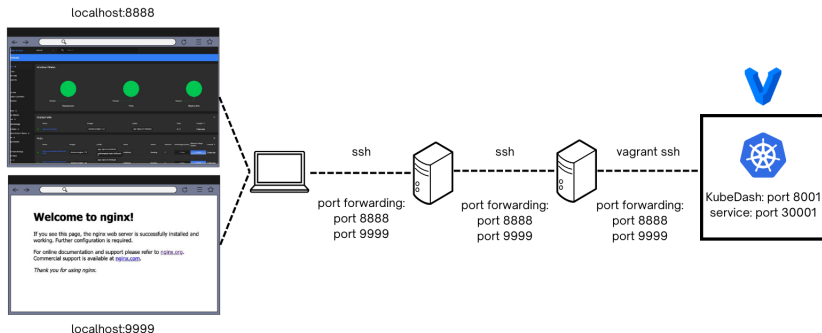
Postępy w pracy

Skonfigurowanie środowiska pracy na maszynie fizycznej

- System operacyjny: CentOS Stream 9
- Instalacja biblioteki libvirt - niezbędnej do wykorzystania Vagranta w wirtualizacji przy pomocy KVM/QEMU
- Instalacja pakietu Vagrant w celu tworzenia plików konfiguracyjnych dla maszyn wirtualnych
- Otwarcie i przekierowanie portów na maszynie fizycznej, aby umożliwić zdalny dostęp do serwisów

Postępy w pracy

Diagram



Postępy w pracy

Pierwsza wersja maszyny wirtualnej

- tworzenie podstawowego pliku Vagrantfile:

```
# -*- mode: ruby -*-
# vi: set ft=ruby :

Vagrant.configure("2") do |config| # version of Vagrant

  config.vm.provider "libvirt" do |vb|
    vb.memory = 2048
  end

  config.vm.network "forwarded_port", guest: 8001, host: 8888
  #8001 = KubeDash

  config.vm.box = "generic/ubuntu2004"

  config.vm.provision "docker"

  # data shared with host and VMs
  #
  # config.vm.synced_folder ".. /hostpath" "/VMpath"
  # config.vm.synced_folder ".. /baremetal_data", "/vagrant_data"

end
```

Postępy w pracy

- Dalsza automatyzacja procesu tworzenia VM:

```
# -*- mode: ruby -*-
# vi: set ft=ruby :

Vagrant.configure("2") do |config| # version of Vagrant
  config.vm.provider "libvirt" do |vb|
    vb.memory = 2048
    vb.cpus = 2
  end

  config.vm.box = "generic/ubuntu2004"

  config.vm.define "minikube-test"
  config.vm.hostname = "minikube-test"

  # 8001 = Kubernetes Dashboard
  config.vm.network "forwarded_port", guest: 8001, host: 8888

  # 30001 = minikube service
  config.vm.network "forwarded_port", guest: 30001, host: 9999

  $install_minikube = <<--SCRIPT
  echo Downloading and installing minikube
  curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube_linux-amd64.deb
  sudo dpkg -i minikube_latest_amd64.deb
  echo $(minikube version)
  echo Done, minikube is ready
  SCRIPT

  $install_kubectl = <<--SCRIPT
  echo Downloading and installing kubectl
  curl -LO "https://dl.k8s.io/release/${curl -L -s https://dl.k8s.io/release/stable.txt}/bin/linux/amd64/kubectl"
  sudo install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl
  echo $(kubectl version --client --output=yaml)
  echo Done, kubectl is ready
  SCRIPT

  config.vm.provision "docker"
  config.vm.provision "shell", inline: $install_minikube
  config.vm.provision "shell", inline: $install_kubectl

  config.vm.provision "shell", inline: "echo Cloning repository ; git clone https://github.com/namysl/yaml-minikube.git; echo Done, YAML files are ready"

  config.vm.provision "shell", inline: "echo VM READY"
end
```

Postępy w pracy

Uruchamianie maszyny wirtualnej

- włączenie maszyny poprzez komendę `vagrant up`
- sprawdzenie statusu
- łączenie się z maszyną przez SSH

```
[ewanamysl@h151 test3_auto]$ vagrant status
Current machine states:

minikube-test                running (libvirt)

The Libvirt domain is running. To stop this machine, you can run
`vagrant halt`. To destroy the machine, you can run `vagrant destroy`.
[ewanamysl@h151 test3_auto]$ vagrant ssh
Last login: Mon Nov 21 12:36:11 2022 from 192.168.121.1
vagrant@minikube-test:~$
```

Postępy w pracy

Pliki YAML

- Stworzenie plików YAML dla deploymentu i service'u
- Hosting na GitHubie, ściąganie plików w trakcie tworzenia VM

```
vagrant@minikube-test:~$ kubectl get svc
```

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
kubernetes	ClusterIP	10.96.0.1	<none>	443/TCP	2d20h
nginx-on-minikube	NodePort	10.98.82.101	<none>	80:30001/TCP	2d20h

YAML deployment

kubectl apply -f yaml-minikube/deployment.yaml

```
1  ---
2  apiVersion: apps/v1
3  kind: Deployment
4  metadata:
5    labels:
6      app: nginx-on-minikube
7      name: nginx-on-minikube
8  spec:
9    replicas: 3
10   selector:
11     matchLabels:
12       app: nginx-on-minikube
13   template:
14     metadata:
15       labels:
16         app: nginx-on-minikube
17     spec:
18       containers:
19         - name: nginx
20           image: "docker.io/nginx:1.23"
21           ports:
22             - containerPort: 80
23       strategy:
24         rollingUpdate:
25           maxSurge: 1
26           maxUnavailable: 1
27       type: RollingUpdate
```

YAML service

kubectl apply -f yaml-minikube/service.yaml

```
1  ---
2  apiVersion: v1
3  kind: Service
4  metadata:
5    labels:
6      app: nginx-on-minikube
7    name: nginx-on-minikube
8  spec:
9    ports:
10     - nodePort: 30001
11       port: 80
12       protocol: TCP
13       targetPort: 80
14    selector:
15      app: nginx-on-minikube
16    type: NodePort
```

Dostęp do Kubernetes Dashboard

The screenshot displays the Kubernetes Dashboard interface. The top navigation bar shows the 'Workloads' section selected. The left sidebar contains a list of resources: Workloads, Cron Jobs, Daemon Sets, Deployments, Jobs, Pods, Replica Sets, Replication Controllers, Stateful Sets, Service, Ingresses, Ingress Classes, Services, Config and Storage, Config Maps, Persistent Volume Claims, Secrets, Storage Classes, Cluster, Cluster Role Bindings, Cluster Roles, Events, and Namespaces.

The main content area is divided into three sections:

- Workload Status:** A summary view showing three green circles representing the status of Workloads. Below each circle, it indicates the number of running pods: Running: 1 for Deployments, Running: 3 for Pods, and Running: 1 for Replica Sets.
- Deployments:** A table listing the deployed applications.
- Pods:** A table listing the individual pods running in the cluster.

Name	Images	Labels	Pods	Created
nginx-on-minikube	docker.io/nginx:1.23	app: nginx-on-minikube	3 / 3	2 days ago

Name	Images	Labels	Node	Status	Restarts	CPU Usage (cores)	Memory Usage (bytes)	Created
nginx-on-minikube-56f8cc04b-67qf	docker.io/nginx:1.23	app: nginx-on-minikube pod-template-hash: 56f8cc04b	minikube	Running	0	0.00m	3.88MiB	2 days ago

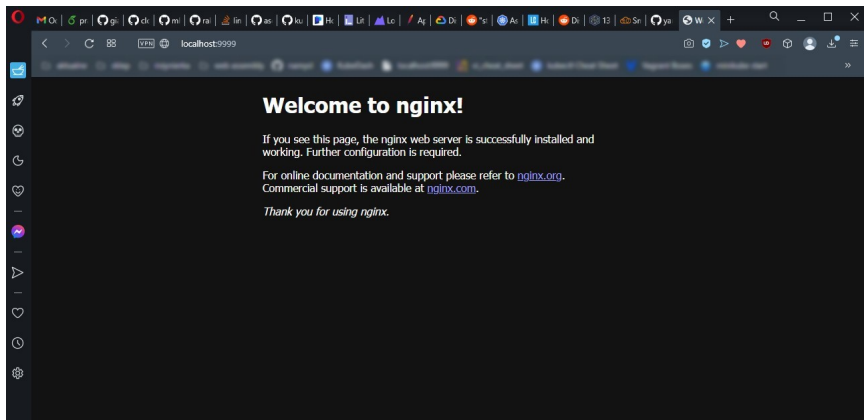
Lokalne testowanie aplikacji

```
vagrant@ubuntu2004:~$ curl http://192.168.49.2:30509
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
vagrant@ubuntu2004:~$
```

Zdalny dostęp



Postępy w pracy

Przetestowano:

- dostęp do podów i wprowadzanie w nich zmian,
- rolling updates, czyli update deploymentu bez downtime'u,
- niszczenie podów,
- zbieranie i interpretowanie logów

Kolejny etap

- Wprowadzenie horizontal pod autoscalera i limitu zasobów,
- Przeprowadzenie stress testów na minikube,
- Rozpoczęcie pracy nad plikiem Vagrantfile dla większej liczby węzłów (1 master, 2 workery)

Bibliografia

Kubernetes - oficjalna dokumentacja.

<https://kubernetes.io/docs/>

Dostęp: 2022-11-05

Vagrant - oficjalna dokumentacja.

<https://www.vagrantup.com/docs/>

Dostęp: 2022-11-05

E. Nemeth, G. Snyder, T. R. Hein, B. Whaley, D. Mackin.

Unix i Linux. Przewodnik administratora systemów.

Wydawnictwo HELION, 2018.