Bartosz Dorobek, Leszek Michoń

SWUS

Projekt - Orkiestracja K8S

Politechnika Warszawska

Wydział Elektroniki i Technik Informacyjnych

31 stycznia 2022

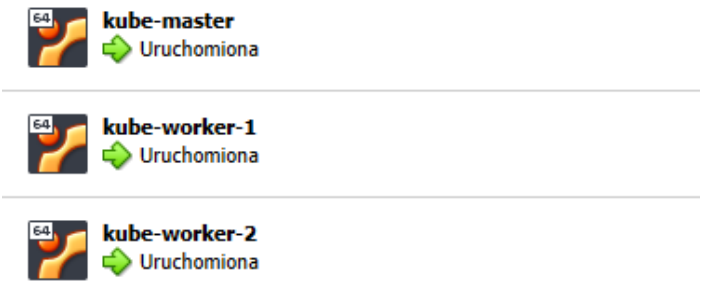
Spis treści:

1. Zakres i cel projektu

Wybrany przez nas projekt polegał na zapoznaniu się ze środowiskiem Kubernetes i z jego zastosowaniami w telekomunikacji. W celu ugruntowania wiedzy i zdobycia praktycznego doświadczenia z tym zagadnieniem projekt poza częścią teoretyczną, zawierał fazę praktyczną, która polegała na samodzielnym utworzeniu klastra Kubernetes składającego się z 3 węzłów (mastera i dwóch workerów) oraz orkiestracji kontenerów w klastrze.

2. Konfiguracja Środowiska i stworzenie klastra

Realizowany klaster składa się z trzech Node'ów. Dla każdego z nich utworzyliśmy oddzielną maszynę wirtualną z systemem Linux Ubuntu 20.04 (64-bit). Utworzone są lokalnie w VBox. Aby zapewnić prawidłowe działanie każda z VM ma przydzielone 2048 MB pamięci RAM, od 10 do 16 GB pamięci na dysku oraz 2 rdzenie procesora. Dla zapewnienia odpowiedniej komunikacji Node'ów z siecią i między sobą na każdej maszynie ustawione są dwie karty sieciowe: Mostkowana (bridget) oraz sieć NAT.



Kube-master:

```
docker0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        inet 172.17.0.1  netmask 255.255.0.0  broadcast 172.17.255.255
        ether 02:42:a2:af:4b:42  txqueuelen 0  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.0.114  netmask 255.255.255.0  broadcast 192.168.0.255
        inet6 fe80::a00:27ff:fe32:ff4e  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:32:ff:4e  txqueuelen 1000  (Ethernet)
        RX packets 249  bytes 22222 (22.2 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 2291  bytes 141022 (141.0 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

enp0s8: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.10.4  netmask 255.255.255.0  broadcast 192.168.10.255
        inet6 fe80::a00:27ff:fe64:1532  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:64:15:32  txqueuelen 1000  (Ethernet)
        RX packets 16593  bytes 1803474 (1.8 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 19843  bytes 10573313 (10.5 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 391390  bytes 84839659 (84.8 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 391390  bytes 84839659 (84.8 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

generic@kube-master:/etc/netplan$ _
```

Kube-worker-1:

```
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.0.115  netmask 255.255.255.0  broadcast 192.168.0.255
        inet6 fe80::a00:27ff:fe70:5d76  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:70:5d:76  txqueuelen 1000  (Ethernet)
        RX packets 4909  bytes 883900 (883.9 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 2108  bytes 310629 (310.6 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

enp0s8: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.10.5  netmask 255.255.255.0  broadcast 192.168.10.255
        inet6 fe80::a00:27ff:fe5d:5d26  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:5d:5d:26  txqueuelen 1000  (Ethernet)
        RX packets 187743  bytes 242542245 (242.5 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 45991  bytes 4642490 (4.6 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 21716  bytes 1576083 (1.5 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 21716  bytes 1576083 (1.5 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

tunl0: flags=193<UP,RUNNING,NOARP>  mtu 1480
        inet 10.0.180.0  netmask 255.255.255.255
        tunnel   txqueuelen 1000  (IPIP Tunnel)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Kube-worker-2:

```
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.0.112  netmask 255.255.255.0  broadcast 192.168.0.255
        inet6 fe80::a00:27ff:feb2:4605  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:b2:46:05  txqueuelen 1000  (Ethernet)
        RX packets 44506  bytes 59994770 (59.9 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 9162  bytes 871321 (871.3 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

enp0s8: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.10.6  netmask 255.255.255.0  broadcast 192.168.10.255
        inet6 fe80::a00:27ff:febc:f8bb  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:bc:f8:bb  txqueuelen 1000  (Ethernet)
        RX packets 48202  bytes 27205308 (27.2 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 40987  bytes 4730086 (4.7 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 22374  bytes 1639840 (1.6 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 22374  bytes 1639840 (1.6 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

tunl0: flags=193<UP,RUNNING,NOARP>  mtu 1480
        inet 10.0.127.0  netmask 255.255.255.255
        tunnel   txqueuelen 1000  (IPIP Tunnel)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

generic@kube-worker-2:~$
```

Na każdym z Node'ów zainstalowany jest Docker (przykłady z kube-master'a):

```
generic@kube-master:~$ sudo apt-get install docker.io -y
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  bridge-utils containerd dns-root-data dnsmasq-base libidn11 pigz runc ubuntu-fan
Suggested packages:
  ifupdown aufs-tools cgroupfs-mount | cgroup-lite debootstrap docker-doc rinse zfs-fuse | zfsutils
The following NEW packages will be installed:
  bridge-utils containerd dns-root-data dnsmasq-base docker.io libidn11 pigz runc ubuntu-fan
0 upgraded, 9 newly installed, 0 to remove and 45 not upgraded.
Need to get 74.5 MB of archives.
After this operation, 361 MB of additional disk space will be used.
```

Uzyskanie do repozytoriów za pomocą HTTPS:

```
generic@kube-master:~$ sudo apt-get install apt-transport-https curl -y
Reading package lists... Done
Building dependency tree
Reading state information... Done
curl is already the newest version (7.68.0-1ubuntu2.7).
curl set to manually installed.
The following NEW packages will be installed:
  apt-transport-https
0 upgraded, 1 newly installed, 0 to remove and 45 not upgraded.
Need to get 4,680 B of archives.
After this operation, 162 kB of additional disk space will be used.
Get:1 http://pl.archive.ubuntu.com/ubuntu focal-updates/universe amd64 apt-transport-https all 2.0.6 [4,680 B]
Fetched 4,680 B in 0s (23.5 kB/s)
Selecting previously unselected package apt-transport-https.
(Reading database ... 71912 files and directories currently installed.)
Preparing to unpack .../apt-transport-https_2.0.6_all.deb ...
Unpacking apt-transport-https (2.0.6) ...
Setting up apt-transport-https (2.0.6) ...
generic@kube-master:~$
```

Dodanie klucza i repozytorium K8S na wszystkich hostach:

```
generic@kube-master:~$ curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add -
OK
generic@kube-master:~$ cat <<EOF | sudo tee /etc/apt/sources.list.d/kubernetes.list
> deb https://apt.kubernetes.io/ kubernetes-xenial main
> EOF
deb https://apt.kubernetes.io/ kubernetes-xenial main
generic@kube-master:~$
```

Po zaktualizowaniu paczek komendą sudo apt-get update, przystąpiliśmy do instalacji komponentów Kubernetesa na wszystkich węzłach:

- Kublet - podstawowy agent węzła. Tworzy, aktualizuje i niszczy kontenery w węźle Kubernetes.
- Kubeadm - narzędzie do tworzenia klastra
- Kubectl - narzędzie wiersza poleceń kubectl pozwala kontrolować klastry Kubernetes.

```
generic@kube-master:~$ sudo apt-get install -y kubelet=1.18.1-00
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  conntrack ebtables kubernetes-cni socat
Suggested packages:
  nftables
The following NEW packages will be installed:
  conntrack ebtables kubelet kubernetes-cni socat
0 upgraded, 5 newly installed, 0 to remove and 45 not upgraded.
Need to get 44.9 MB of archives.
After this operation, 187 MB of additional disk space will be used.
Get:1 http://pl.archive.ubuntu.com/ubuntu focal/main amd64 conntrack amd64 1:1.4.5-2 [30.3 kB]
Get:2 http://pl.archive.ubuntu.com/ubuntu focal/main amd64 ebtables amd64 2.0.11-3build1 [80.3 kB]
Get:4 http://pl.archive.ubuntu.com/ubuntu focal/main amd64 socat amd64 1.7.3.3-2 [323 kB]
Get:3 https://packages.cloud.google.com/apt kubernetes-xenial/main amd64 kubernetes-cni amd64 0.8.7-00 [25.0 MB]
Get:5 https://packages.cloud.google.com/apt kubernetes-xenial/main amd64 kubelet amd64 1.18.1-00 [19.4 MB]
Fetched 44.9 MB in 6s (7,919 kB/s)
Selecting previously unselected package conntrack.
(Reading database ... 71916 files and directories currently installed.)
Preparing to unpack .../conntrack_1%3a1.4.5-2_amd64.deb ...
Unpacking conntrack (1:1.4.5-2) ...
Selecting previously unselected package ebtables.
Preparing to unpack .../ebtables_2.0.11-3build1_amd64.deb ...
Unpacking ebtables (2.0.11-3build1) ...
Selecting previously unselected package kubernetes-cni.
Preparing to unpack .../kubernetes-cni_0.8.7-00_amd64.deb ...
Unpacking kubernetes-cni (0.8.7-00) ...
Selecting previously unselected package socat.
Preparing to unpack .../socat_1.7.3.3-2_amd64.deb ...
Unpacking socat (1.7.3.3-2) ...
Selecting previously unselected package kubelet.
Preparing to unpack .../kubelet_1.18.1-00_amd64.deb ...
Unpacking kubelet (1.18.1-00) ...
Setting up conntrack (1:1.4.5-2) ...
Setting up ebtables (2.0.11-3build1) ...
Setting up socat (1.7.3.3-2) ...
Setting up kubernetes-cni (0.8.7-00) ...
Setting up kubelet (1.18.1-00) ...
Created symlink /etc/systemd/system/multi-user.target.wants/kubelet.service → /lib/systemd/system/kubelet.service.
Processing triggers for man-db (2.9.1-1) ...
generic@kube-master:~$
```

```
generic@kube-master:~$ sudo apt-get install -y kubeadm=1.18.1-00
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  cri-tools kubectl
The following NEW packages will be installed:
  cri-tools kubeadm kubectl
0 upgraded, 3 newly installed, 0 to remove and 46 not upgraded.
Need to get 28.3 MB of archives.
After this operation, 117 MB of additional disk space will be used.
Get:1 https://packages.cloud.google.com/apt kubernetes-xenial/main amd64 cri-tools amd64 1.19.0-00 [11.2 MB]
Get:2 https://packages.cloud.google.com/apt kubernetes-xenial/main amd64 kubectl amd64 1.23.1-00 [8,928 kB]
Get:3 https://packages.cloud.google.com/apt kubernetes-xenial/main amd64 kubeadm amd64 1.18.1-00 [8,163 kB]
Fetched 28.3 MB in 5s (5,678 kB/s)
Selecting previously unselected package cri-tools.
(Reading database ... 72004 files and directories currently installed.)
Preparing to unpack .../cri-tools_1.19.0-00_amd64.deb ...
Unpacking cri-tools (1.19.0-00) ...
Selecting previously unselected package kubectl.
Preparing to unpack .../kubectl_1.23.1-00_amd64.deb ...
Unpacking kubectl (1.23.1-00) ...
Selecting previously unselected package kubeadm.
Preparing to unpack .../kubeadm_1.18.1-00_amd64.deb ...
Unpacking kubeadm (1.18.1-00) ...
Setting up kubectl (1.23.1-00) ...
Setting up cri-tools (1.19.0-00) ...
Setting up kubeadm (1.18.1-00) ...
generic@kube-master:~$
```

```
generic@kube-master:~$ sudo apt-get install -y kubectl=1.18.1-00
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages will be DOWNGRADED:
  kubectl
0 upgraded, 0 newly installed, 1 downgraded, 0 to remove and 47 not upgraded.
E: Packages were downgraded and -y was used without --allow-downgrades.
generic@kube-master:~$
```

Potwierdzenie poprawnej instalacji:

```
generic@kube-master:~$ sudo apt-mark hold kubelet kubeadm kubectl
kubelet set on hold.
kubeadm set on hold.
kubectl set on hold.
generic@kube-master:~$
```

Konfiguracja /etc/fstab - usunięcie swap entry i dodanie na workerach połączenia od mastera przez interfejs eps0s8 o adresie IP 192.168.10.4

```
127.0.0.1 localhost
127.0.1.1 kube-worker-1

# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters

192.168.10.4 kube-master
```

```
127.0.0.1 localhost
#127.0.1.1 kube-master

# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters

192.168.10.4 kube-master
```

```
127.0.0.1 localhost
127.0.1.1 kube-worker-2

# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters

192.168.10.4 kube-master
```

```
generic@kube-master:~$ sudo swapoff -a
[sudo] password for generic:
generic@kube-master:~$ sudo nano /etc/fstab
generic@kube-master:~$ free -h
              total        used        free      shared  buff/cache   available
Mem:          1.9Gi       239Mi       327Mi       1.0Mi       1.4Gi       1.5Gi
Swap:            0B          0B          0B
generic@kube-master:~$
```

Inicjalizacja klastra na Master Node'dzie:

sudo kubeadm init --control-plane-endpoint kube-master:6443 --pod-network-cidr 10.10.0.0/16

```
generic@kube-master:~$ sudo kubeadm init --control-plane-endpoint kube-master:6443 --pod-network-cidr 10.10.0.0/16
[0109 13:06:30.813633   16423 version.go:252] remote version is much newer: v1.23.1; falling back to: stable-1.18
W0109 13:06:38.783728   16423 configset.go:202] WARNING: kubeadm cannot validate component configs for API groups [kubelet.config.k8s.io kubeproxy.config.k8s.io]
[init] Using Kubernetes version: v1.18.20
[preflight] Running pre-flight checks
        [WARNING IsDockerSystemdCheck]: detected "cgroupfs" as the Docker cgroup driver. The recommended driver is "systemd". Please follow the guide at https://kubernetes.io/docs/setup/cri/
        [WARNING SystemVerification]: this Docker version is not on the list of validated versions: 20.10.7. Latest validated version: 19.03
[preflight] Pulling images required for setting up a Kubernetes cluster
[preflight] This might take a minute or two, depending on the speed of your internet connection
[preflight] You can also perform this action in beforehand using 'kubeadm config images pull'
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Starting the kubelet
[certs] Using certificateDir folder "/etc/kubernetes/pki"
[certs] Generating "ca" certificate and key
[certs] Generating "apiserver" certificate and key
[certs] apiserver serving cert is signed for DNS names [kube-master kubernetes kubernetes.default kubernetes.default.svc kubernetes.default.svc.cluster.local kube-master] and IPs [10.96.0.1 192.168.10.4]
[certs] Generating "apiserver-kubelet-client" certificate and key
[certs] Generating "front-proxy-ca" certificate and key
[certs] Generating "front-proxy-client" certificate and key
[certs] Generating "etcd/ca" certificate and key
[certs] Generating "etcd/server" certificate and key
[certs] etcd/server serving cert is signed for DNS names [kube-master localhost] and IPs [192.168.10.4 127.0.0.1 ::1]
[certs] Generating "etcd/peer" certificate and key
[certs] etcd/peer serving cert is signed for DNS names [kube-master localhost] and IPs [192.168.10.4 127.0.0.1 ::1]
[certs] Generating "etcd/healthcheck-client" certificate and key
[certs] Generating "apiserver-etcd-client" certificate and key
[certs] Generating "sa" key and public key
[kubeconfig] Using kubeconfig folder "/etc/kubernetes"
[kubeconfig] Writing "admin.conf" kubeconfig file
[kubeconfig] Writing "kubelet.conf" kubeconfig file
[kubeconfig] Writing "controller-manager.conf" kubeconfig file
[kubeconfig] Writing "scheduler.conf" kubeconfig file
[control-plane] Using manifest folder "/etc/kubernetes/manifests"
[control-plane] Creating static Pod manifest for "kube-apiserver"
[control-plane] Creating static Pod manifest for "kube-controller-manager"
W0109 13:07:57.978217   16423 manifests.go:225] the default kube-apiserver authorization-mode is "Node,RBAC"; using "Node,RBAC"
[control-plane] Creating static Pod manifest for "kube-scheduler"
W0109 13:07:58.001964   16423 manifests.go:225] the default kube-apiserver authorization-mode is "Node,RBAC"; using "Node,RBAC"
[etcd] Creating static Pod manifest for local etcd in "/etc/kubernetes/manifests"
[wait-control-plane] Waiting for the kubelet to boot up the control plane as static Pods from directory "/etc/kubernetes/manifests". This can take up to 4m0s
[apiclient] All control plane components are healthy after 24.522095 seconds
[upload-config] Storing the configuration used in ConfigMap "kubeadm-config" in the "kube-system" Namespace
[kubelet] Creating a ConfigMap "kubelet-config-1.18" in namespace kube-system with the configuration for the kubelets in the cluster
[upload-certs] Skipping phase. Please see --upload-certs
[mark-control-plane] Marking the node kube-master as control-plane by adding the label "node-role.kubernetes.io/master=''"
[mark-control-plane] Marking the node kube-master as control-plane by adding the taints [node-role.kubernetes.io/master:NoSchedule]
[bootstrap-token] Using token: mhpk4b.btwe8xii0wri0yOse
[bootstrap-token] Configuring bootstrap tokens, cluster-info ConfigMap, RBAC Roles
[bootstrap-token] configured RBAC rules to allow Node Bootstrap tokens to get nodes
[bootstrap-token] configured RBAC rules to allow Node Bootstrap tokens to post CSRs in order for nodes to get long term certificate credentials
[bootstrap-token] configured RBAC rules to allow the csrapprover controller automatically approve CSRs from a Node Bootstrap Token
[bootstrap-token] configured RBAC rules to allow certificate rotation for all node client certificates in the cluster
[bootstrap-token] Creating the "cluster-info" ConfigMap in the "kube-public" namespace
[kubelet-finalize] Updating "/etc/kubernetes/kubelet.conf" to point to a rotatable kubelet client certificate and key
[addons] Applied essential addon: CoreDNS
[addons] Applied essential addon: kube-proxy

Your Kubernetes control-plane has initialized successfully!
```

Uwierzytelnienie, aby połączyć się z utworzonym klastrem:

```
generic@kube-master:~$ mkdir -p $HOME/.kube
generic@kube-master:~$ sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
[sudo] password for generic:
generic@kube-master:~$ sudo chown $(id -u):$(id -g) $HOME/.kube/config
generic@kube-master:~$
```

Utworzony klaster:

```
generic@kube-master:~$ kubectl cluster-info
Kubernetes control plane is running at https://kube-master:6443
KubeDNS is running at https://kube-master:6443/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
generic@kube-master:~$
```

Zastosowaliśmy jeden z najpopularniejszych pluginów do sieciowania Calico:

```
generic@kube-master:~$ curl https://docs.projectcalico.org/manifests/calico.yaml -O
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  212k  100  212k    0     0   520k      0 --:--:-- --:--:-- --:--:--  520k
generic@kube-master:~$
```

```
generic@kube-master:~$ kubectl apply -f calico.yaml
configmap/calico-config created
customresourcedefinition.apiextensions.k8s.io/bgpconfigurations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/bgppeers.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/blockaffinities.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/caliconodestatuses.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/clusterinformations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/felixconfigurations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/globalnetworkpolicies.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/globalnetworksets.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/hostendpoints.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipamblocks.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipamconfigs.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipamhandles.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ippools.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipreservations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/kubecontrollersconfigurations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/networkpolicies.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/networksets.crd.projectcalico.org created
clusterrole.rbac.authorization.k8s.io/calico-kube-controllers created
clusterrolebinding.rbac.authorization.k8s.io/calico-kube-controllers created
clusterrole.rbac.authorization.k8s.io/calico-node created
clusterrolebinding.rbac.authorization.k8s.io/calico-node created
daemonset.apps/calico-node created
serviceaccount/calico-node created
deployment.apps/calico-kube-controllers created
serviceaccount/calico-kube-controllers created
poddisruptionbudget.policy/calico-kube-controllers created
generic@kube-master:~$
```

Wypisanie podów:

```
generic@kube-master:~$ kubectl get pods -A
NAMESPACE     NAME                                       READY   STATUS    RESTARTS   AGE
kube-system   calico-kube-controllers-7f94cf5997-6bzzc   1/1     Running   0          10m
kube-system   calico-node-fgh7s                          1/1     Running   0          10m
kube-system   coredns-66bff467f8-5d4rx                   1/1     Running   0          64m
kube-system   coredns-66bff467f8-rzxnf                   1/1     Running   0          64m
kube-system   etcd-kube-master                           1/1     Running   0          64m
kube-system   kube-apiserver-kube-master                 1/1     Running   0          64m
kube-system   kube-controller-manager-kube-master        1/1     Running   0          64m
kube-system   kube-proxy-ljvfr                           1/1     Running   0          64m
kube-system   kube-scheduler-kube-master                 1/1     Running   1          64m
generic@kube-master:~$
```

Aby dodać do klastra worker-Node'y, wygenerowaliśmy Token:

```
generic@kube-master:~$ sudo kubeadm token create
[sudo] password for generic:
W0109 14:16:02.298989   47432 configset.go:202] WARNING: kubeadm cannot validate component configs for API groups [kubelet.config.k8s.io kubeproxy.config.k8s.io]
nn66jt.98nc0v232g6txgv0
generic@kube-master:~$ sudo kubeadm token list
TOKEN                     TTL         EXPIRES                     USAGES                   DESCRIPTION                                                EXTRA GROUPS
mhpk4b.btwe8xi0wri0y0so   22h         2022-01-10T13:08:23Z        authentication,signing   The default bootstrap token generated by 'kubeadm init'.   system:bootstrappers:kubeadm:default-node-token
nn66jt.98nc0v232g6txgv0   23h         2022-01-10T14:16:02Z        authentication,signing   <none>                                                     system:bootstrappers:kubeadm:default-node-token
generic@kube-master:~$ openssl x509 -pubkey -in /etc/kubernetes/pki/ca.crt |
>    openssl rsa -pubin -outform der 2>/dev/null |
>    openssl dgst -sha256 -hex | sed 's/^.* //'
1b7f104f62f9ca675b07a37a02e3cc08f70f44467b6b1b5fb7ce711a28d10e32
generic@kube-master:~$
```

```
generic@kube-worker-1:~$ sudo kubeadm join --token nn66jt.98nc0v232g6txgv0 kube-master:6443 --discovery-token-ca-cert-hash sha256:1b7f104f62f9ca675b07a37a02e3cc08f70f44467b6b1b5fb7ce711a28d10e32
[sudo] password for generic:
W0109 14:26:44.049141   25806 join.go:346] [preflight] WARNING: JoinControlPane.controlPlane settings will be ignored when control-plane flag is not set.
[preflight] Running pre-flight checks
        [WARNING IsDockerSystemdCheck]: detected "cgroupfs" as the Docker cgroup driver. The recommended driver is "systemd". Please follow the guide at https://kubernetes.io/docs/setup/cri/
        [WARNING SystemVerification]: this Docker version is not on the list of validated versions: 20.10.7. Latest validated version: 19.03
[preflight] Reading configuration from the cluster...
[preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -oyaml'
[kubelet-start] Downloading configuration for the kubelet from the "kubelet-config-1.18" ConfigMap in the kube-system namespace
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
[kubelet-start] Starting the kubelet
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap...

This node has joined the cluster:
* Certificate signing request was sent to apiserver and a response was received.
* The Kubelet was informed of the new secure connection details.

Run 'kubectl get nodes' on the control-plane to see this node join the cluster.
```

```
generic@kube-worker-2:~$ sudo kubeadm join --token nn66jt.98nc0v232g6txgv0 kube-master:6443 --discovery-token-ca-cert-hash sha256:1b7f104f62f9ca675b07a37a02e3cc08f70f44467b6b1b5fb7ce711a28d10e32
[sudo] password for generic:
W0109 14:33:05.041765   26297 join.go:346] [preflight] WARNING: JoinControlPane.controlPlane settings will be ignored when control-plane flag is not set.
[preflight] Running pre-flight checks
        [WARNING IsDockerSystemdCheck]: detected "cgroupfs" as the Docker cgroup driver. The recommended driver is "systemd". Please follow the guide at https://kubernetes.io/docs/setup/cri/
        [WARNING SystemVerification]: this Docker version is not on the list of validated versions: 20.10.7. Latest validated version: 19.03
[preflight] Reading configuration from the cluster...
[preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -oyaml'
[kubelet-start] Downloading configuration for the kubelet from the "kubelet-config-1.18" ConfigMap in the kube-system namespace
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
[kubelet-start] Starting the kubelet
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap...

This node has joined the cluster:
* Certificate signing request was sent to apiserver and a response was received.
* The Kubelet was informed of the new secure connection details.

Run 'kubectl get nodes' on the control-plane to see this node join the cluster.

generic@kube-worker-2:~$
```

Poprawnie skonfigurowany klaster z dodanymi dwoma Worker Node'ami:

```
generic@kube-master:~$ kubectl get nodes
NAME           STATUS   ROLES    AGE    VERSION
kube-master    Ready    master   86m    v1.18.1
kube-worker-1  Ready    <none>   8m4s   v1.18.1
kube-worker-2  Ready    <none>   103s   v1.18.1
generic@kube-master:~$
```

3. Orkiestracja kontenerów na klastrze K8S

3a. Stworzenie deskryptora kontenera (Dockerfile), zbudowanie obrazu oraz umieszczenie go w repozytorium obrazów – DockerHub.

Obraz kontenera zbudowaliśmy na bazie systemu Ubuntu. Po zainstalowaniu odpowiednich pakietów dockera, obraz pojawił się na liście:

```
generic@kube-master:~$ sudo apt-get install docker-ce docker-ce-cli containerd.io
Reading package lists... Done
Building dependency tree
Reading state information... Done
You might want to run 'apt --fix-broken install' to correct these.
The following packages have unmet dependencies:
 containerd.io : Conflicts: containerd
                 Conflicts: runc
 docker-ce : Conflicts: docker.io but 20.10.7-0ubuntu5~20.04.2 is to be installed
             Recommends: docker-ce-rootless-extras but it is not going to be installed
 docker-ce-cli : Conflicts: docker.io but 20.10.7-0ubuntu5~20.04.2 is to be installed
                 Recommends: docker-scan-plugin but it is not going to be installed
 linux-generic : Depends: linux-headers-generic (= 5.4.0.96.100) but 5.4.0.92.96 is to be installed
E: Unmet dependencies. Try 'apt --fix-broken install' with no packages (or specify a solution).
generic@kube-master:~$
```

```
root@kube-master:~# docker images
REPOSITORY                                TAG        IMAGE ID       CREATED         SIZE
ubuntu                                    latest     d13c942271d6   2 weeks ago     72.8MB
nginx                                     latest     605c77e624dd   3 weeks ago     141MB
calico/node                               v3.21.2    f1bca4d4ced2   6 weeks ago     214MB
calico/pod2daemon-flexvol                 v3.21.2    7778dd57e506   6 weeks ago     21.3MB
calico/cni                                v3.21.2    4c5c32530391   6 weeks ago     239MB
calico/kube-controllers                   v3.21.2    b20652406028   6 weeks ago     132MB
k8s.gcr.io/kube-proxy                     v1.18.20   27f8b8d51985   7 months ago    117MB
k8s.gcr.io/kube-apiserver                 v1.18.20   7d8d2960de69   7 months ago    173MB
k8s.gcr.io/kube-scheduler                 v1.18.20   a05a1a79adaa   7 months ago    96.1MB
k8s.gcr.io/kube-controller-manager        v1.18.20   e7c545a60706   7 months ago    162MB
k8s.gcr.io/pause                          3.2        80d28bedfe5d   23 months ago   683kB
k8s.gcr.io/coredns                        1.6.7      67da37a9a360   24 months ago   43.8MB
k8s.gcr.io/etcd                           3.4.3-0    303ce5db0e90   2 years ago     288MB
root@kube-master:~#
```

```
root@kube-master:~# docker run -it ubuntu /bin/bash
root@bf2fce41364c:/#
```

Aplikacja webowa w języku python, korzystająca z biblioteki Flask. Po uruchomieniu w przeglądarce wyświetla ona napis “SWUS-2”.

```python
from flask import Flask
app = Flask(__name__)

@app.route("/")
def main():
    return "SWUS-2"

if __name__ == "__main__":
    print ("Starting the application")
    app.run(host='0.0.0.0')
```

requirements.txt

```
1  flask
```

Zawartości pliku Dockerfile, opisująca między innymi ekspozycję portu 5000:

Dockerfile > ...

```
1   FROM ubuntu
2
3   #MAINTAINER bartosz "bartoszdorobek19@gmail.com"
4
5   RUN apt-get update -y
6   RUN apt-get install python3 -y
7   RUN apt-get install python3-pip python3-dev -y
8   #RUN nginx
9
10  WORKDIR /
11
12  COPY ./requirements.txt ./requirements.txt
13
14  RUN pip3 install -r /requirements.txt
15
16  COPY . /
17
18  EXPOSE 5000
19
20  CMD ["python3", "src/app.py"]
```

Zbudowanie obrazu aplikacji za pomocą komendy: docker build –t [image_name]:[tag]

```
root@kube-master:~/my_docker# ls -lrt
total 12
-rw-r--r-- 1 root root  313 Jan 21 11:45 Dockerfile
-rw-r--r-- 1 root root    6 Jan 21 11:45 requirements.txt
drwxr-xr-x 2 root root 4096 Jan 21 11:50 src
root@kube-master:~/my_docker# docker build -t swus2:v1 .
Sending build context to Docker daemon  4.608kB
Step 1/10 : FROM ubuntu
 ---> d13c942271d6
Step 2/10 : RUN apt-get update -y
 ---> Running in 56b057839985
Get:1 http://archive.ubuntu.com/ubuntu focal InRelease [265 kB]
Get:2 http://security.ubuntu.com/ubuntu focal-security InRelease [114 kB]
Get:3 http://archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Get:4 http://security.ubuntu.com/ubuntu focal-security/main amd64 Packages [1470 kB]
Get:5 http://archive.ubuntu.com/ubuntu focal-backports InRelease [108 kB]
Get:6 http://archive.ubuntu.com/ubuntu focal/restricted amd64 Packages [33.4 kB]
Get:7 http://archive.ubuntu.com/ubuntu focal/multiverse amd64 Packages [177 kB]
Get:8 http://archive.ubuntu.com/ubuntu focal/universe amd64 Packages [11.3 MB]
Get:9 http://security.ubuntu.com/ubuntu focal-security/restricted amd64 Packages [889 kB]
Get:10 http://security.ubuntu.com/ubuntu focal-security/multiverse amd64 Packages [30.1 kB]
Get:11 http://security.ubuntu.com/ubuntu focal-security/universe amd64 Packages [839 kB]
Get:12 http://archive.ubuntu.com/ubuntu focal/main amd64 Packages [1275 kB]
Get:13 http://archive.ubuntu.com/ubuntu focal-updates/multiverse amd64 Packages [33.7 kB]
Get:14 http://archive.ubuntu.com/ubuntu focal-updates/main amd64 Packages [1895 kB]
Get:15 http://archive.ubuntu.com/ubuntu focal-updates/universe amd64 Packages [1120 kB]
Get:16 http://archive.ubuntu.com/ubuntu focal-updates/restricted amd64 Packages [952 kB]
Get:17 http://archive.ubuntu.com/ubuntu focal-backports/universe amd64 Packages [22.4 kB]
Get:18 http://archive.ubuntu.com/ubuntu focal-backports/main amd64 Packages [50.8 kB]
Fetched 20.7 MB in 57s (365 kB/s)
Reading package lists...
Removing intermediate container 56b057839985
 ---> c64896460d9b
Step 3/10 : RUN apt-get install python3 -y
 ---> Running in 5dd3d46bfb75
Reading package lists...
Building dependency tree...
```

Potwierdzenie poprawnego zbudowania obrazu

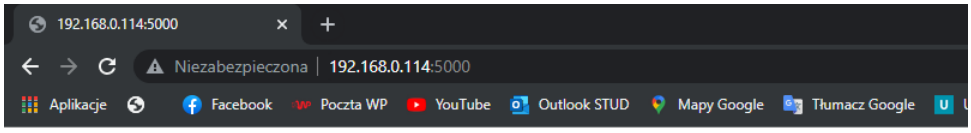
```
Successfully built 3e49720c08b8
Successfully tagged swus2:v1
root@kube-master:~/my_docker# docker images
REPOSITORY                                TAG        IMAGE ID       CREATED              SIZE
swus2                                     v1         3e49720c08b8   About a minute ago   419MB
ubuntu                                    latest     d13c942271d6   2 weeks ago          72.8MB
calico/node                               v3.21.2    f1bca4d4ced2   6 weeks ago          214MB
calico/pod2daemon-flexvol                 v3.21.2    7778dd57e506   6 weeks ago          21.3MB
calico/cni                                v3.21.2    4c5c32530391   6 weeks ago          239MB
calico/kube-controllers                   v3.21.2    b20652406028   6 weeks ago          132MB
k8s.gcr.io/kube-proxy                     v1.18.20   27f8b8d51985   7 months ago         117MB
k8s.gcr.io/kube-apiserver                 v1.18.20   7d8d2960de69   7 months ago         173MB
k8s.gcr.io/kube-controller-manager        v1.18.20   e7c545a60706   7 months ago         162MB
k8s.gcr.io/kube-scheduler                 v1.18.20   a05a1a79adaa   7 months ago         96.1MB
k8s.gcr.io/pause                          3.2        80d28bedfe5d   23 months ago        683kB
k8s.gcr.io/coredns                        1.6.7      67da37a9a360   24 months ago        43.8MB
k8s.gcr.io/etcd                           3.4.3-0    303ce5db0e90   2 years ago          288MB
root@kube-master:~/my_docker#
```

Lokalne uruchomienie obrazu:

```
root@kube-master:~/my_docker# docker run -p5000:5000 swus2:v1
 * Serving Flask app 'app' (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on all addresses.
   WARNING: This is a development server. Do not use it in a production deployment.
 * Running on http://172.17.0.2:5000/ (Press CTRL+C to quit)
```

Potwierdzenie poprawnego działania aplikacji webowej na stronie:

```
generic@kube-master:~$ curl http://localhost:5000
SWUS-2generic@kube-master:~$ sudo su -
```



SWUS-2

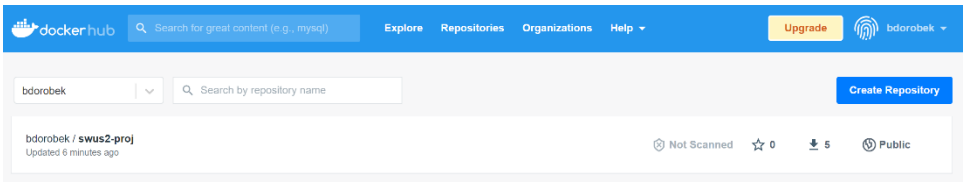
```
root@kube-master:~# docker run -d -t --name "SwusContainer" swus2:v1 /bin/bash
e72e2e1418154202435e1bf553880f6b80490fe2a94b1d0873bf9f031265c72a
root@kube-master:~# docker container ls
CONTAINER ID   IMAGE          COMMAND          CREATED          STATUS          PORTS      NAMES
e72e2e141815   swus2:v1       "/bin/bash"      9 seconds ago    Up 6 seconds    5000/tcp   SwusContainer
a3ac86a8df56   f1bca4d4ced2   "start_runit"    18 minutes ago   Up 18 minutes              k8s_calico-node_calico-node-fgh7s_kube-system_6d7b
```

Wypchnięcie obrazu aplikacji do DockerHuba – *docker push*:

```
root@kube-master:~# docker commit e72e2e141815 bdorobek/swus2-proj
sha256:03ac3388718727ac5ff3426c3919f0a116a1c66a06f119d63a6cee540674b37c
root@kube-master:~# docker image ls
REPOSITORY                                TAG        IMAGE ID       CREATED          SIZE
bdorobek/swus2-proj                       latest     03ac33887187   27 seconds ago   419MB
swus2                                     v1         76866a53cb53   27 hours ago     419MB
ubuntu                                    latest     d13c942271d6   2 weeks ago      72.8MB
calico/node                               v3.21.2    f1bca4d4ced2   6 weeks ago      214MB
calico/pod2daemon-flexvol                 v3.21.2    7778dd57e506   6 weeks ago      21.3MB
calico/cni                                v3.21.2    4c5c32530391   6 weeks ago      239MB
calico/kube-controllers                   v3.21.2    b20652406028   6 weeks ago      132MB
k8s.gcr.io/kube-proxy                     v1.18.20   27f8b8d51985   7 months ago     117MB
k8s.gcr.io/kube-apiserver                 v1.18.20   7d8d2960de69   7 months ago     173MB
k8s.gcr.io/kube-controller-manager        v1.18.20   e7c545a60706   7 months ago     162MB
k8s.gcr.io/kube-scheduler                 v1.18.20   a05a1a79adaa   7 months ago     96.1MB
k8s.gcr.io/pause                          3.2        80d28bedfe5d   23 months ago    683kB
k8s.gcr.io/coredns                        1.6.7      67da37a9a360   24 months ago    43.8MB
k8s.gcr.io/etcd                           3.4.3-0    303ce5db0e90   2 years ago      288MB
root@kube-master:~# docker login
Authenticating with existing credentials...
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
root@kube-master:~# docker push bdorobek/swus2-proj:latest
The push refers to repository [docker.io/bdorobek/swus2-proj]
1e245c2f743e: Pushed
9e3e66d20871: Pushed
17f85308e52c: Pushed
6e61e1cf0255: Pushed
1cda4902661e: Pushed
a857e181507e: Pushed
0eba131dffd0: Mounted from library/ubuntu
latest: digest: sha256:0de6a6408eb4fbf21d5e7325c5251b6d379c0b6225350373b408d264b8a6979c size: 1790
root@kube-master:~#
```

Repozytorium z obrazem naszej aplikacji na platformie DockerHub:



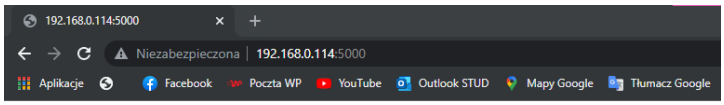
3b. Stworzenie deskryptora poda w K8S

Deskryptor poda z obrazem naszej aplikacji MME.yaml:

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: mme
  labels:
    name: mme
spec:
  containers:
  - name: mme
    image: bdorobek/swus2-proj
    ports:
      - containerPort: 5000
```

Kontener służy jako przykład mechanizmu orkiestracji.

```
generic@kube-master:~$ kubectl get pods -o wide
I0130 20:48:16.208723   35951 request.go:665] Waited for 1.084956862s due to client-side throttling, not priority and fairness, request: GET:https://kube-master:6443/apis/coordination.k8s.io/v1beta1?timeout=32s
NAME   READY   STATUS    RESTARTS   AGE   IP           NODE          NOMINATED NODE   READINESS GATES
mme    1/1     Running   0          11m   10.0.127.20   kube-worker-2   <none>           <none>
generic@kube-master:~$
```



SWUS-2

Deskryptor drugiego poda upf.yaml

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: upf
  labels:
    app: upf
spec:
  containers:
  - name: upf
    image: nginx
    ports:
      - containerPort: 80
```

```
generic@kube-master:~$ kubectl get pods -o wide
I0130 21:02:38.243877   50062 request.go:665] Waited for 1.069952991s due to client-side throttling, not priority and fairness, request: GET:https://kube-master:6443/apis/coordination.k8s.io/v1?timeout=32s
NAME    READY   STATUS    RESTARTS   AGE   IP            NODE            NOMINATED NODE   READINESS GATES
mme     1/1     Running   0          25m   10.0.127.20   kube-worker-2   <none>           <none>
upf     1/1     Running   0          45s   10.0.180.45   kube-worker-1   <none>           <none>
generic@kube-master:~$
```

3c. Stworzenie deskryptora deploymentu w K8S

Deskryptor deploymentu network.yaml:

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: mme
  labels:
    app: mme
spec:
  replicas: 2
  selector:
    matchLabels:
      app: mme
  template:
    metadata:
      labels:
        app: mme
    spec:
      containers:
        - name: mme
          image: bdorobek/swus2-proj
          securityContext:
            runAsUser: 1001
            runAsNonRoot: true
            allowPrivilegeEscalation: false
            capabilities:
              drop: ["all"]
          resources:
            requests:
              memory: "10Mi"
              cpu: "100m"
            limits:
              memory: "20Mi"
              cpu: "250m"
          livenessProbe:
            httpGet:
              path: /healthz
              port: 5000
            initialDelaySeconds: 10
            periodSeconds: 15
          readinessProbe:
            httpGet:
              path: /healthz
              port: 5000
            initialDelaySeconds: 10
            periodSeconds: 15
          ports:
          - containerPort: 5000
---
```

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: upf
  labels:
    app: upf
spec:
  replicas: 2
  selector:
    matchLabels:
      app: upf
  template:
    metadata:
      labels:
        app: upf
    spec:
      containers:
        - name: upf
          image: nginx
          securityContext:
            runAsUser: 1001
            runAsNonRoot: true
            allowPrivilegeEscalation: false
            capabilities:
              drop: ["all"]
          resources:
            requests:
              memory: "10Mi"
              cpu: "100m"
            limits:
              memory: "20Mi"
              cpu: "250m"
          livenessProbe:
            httpGet:
              path: /healthz
              port: 80
            initialDelaySeconds: 10
            periodSeconds: 15
          readinessProbe:
            httpGet:
              path: /healthz
              port: 80
            initialDelaySeconds: 10
            periodSeconds: 15
          ports:
          - containerPort: 80
```

W deskryptorze zdefiniowaliśmy parametr replicas, który zmieniliśmy w trakcie z testów z wartości 2 na 3. Zdefiniowaliśmy także takie parametry jak periodSeconds i initialDelaySeconds, w celu uniknięcia błędu CrashLoopBackOff, mogącego wskazywać na problem przy uruchamianiu poda.

```
generic@kube-master:~$ kubectl apply -f network.yaml
deployment.apps/mme created
deployment.apps/upf created
generic@kube-master:~$ kubectl get pods -o wide
NAME                   READY   STATUS             RESTARTS   AGE     IP             NODE            NOMINATED NODE   READINESS GATES
mme                    1/1     Running            2          3h29m   10.0.127.22    kube-worker-2   <none>           <none>
mme-56589576b8-7wxkd   0/1     Running            1          12s     10.0.127.37    kube-worker-2   <none>           <none>
mme-56589576b8-v8p2h   0/1     ErrImagePull       0          12s     <none>         kube-worker-1   <none>           <none>
upf                    1/1     Running            1          131m    10.0.180.55    kube-worker-1   <none>           <none>
upf-d9b844c8d-pb7jd    0/1     Running            0          12s     10.0.180.3     kube-worker-2   <none>           <none>
upf-d9b844c8d-rgbhj    0/1     ErrImagePull       0          12s     <none>         kube-worker-1   <none>           <none>
generic@kube-master:~$
```

Wszystkie pody rozłożyły się równomiernie na dostępnych worker Node'ach – na każdym po trzy.

W przypadku usunięcia poda poleceniem *kubectl delete pod [pod_name]* w punkcie b, kiedy nie ma ustawionego deploymentu, pod po prostu znika. Kiedy ustawiliśmy deployment w punkcie c, który ma zdefiniowaną liczbę replik poda, próba usunięcia jednej z nich kończy się utworzeniem nowej, tak aby sumaryczna liczba zgadzała się z tą zdefiniowaną w deskryptorze deploymentu.

Dzięki zastosowaniu Kubernetes zarządzanie sieciami opartymi o wiele kontenerów jest dużo bardziej efektywne od tradycyjnego podejścia, ponieważ jest to system łatwo skalowalny. Wraz ze wzrostem zapotrzebowania możemy automatycznie aktywować kolejne maszyny, oraz zarządzać ich cyklem życia.