

Kubernetes: model działania

Od trzech instancji do działającej usługi

mgr inż. Jakub Woźniak

Zarządzanie Systemami Rozproszonymi · Semestr zimowy 2026/27

Przypadek

Mamy obraz workera systemu zamówień. Chcemy utrzymywać trzy instancje na kilku serwerach, także po zakończeniu procesu lub awarii serwera.

- Kto wybierze miejsce uruchomienia każdej instancji?
- Kto zauważy brak instancji i uruchomi zastępstwo?
- Jak klient znajdzie API, gdy jego instancje zostaną wymienione?

Kontynuacja wykładu 2: obraz umożliwia tworzenie wymieniających instancji.

- Klaster, węzeł i Pod: gdzie działa aplikacja.
- Stan oczekiwany, kontrolery i Deployment: kto utrzymuje instancje.
- Scheduler: gdzie zmieści się kolejny Pod.
- Service i etykiety: jak skierować ruch do aplikacji.
- 🏰 AKS: odpowiedzialność i pojemność klastra.
- Diagnostyka: Pody działają, ale Service nie obsługuje ruchu.

Pojęcie	Rola
Klaster	Control plane oraz węzły wykonujące aplikacje
Węzeł	Maszyna fizyczna lub wirtualna uruchamiająca Pody
Pod	Jednostka uruchomienia jednego lub kilku kontenerów na jednym węźle

- Kontenery Poda współdzielą przestrzeń sieciową i mogą montować wspólne wolumeny.
- W przykładzie: jeden Pod zawiera jeden kontener workera; trzy repliki to trzy Pody.
- Zastąpienie Poda oznacza nowy obiekt; jego adres IP może się zmienić.

spec: stan oczekiwany

Chcemy utrzymywać trzy repliki workera.

status: stan obserwowany

Istnieją trzy Pody, ale tylko dwa są gotowe.

- API Kubernetes udostępnia obiekty opisujące konfigurację i obserwowany stan.
- Kontrolery cyklicznie dążą do stanu oczekiwanego; reakcje wymagają czasu.
- Brak zasobów lub błąd aplikacji może uniemożliwić osiągnięcie celu.

Gotowość oznacza zdolność do obsługi pracy według warunków Poda. Sondy omówimy na wykładzie 4.

Deployment: worker



ReplicaSet



Pod A / B / C

- Deployment opisuje szablon Poda i zarządza aktualizacją aplikacji.
- ReplicaSet utrzymuje zadaną liczbę Podów na podstawie szablonu i selektora etykiet.
- Zmiana szablonu, np. obrazu, powoduje wdrożenie przez odpowiedni ReplicaSet.

Liczba a gotowość

Trzy istniejące Pody, z których jeden jest niegotowy, nie oznaczają, że ReplicaSet automatycznie utworzy czwarty Pod.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: worker
spec:
  replicas: 3
  selector:
    matchLabels: {app: worker}
  template:
    metadata:
      labels: {app: worker}
    spec:
      containers:
        - name: worker
          image: registry.example.test/worker:v1
```

selector wskazuje etykiety Podów. Przykład pomija zasoby, sondy i konfigurację; rejestr jest przykładowy.

Control plane

- API server: dostęp do obiektów.
- etcd: trwały zapis stanu obiektów.
- Kontrolery: reakcje na różnice stanu.
- Scheduler: wybór węzła dla Poda.

Na węźle

- Kubelet obsługuje przypisane Pody.
- Runtime uruchamia kontenery na zlecenie kubeleta.
- Komponenty sieci i magazynu zapewniają łączność i wolumeny.



Odczyt stanu



Porównanie z celem



Działanie

ReplicaSet: oczekiwane 3 Pody, istnieją 2

1. Kontroler tworzy obiekt brakującego Poda przez API.
 2. Scheduler wybiera węzeł i zapisuje przypisanie przez API.
 3. Kubelet zleca runtime'owi uruchomienie kontenera.
 4. Stan jest ponownie obserwowany; pętla trwa dalej.
- Zapis obiektu, uruchomienie procesu i osiągnięcie gotowości to osobne etapy.

Request CPU to deklarowane zapotrzebowanie. 1000m = 1 CPU. Allocatable to pojemność udostępniona Podom.

Węzeł	Allocatable CPU	Suma requests	Nowy Pod: 600m
A	2000m	1600m	Nie: $1600 + 600 > 2000$
B	2000m	1000m	Tak: $1000 + 600 \leq 2000$

- Przy sprawdzaniu pojemności CPU scheduler porównuje requests z allocatable.
- Niskie bieżące zużycie CPU nie zwalnia zadeklarowanych requests.
- Wyliczenie zakłada zgodność pamięci i pozostałych warunków przypisania.

- Przechodzimy od workera do API zamówień: API przyjmuje żądania klientów.
- Pody API są wymieniane, a ich adresy IP mogą się zmieniać.
- Service udostępnia stabilną nazwę DNS i, w domyślnym typie ClusterIP, wirtualny adres IP w klastrze.
- Selektor etykiet wskazuje Pody obsługujące ten Service.

Dlaczego worker nie potrzebował Service?

Worker sam inicjuje połączenie z kolejką. Konsumpcja wiadomości nie wymaga wystawienia punktu dostępu do workera.

```
apiVersion: v1
kind: Service
metadata:
  name: orders-api
spec:
  selector: {app: orders-api}
  ports:
    - port: 80
      targetPort: 8080
```

- Pasuje do Podów z etykietą app=orders-api w tym samym namespace.
 - Klient używa portu 80 Service; aplikacja nasłuchuje na porcie 8080
- Poda.

Domyślny typ: ClusterIP. Nazwa Deploymentu nie decyduje o dopasowaniu.

Droga żądania — typowy Service ClusterIP

Klient → adres Service:80 → wybrany gotowy Pod:8080

Skąd wiadomo, które Pody wybrać?

Kontroler zapisuje adresy i warunki gotowości w obiektach EndpointSlice. kube-proxy lub zastępujący go komponent sieciowy odczytuje te dane i konfiguruje obsługę ruchu Service.

EndpointSlice opisuje endpointy; sam nie przekazuje pakietów. Omawiamy domyślne kierowanie ruchu do gotowych Podów.

Kubernetes — EndpointSlices

Obserwacja	Pierwszy odczyt	Hipoteza
Pending	<code>kubectl describe pod <nazwa></code>	Np. brak CPU; zdarzenia wskażą etap problemu
CrashLoopBackOff	<code>kubectl logs <nazwa> --previous</code>	Kontener kończy pracę i jest restartowany
Service bez gotowych endpointów	Etykiety i EndpointSlice	Błędny selektor lub brak gotowości

Pending obejmuje oczekiwanie na przypisanie oraz przygotowanie kontenerów.
CrashLoopBackOff opisuje oczekiwanie przed kolejnym restartem kontenera.

Microsoft: control plane



API Kubernetes



Zespół: aplikacje

- Zespół odpowiada za manifesty, uprawnienia, obrazy i aktualizacje aplikacji.
- Odpowiedzialność za pojemność i aktualizacje węzłów zależy od trybu usługi i konfiguracji.
- Zarządzany control plane nie naprawia błędnego selektora Service.

Microsoft — AKS core concepts

Więcej replik



Brak miejsca



Pody Pending

- HPA (Horizontal Pod Autoscaler) zmienia liczbę replik na podstawie metryk.
- Autoscaler węzłów może dodać węzeł, jeśli umożliwi to przypisanie oczekujących Podów.
- Wymaga włączenia i odpowiedniej konfiguracji; podlega limitom i dostępności zasobów dostawcy.
- Dodanie węzła wymaga czasu i nie usuwa błędu selektora ani aplikacji.

Microsoft — AKS cluster autoscaler. Obliczenia HPA na wykładzie 4.

```
$ kubectl get pods --show-labels
NAME                READY   STATUS    LABELS
orders-7d-a         1/1     Running   app=orders-api
orders-7d-b         1/1     Running   app=orders-api
$ kubectl get service orders-api -o yaml
spec:
  selector: {app: orders}
$ kubectl get endpointslice -l kubernetes.io/service-name=orders-api
NAME                ADDRESSTYPE   PORTS    ENDPOINTS
orders-api-abcde    IPv4          8080     <none>
```

Diagnoza

Wskaż minimalną zmianę i sposób potwierdzenia jej skuteczności. Czy restart Podów pomoże?

Skrócone dane dydaktyczne; wszystkie obiekty są w tym samym namespace.

- Kubernetes utrzymuje opisany stan przez wiele współpracujących pętli kontroli.
- Deployment zarządza replikami i aktualizacją; scheduler wybiera węzeł, a kubelet organizuje uruchomienie kontenerów.
- Liczba Podów, uruchomienie procesów i gotowość do ruchu to osobne obserwacje.
- Service zapewnia stabilny dostęp; etykiety i gotowość decydują o jego endpointach.
-  Zarządzany klaster nadal wymaga poprawnej konfiguracji aplikacji oraz dostępnej pojemności.

- [Kubernetes — Pods](#)
- [Kubernetes — Components](#)
- [Kubernetes — Controllers](#)
- [Kubernetes — Deployments](#)
- [Kubernetes — Resource management](#)
- [Kubernetes — Service](#)
- [Kubernetes — EndpointSlices](#)
- [Microsoft — AKS concepts](#)
- [Microsoft — AKS cluster autoscaler](#)

Dane przypadków są przykładami dydaktycznymi.