

Računalstvo u oblaku II: orkestracija i DevOps

Predavanje 12 • kolegij Računalne mreže

doc. dr. sc. Sandi Baressi Šegota

Fakultet informatike Sveučilišta Jurja Dobrile u Puli

Sadržaj predavanja

- 1 Orkestracija i Kubernetes
- 2 Infrastruktura kao kod
- 3 CI/CD i DevOps
- 4 Mreže u oblaku
- 5 Sažetak

Orkestracija i Kubernetes

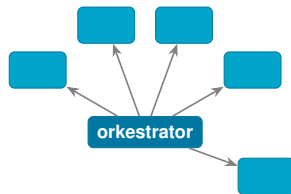
Ishodi učenja

Nakon ovog predavanja moći ćete:

- objasniti potrebu za **orkestracijom** kontejnera;
- opisati arhitekturu i objekte **Kubernetesa**;
- objasniti **infrastrukturu kao kod** (IaC);
- opisati **CI/CD** i DevOps kulturu;
- razumjeti **VPC** i mrežne usluge oblaka.

Zašto orkestracija?

- jedan kontejner je lako, ali **stotine**?
- treba **pokretati**, **skalirati**, **oporavljati**
- ručno — neizvedivo



tko pokreće, skalira, oporavlja *stotine* kontejnera?

Što nudi orkestrator?

- **raspoređivanje** kontejnera na čvorove
- **skaliranje** i uravnoteženje opterećenja
- **samoizlječenje** i postupna ažuriranja

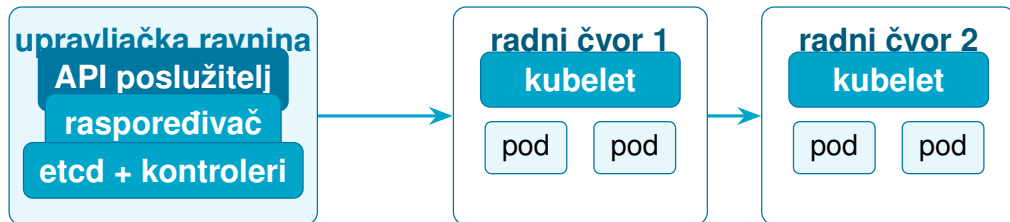
Što je Kubernetes?

Orkestrator kontejnera

Sustav koji **automatizira** postavljanje, skaliranje i upravljanje kontejneriziranim aplikacijama.

Opisujete **željeno stanje**, a Kubernetes ga *održava*.

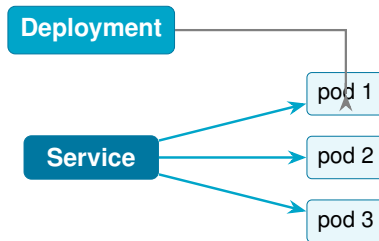
Arhitektura Kubernetesa



- **upravljačka ravnina**: API poslužitelj, raspoređivač, etcd
- **radni čvorovi**: kubelet + podovi

Pod, Deployment, Service

- **Pod**: najmanja jedinica (1+ kontejner)
- **Deployment**: održava broj podova
- **Service**: stabilna pristupna točka



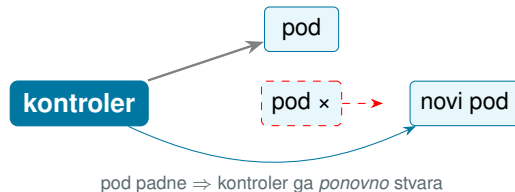
Deployment održava
željeni broj podova;
Service daje stabilnu
pristupnu točku

Skaliranje

- promijeni **broj replika** $\Rightarrow$ K8s dodaje/uklanja podove
- **horizontalno** skaliranje prema opterećenju
- automatski (**HPA**) na temelju metrika

Samoizlječenje

- pod **padne** $\Rightarrow$ automatski novi
- stalno teži **željenom** stanju
- veća **pouzdanost** bez ljudske intervencije



Infrastruktura kao kod

Namespace i organizacija

- **namespace**: logička podjela klastera
- odvaja timove, okruženja (dev/prod)
- olakšava **kvote** i pristupna prava

ConfigMap i Secret

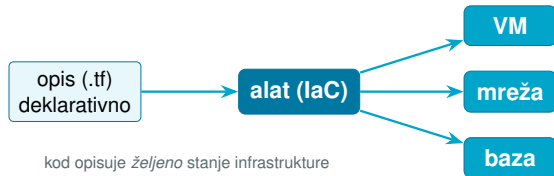
- **ConfigMap**: konfiguracija odvojena od koda
- **Secret**: osjetljivi podaci (lozinke, ključevi)
- aplikacija ih čita bez „zapečene” konfiguracije

Problem ručne konfiguracije

- ručno klikanje $\Rightarrow$ **pogreške** i nedosljednost
- teško **ponoviti** okruženje
- nema **evidencije** promjena

Infrastruktura kao kod (IaC)

- infrastruktura opisana **kodom**
- verzionirano, **ponovljivo**
- alat je *primjenjuje*



Deklarativno vs imperativno

- **deklarativno**: opišeš *što* želiš (željeno stanje)
- **imperativno**: opišeš *korake* kako to postići
- IaC alati su uglavnom **deklarativni**

Terraform

- popularan, **deklarativni** IaC alat
- **plan** pa **apply**: vidiš promjene prije primjene
- vodi **stanje** (state) infrastrukture

Prednosti IaC

- **ponovljiva** i dosljedna okruženja
- **pregled** promjena (code review, povijest)
- brz **oporavak** i repliciranje

GitOps

- **Git** kao izvor istine za infrastrukturu
- promjene kroz **pull request** i pregled
- alat automatski *uskladuje* stvarno i željeno stanje

CI/CD i DevOps

DevOps kultura

- spaja **razvoj** i **operacije**
- **automatizacija** i brza povratna sprega
- česti, mali, **pouzdani** isporuci

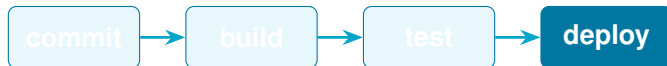
Kontinuirana integracija (CI)

- developeri **često** spajaju kod
- automatski **build** i **testovi**
- rane pogreške $\Rightarrow$ jeftiniji popravak

Kontinuirana isporuka/postavljanje (CD)

- nakon uspješnih testova $\Rightarrow$ **automatsko** postavljanje
- **isporuka**: spremno za produkciju
- **postavljanje**: automatski u produkciju

CI/CD cjevovod



automatizirani cjevovod: od koda do produkcije

- **commit** → build → test → **deploy**
- svaka promjena prolazi *isti* put

Mreže u oblaku

Alati CI/CD

- cjevovodi definirani **kodom** (u repozitoriju)
- **izvršitelji** (runneri) pokreću korake
- integracija s testovima, registrom slika, oblakom

Rollback i sigurne isporuke

- **plavo-zeleno**: dvije okoline, brzo prebacivanje
- **kanarinac**: nova verzija za *dio* korisnika
- problem $\Rightarrow$ brz **povrat** (rollback)

Virtualna privatna mreža (VPC)

- **izolirana** mreža u oblaku
- vlastiti **adresni prostor**
- podjela na **podmreže**



Javne i privatne podmreže

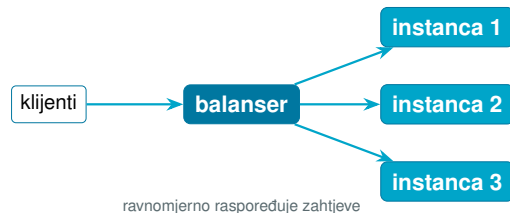
- **javna**: dostupna s Interneta (npr. balanser)
- **privatna**: samo interno (baze, aplikacije)
- **gateway** povezuje s Internetom

Sigurnosne grupe

- **vatrozid** na razini instance/podmreže
- pravila po **portu** i **izvoru**
- princip **najmanje privilegije**

Balanser opterećenja

- raspoređuje promet na **više** instanci
- **provjere zdravlja** (health checks)
- veća **dostupnost** i propusnost



Otkrivanje servisa i DNS

- servisi se pronalaze po **imenu**, ne po IP-u
- interni **DNS** preslikava imena u adrese
- olakšava **dinamičko** skaliranje

CDN i rub mreže

- **CDN** kešira sadržaj *bliže* korisniku
- manja **latencija**, manje opterećenje izvora
- obrada na **rubu** (edge) za brže odgovore

Nadzor i observabilnost

- **metrike**, **logovi** i **tragovi** (traces)
- rano otkrivanje **problema** i uskih grla
- **upozorenja** (alerting) na anomalije

Od oblaka prema HomeLabu

- ista načela vrijede i **kod kuće** u malom
- kontejneri, mreže, reverse proxy — na **vlastitom** poslužitelju
- sljedeće predavanje: **HomeLab**

Sažetak

Ključne poruke

- **orkestracija** (Kubernetes) automatizira rad s mnogo kontejnera
- K8s objekti: **Pod**, **Deployment**, **Service**; skaliranje i samoizlječenje
- **IaC** opisuje infrastrukturu kodom (deklarativno, ponovljivo)
- **CI/CD** automatizira put od koda do produkcije
- **VPC**, podmreže i balanseri organiziraju mrežu u oblaku

Sljedeće predavanje

HomeLab: self-hosting servisa na vlastitoj infrastrukturi.

Pitanja za provjeru

- Zašto je potrebna orkestracija kontejnera?
- Koja je uloga Deploymenta i Servicea u Kubernetesu?
- Što znači „infrastruktura kao kod“?
- Koja je razlika između CI i CD?
- Čemu služe javne i privatne pod mreže u VPC-u?

Hvala na pažnji!

Pitanja i rasprava

doc. dr. sc. Sandi Baressi Šegota • FIPU