Raspodijeljeni sustavi: Vježba na satu - 23. 1. 2026.

Zadatak: Deployment mikroservisne arhitekture na DigitalOcean Droplet VPS

Priprema i spajanje na Droplet VPS kroz SSH protokol

1. Izradite DigitalOcean račun na https://www.digitalocean.com/ ako ga već nemate. Kroz GitHub Student Developer Pack (https://education.github.com/pack) moguće je ostvariti 200$ besplatnih *computational* kredita za DigitalOcean Cloud usluge na godinu dana od aktivacije (https://www.digitalocean.com/github-students). Detaljnije upute na Merlinu.
2. Izradite novi Droplet na DigitalOcean platformi s najmanjim resursima (1 regular vCPU, 512MB RAM, 10GB disk, 0.5TB transfer/month). Odaberite Ubuntu OS i postavite SSH ključ za autentifikaciju. Odaberite **Frankfurt** kao podatkovni centar.
3. **Vaš javni SSH ključ** možete generirati *bash* naredbom `ssh-keygen`. Za potrebe ove vježbe, možete koristiti zadane postavke (*passphrase* može ostati prazan). Ova naredba će, prema zadanim opcijama generirati dvije datoteke: `id_rsa` (privatni ključ) i **`id_rsa.pub`** (javni ključ). Trebat će vam samo **javni ključ**.
4. Kopirajte **sadržaj datoteke** `id_rsa.pub` i zalijepite ga u DigitalOcean sučelje prilikom izrade Dropleta. ***Bash* Naredba**: `cat ~/.ssh/id_rsa.pub` gdje `~` predstavlja *home* direktorij vašeg korisnika. **Zalijepite izrađeni javni ključ u DigitalOcean sučelje.**
5. Nakon izrade Dropleta, zabilježite njegovu javnu **ipv4 adresu**, npr. `207.34.225.119`.
6. Koristeći SSH klijent u terminalu, povežite se na vaš Droplet koristeći naredbu:

```
→ ssh root@<DROPLET_IPV4_ADDRESS>
```

- zamijenite `<DROPLET_IPV4_ADDRESS>` s javnom IPv4 adresom vašeg Dropleta

7. Ako ste sve uspješno postavili, trebali biste biti prijavljeni na vaš DigitalOcean Droplet kao `root` korisnik, a Ubuntu server bi već trebao biti spreman za korištenje.
8. Istražite datotečni sustav Ubuntu servera. Nalazit ćete se u direktoriju `/root`, što je *home* direktorij `root` korisnika. Prebacite se u `/home` direktorij gdje se nalaze *home* direktoriji ostalih korisnika. Novog korisnika na Ubuntu serveru možete dodati naredbom **`adduser <KORISNICKO_IME>`** (npr. `adduser fipu_student`).

> *Need a VPS refresh?* Poslužite se skriptama iz OS-a: *OS4 Rad na virtualnom stroju 1/2*, *OS5 Rad na virtualnom stroju 2/2*. https://github.com/lukablaskovic/FIPU-OS

9. Novi korisnik odmah će dobiti svoj *home* direktorij u `/home/<KORISNICKO_IME>`, a zašto mu odmah ne bi i dali `sudo` privilegije? To možete napraviti naredbom: **`usermod -aG sudo <KORISNICKO_IME>`**.

- Možete se prebaciti u novog korisnika naredbom **`su - <KORISNICKO_IME>`**, a vratiti se natrag u `root` korisnika naredbom `exit`.

10. Omogućit ćemo novom korisniku da se i on pridruži zabavi na našem Dropletu putem javnog SSH ključa. Unutar *home* direktorija novog korisnika, izradite `.ssh` direktorij i `authorized_keys` datoteku:

```
→ mkdir -p /home/<KORISNICKO_IME>/.ssh
→ touch /home/<KORISNICKO_IME>/.ssh/authorized_keys
```

1. Vaš korisnik mora također pokrenuti naredbu `ssh-keygen` na svojem lokalnom računalu kako bi generirao svoj SSH par ključeva. Zamolite ga da vam pošalje sadržaj javnog ključa (`id_rsa.pub`), te ga unesite u datoteku `authorized_keys` unutar `.ssh` direktorija novog korisnika na vašem Dropletu:

```
→ echo "<KORISNICKI_JAVNI_KLJUC>" >>
/home/<KORISNICKO_IME>/.ssh/authorized_keys
```

- ili koristeći CLI editore poput `nano` ili `vim`.

12. Podesite nekoliko dozvola na `.ssh` direktoriju i `authorized_keys` datoteci. Koga zanimaju dozvole, skok na *OS5* lekciju :)

```
→ chmod 700 ~/.ssh
→ chmod 600 ~/.ssh/authorized_keys
```

- korisnik bi se sada trebao moći spojiti na vaš Droplet koristeći svoj SSH ključ naredbom: `ssh <KORISNICKO_IME>@<DROPLET_IPV4_ADDRESS>`

Instalacija Docker enginea na Dropletu i deployment jednostavnog mikroservisa

Docker engine je super jednostavno instalirati na osobna računala, nažalost na serverima je sve nešto kompliciranije. Srećom, Docker tim nam je pripremio službenu dokumentaciju, a mi moramo samo ispravno *copy-pastati* naredbe!

1. Otvorite dokumentaciju: https://docs.docker.com/engine/install/ubuntu/ i scrollajte do sekcije: `Install using the apt repository`.
2. Krenite kopirati i *runnati* jednu po jednu naredbu iz dokumentacije na vašem Dropletu. **Pazite da ne preskočite niti jednu naredbu i *runnajte* ih točnim redoslijedom** kako su navedene.
3. Jednom kada ste završili s instalacijom, provjerite da Docker engine servis radi na Ubuntu serveru naredbom: **`sudo systemctl status docker`**, ako ne radi, onda ga pokrenite naredbom: **`sudo systemctl start docker`**. Provjerite i verziju dockera naredbom: **`docker --version`**.
4. Čestitamo! Uspješno ste instalirali Docker engine na vaš DigitalOcean Droplet VPS!
5. Na vašem računalu, **izradite neki jednostavni kontejnerizirani servis** (npr. običan Python `Hello world` ili `aiohttp web server` iz skripte RS7). Izradite predložak, a

nakon toga pokrenite kontejnere i provjerite da sve radi lokalno.

6. Ako radi lokalno, trebalo bi i na našem Dropletu (nije li to i poanta Dockera?). Ipak, to ne mora biti točno. Droplet ima ograničene resurse, pa ako vaš servis zahtijeva više memorije ili CPU-a nego što Droplet može ponuditi, neće raditi. Također, Docker koji radi na Dropletu radi na arhitekturi servera (vjerojatno x86_64), **tako da moramo izraditi Docker predložak koji je kompatibilan s tom arhitekturom**.

- **Docker će prema zadanim postavkama izraditi predložak za arhitekturu na kojoj se nalazi Docker engine**. Drugim rječima, ako koristite Windows računalo s x86__64 arhitekturom, Docker će izraditi predložak za tu arhitekturu tako da taj predložak možete odmah deployati na vaš Droplet. Međutim, ako koristite Apple M seriju računala (ARM arhitektura), Docker će izraditi predložak za ARM arhitekturu koji neće raditi na našem x86_64 Dropletu. Daleko najozbiljnije rješenje je izrada multi-arch predloška koristeći `buildx` funkcionalnost, ali mi se nećemo za sada time zamarati. Koga zanima link: https://docs.docker.com/build/building/multi-platform/

7. Ako koristite Apple računalo s M čipom (ili neko drugo računalo s *ARM arch*), morate specificirati arhitekturu prilikom izrade Docker predloška.

```
→ docker buildx build \
--platform linux/amd64 \
-t image:tag .
```

- hint: oznaka `\` na kraju linija nije dio CLI naredbe, već samo označava da se naredba nastavlja u sljedećoj liniji

Ako ne koristite, jednostavno možete izgraditi predložak kao i obično koristeći `docker build -t image:tag .`

8. Sljedeće pitanje je: kako ćemo prebaciti naš Docker predložak s lokalnog računala na naš Droplet? Možemo kroz `docker save` / `docker load` naredbe koje će izvesti/uvesti predložak kroz `.tar` datoteku, a na sam Droplet ćemo tu datoteku prebaciti kroz `scp` naredbu (secure copy). Ipak, mi ćemo koristiti jednostavniji pristup preko **DockerHub-a**.
9. **Otvorite DockerHub račun** na https://hub.docker.com/ ako ga već nemate te izradite novi **DockerHub repozitorij** za vaš Docker predložak. Odaberite javni repozitorij i dajte mu naziv prema vašem predlošku, DockerHub će automatski na predložak zalijepiti vaš DockerHub korisnički račun kao *namespace* (npr. `fipu_studeni/my_docker_image`).
10. Na vašem računalu (ne Dropletu), prijavite se u DockerHub kroz terminal naredbom: **`docker login`** i unesite vaše DockerHub korisničko ime i lozinku - slijedite upute za prijavu.
11. Tagirajte vaš lokalni Docker predložak s imenom vašeg DockerHub repozitorija koristeći naredbu:

```
→ docker tag local_image:tag your_dockerhub_username/repository_name:tag
```

- npr. ako vam se predložak zove: `my_app:latest` i vaš DockerHub korisnički račun je `fipu_student`, a repozitorij se zove `my_docker_image`, naredba bi izgledala ovako:

```
→ docker tag my_app:latest fipu_student/my_docker_image:latest
```

- dobra praksa je **izjednačiti naziv lokalnog predloška i repozitorija**

12. *Pushajte* vaš Docker predložak na DockerHub koristeći naredbu: **`docker push your_dockerhub_username/repository_name:tag`**, npr. `docker push fipu_student/my_docker_image:latest`.
13. Sada se vratite na vaš Droplet putem SSH-a i *pullajte* vaš Docker predložak s DockerHub-a koristeći naredbu: **`docker pull your_dockerhub_username/repository_name:tag`**.
14. Nakon što je predložak uspješno preuzet na vaš Droplet, možete pokrenuti kontejner koristeći `docker run` naredbu kao i do sada.
15. Provjerite da li vaš servis radi na vašem Dropletu. Ako je to web servis, možete koristiti `curl` naredbu ili otvoriti web preglednik i otići na `http://<DROPLET_IPV4_ADDRESS>:<PORT>` gdje je `<PORT>` port na kojem vaš servis sluša. **Pripazite da *mappirate* port ako se radi o mikroservisu s web poslužiteljem!**
16. Čestitamo! Uspješno ste *deployali* vaš Docker mikroservis na DigitalOcean Droplet VPS koristeći DockerHub kao posrednika za prijenos predloška.

Deployment mikroservisne arhitekture bez load balancera

Sada kada znate kako *deployati* jedan mikroservis na vaš Droplet, **vrijeme je da *deployate* cijelu mikroservisnu arhitekturu s više servisa**. Za sada nećemo koristiti *load balancer*, već ćemo svaki mikroservis *deployati* na zasebni port.

1. **Pripremite mikroservisnu arhitekturu** s barem tri mikroservisa (npr. korisnički servis, proizvodni servis, narudžbeni servis). **Svaki mikroservis treba imati svoj Docker predložak koji radi lokalno kada se pokrene**.
2. **Izradite DockerHub repozitorij za svaki mikroservis** i *pushajte* njihove predloške na DockerHub koristeći korake opisane u prethodnom dijelu vodiča.
3. Pripremite *root* direktorij i Docker Compose datoteku (`docker-compose.yml`) na vašem lokalnom računalu koja definira sve mikroservise i njihove portove. Primjer u repozitoriju kolegija - skripta RS7.
4. Nakon što ste izradili `docker-compose.yml` datoteku, prebacite je na vaš Droplet koristeći `scp` naredbu:

```
→ scp <PATH_TO_DOCKER_COMPOSE_FILE> root@<DROPLET_IPV4_ADDRESS>:
<DROPLET_DESTINATION_PATH>
```

- *Primjer:* `scp ./docker-compose.yml root@123.456.78.90:/home/fipu_student/`

5. Spojite se na vaš Droplet putem SSH-a i navigirajte do direktorija gdje ste prebacili `docker-compose.yml` datoteku.
6. Pokrenite mikroservisnu arhitekturu koristeći Docker Compose naredbu **`docker compose up -d`**. Zastavica `-d` pokreće servise u *detached* modu (u pozadini) - na taj način nećete "zauzeti" vaš terminal, niti izgubiti SSH vezu.
7. Provjerite da su svi mikroservisi pokrenuti koristeći naredbu: **`docker compose ps`** ili

`docker ps`.

8. Testirajte svaki mikroservis pristupajući im putem njihovih mapiranih portova na vašem Dropletu (npr. `http://<DROPLET_IPV4_ADDRESS>:<SERVICE_PORT>`).
9. Čestitamo! Uspješno ste deployali mikroservisnu arhitekturu na DigitalOcean Droplet VPS.
10. Aplikaciju možete zaustaviti naredbom: `docker compose down` unutar direktorija gdje se nalazi `docker-compose.yml` datoteka.

Ako ste zapeli, možete pogledati primjer `e-commerce-app` aplikacije iz RS7 skripte koja koristi 4 mikroservisa i *frontend* Vue.js aplikaciju s korespondirajućim `Dockerfile` datotekama i Docker Compose datotekom.