Softverski definisane mreže (SDN)

- U tradicionalnim mrežama ravan podataka i kontrolna ravan implementirani su na svakom uređaju
- SDN kontrolna ravan je razdvojena od ravni podataka i *logički centralizovana*
- Inicijalno motivisan problemima koji prate virtuelizaciju servera u "big data" eri
  - Mrežni administratori moraju da osiguraju da je VLAN kojem pripada VM dodijeljen onom portu sviča na koji je povezan fizički server.
  - Potrebno je rekonigurisati VLAN svaki put kada se VM pomjeri na drugi server.
- Drugi motiv: korišćenje pametnih telefona, tableta i *notebook* računara za pristup resursima kompanijskih mreža
  - Administratori moraju biti u stanju da rapidno odgovore na dinamičke promjene inteziteta saobraćaja, QoS zahtjeve i sigurnosne zahtjeve.
  - Upravljanje tradicionalnim mrežama je veoma kompleksno i sporo.

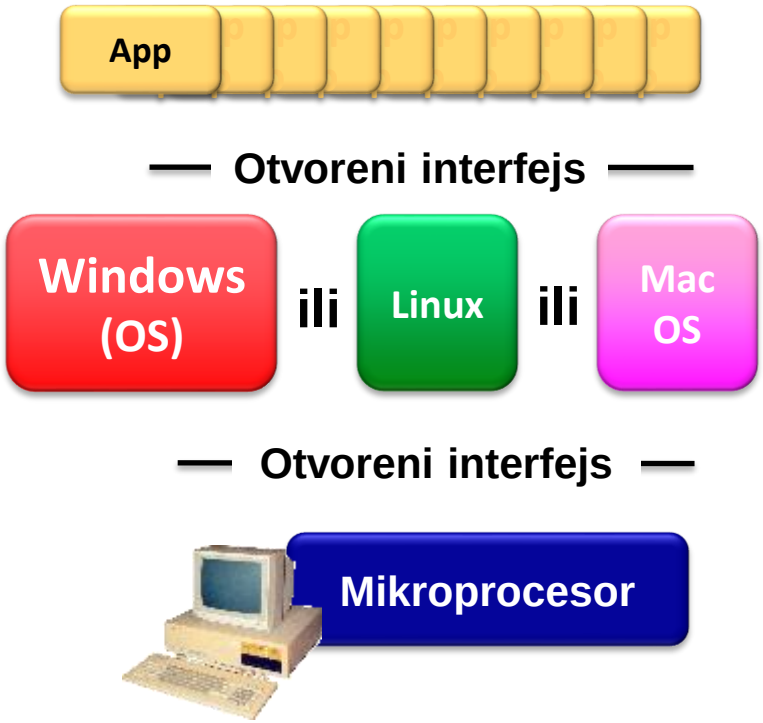
Softverski definisane mreže (SDN)

Zašto logički centralizovana kontrolna ravan?

- Jednostavnije upravljanje mrežom: rijeđe su greške u konfiguraciji, veća fleksibilnost u kontroli saobraćajnih tokova
- API između kontrolne ravni i ravni podataka omogućava "programiranje" rutera
  - Centralizovano "programiranje" je lakše: tabele prosleđivanja se računaju centralizovano i distribuiraju.
  - Distribuirano "programiranje" je teže: tabele prosleđivanja računaju distribuirani algoritmi (protokoli) koji su implementirani na svakom ruteru .
- Otvorena implementacija kontrolne ravni

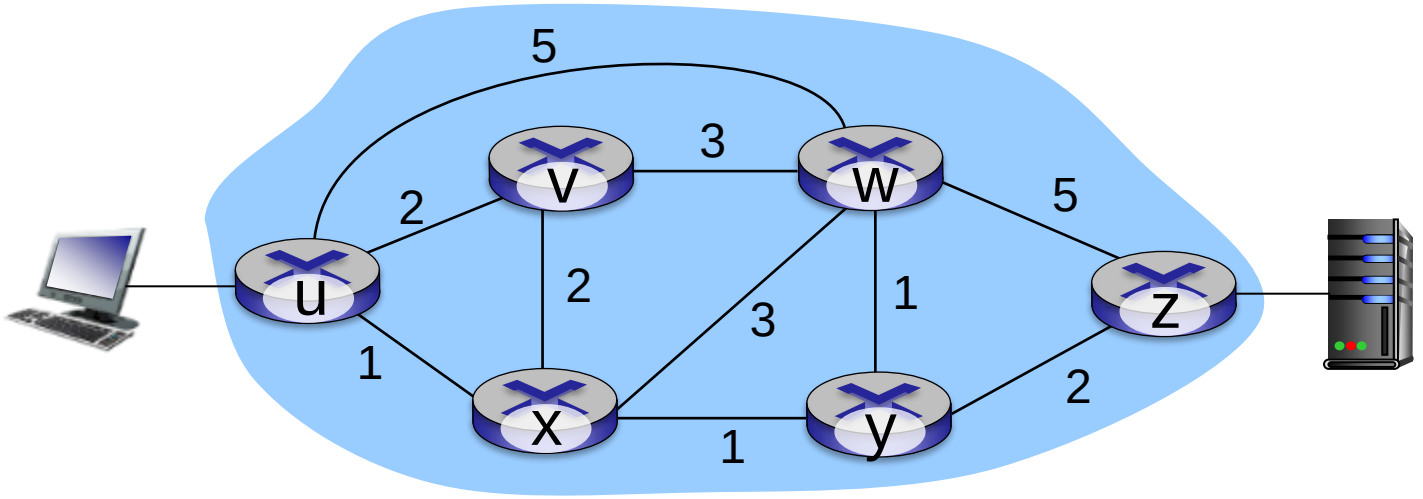
Analogija: evolucija od mejnfrejma do PC računara

Vertikalno integrisan
Zatvorenog koda
Spore inovacije
Mala industrija



Horizontalno integrisan
Otvoreni interfejsi
Brze inovacije
Ogromna industrija

Inženjering saobraćaja: kompleksnost distribuiranog rutiranja

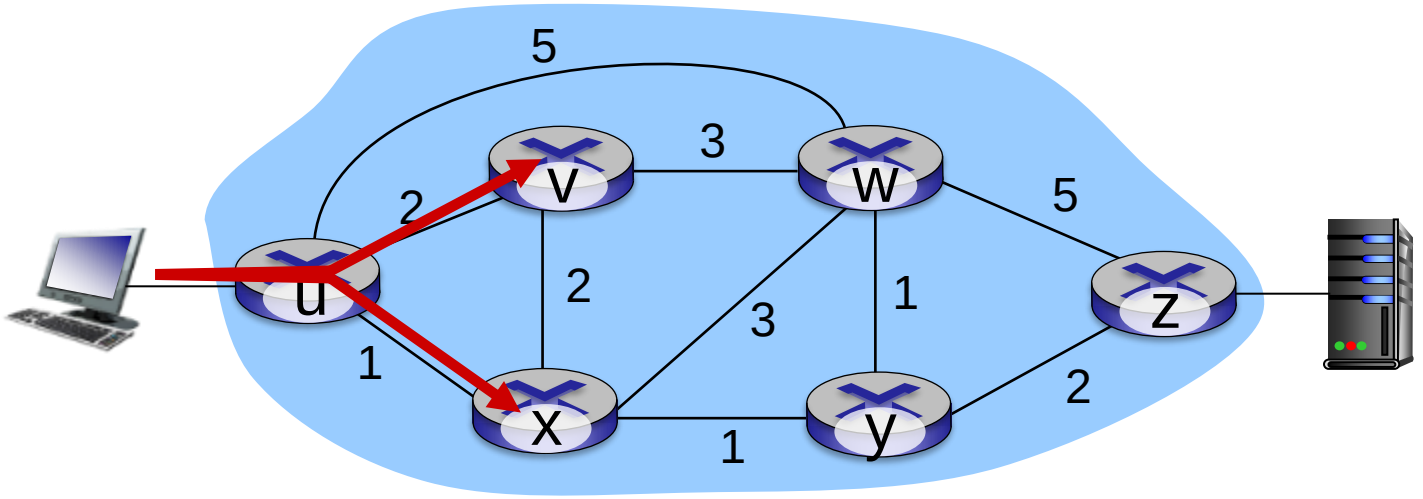


P: Šta ako mrežni operator želi da se saobraćaj od *u* do *z* prenosi rutom *uvwz*, a saobraćaj od *x* do *z* rutom?

O: moramo definisati težinske faktore linkova tako da allgoritam rutiranja bira baš te rute (ili nam je potreban novi algoritam rutiranja)!

Težinski faktori linkova su jedini vid kontrole: loše!

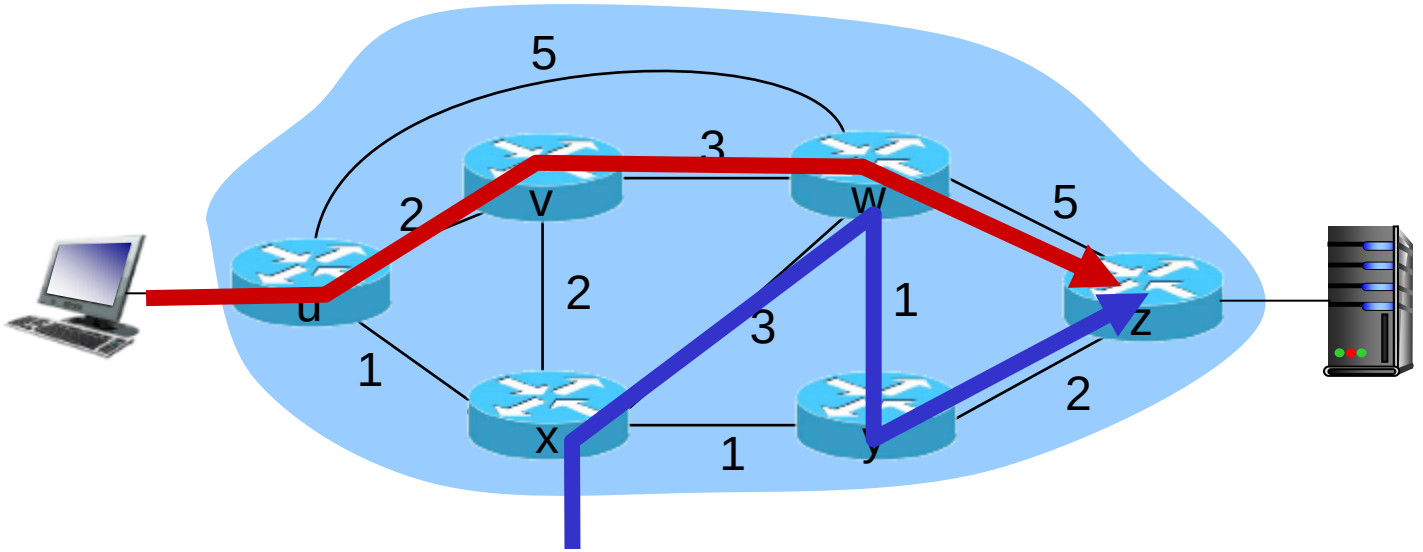
Inženjering saobraćaja: teško



P: šta ako mrežni operator želi da podijeli saobraćaj između *u* i *z* preko dvije rute: uvwz *i uxyz* (balansiranje saobraćaja)?

O: nije moguće to uraditi (potreban novi algoritam rutiranja)

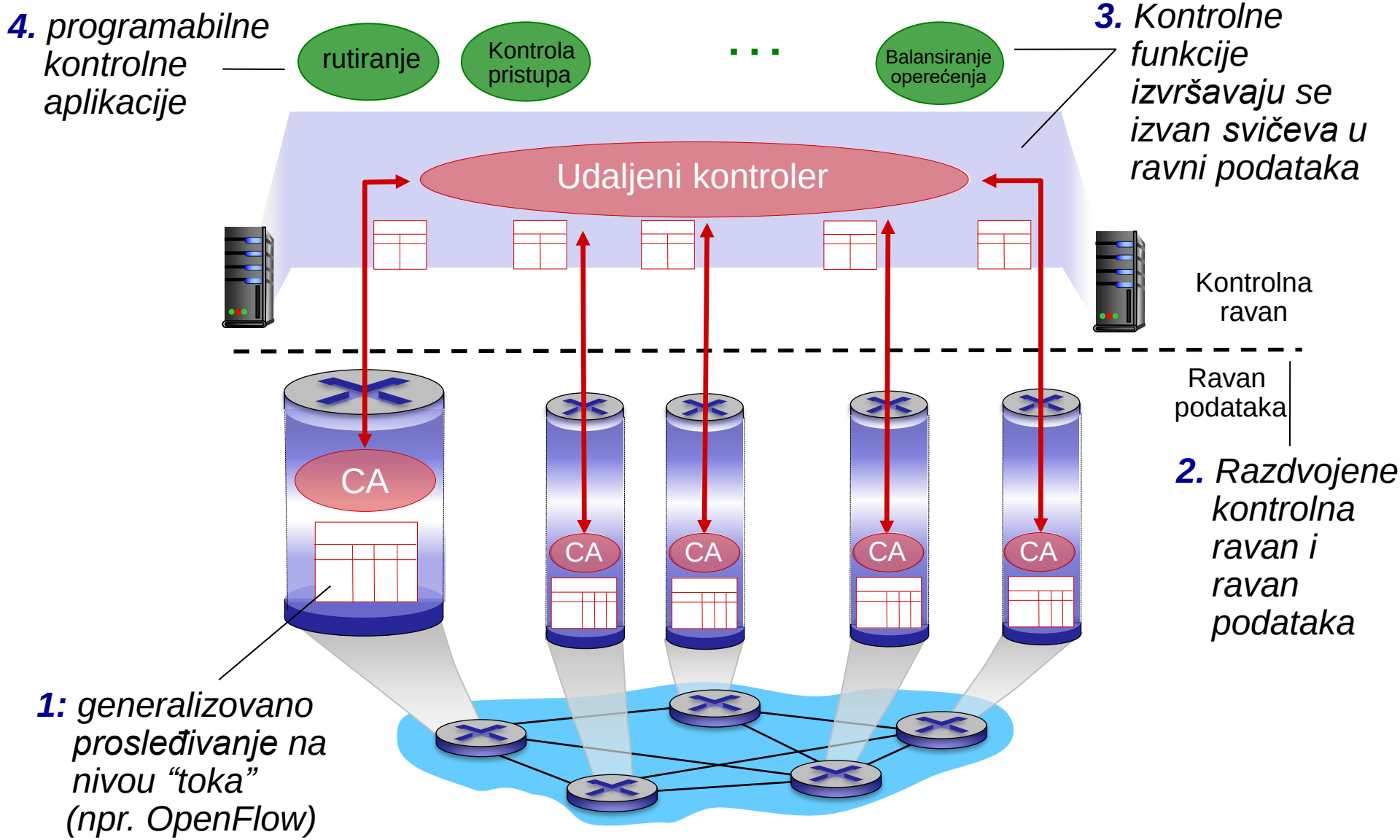
Inženjering saobraćaja: teško



P: Šta ako *w* želi da rutira plavi i crveni saobraćaj na različite načine?

O: Nije moguće (sa rutiranjem na osnovu destinacione IP adrese, *link-state* i *distance-vector* rutiranjem)

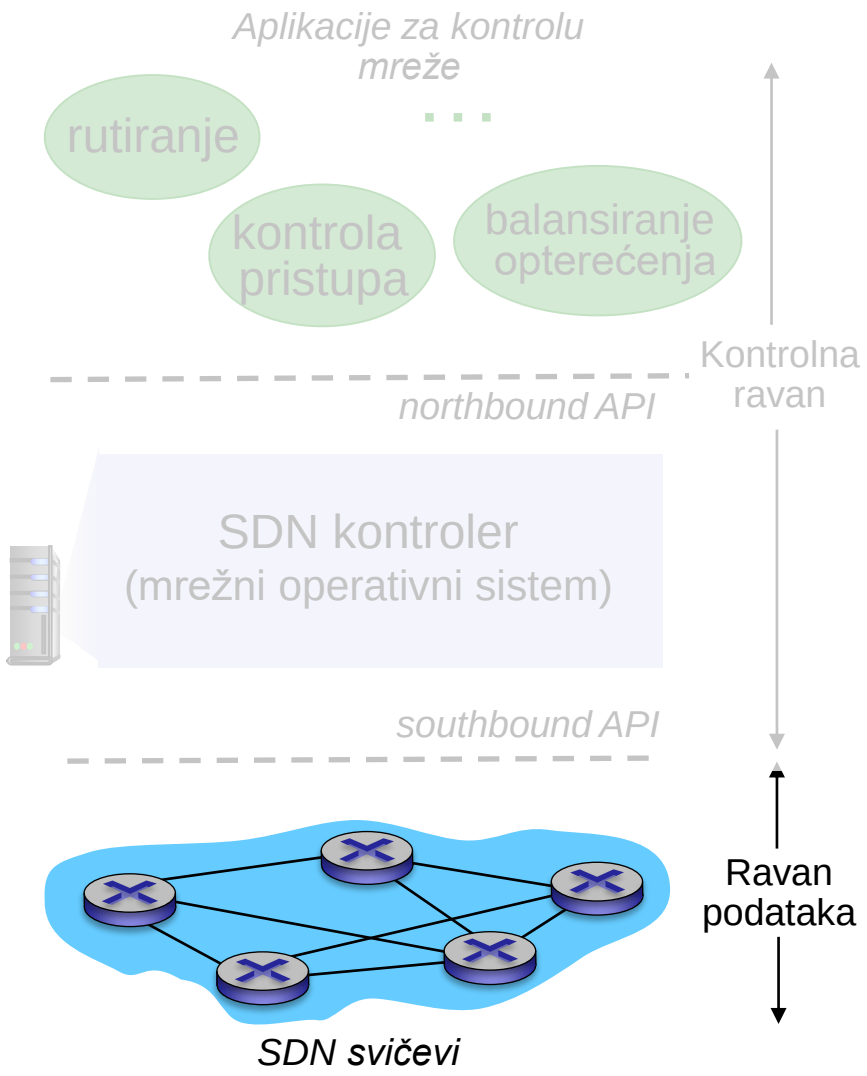
Softverski definisano umrežavanje (SDN)



SDN perspektiva: svičevi u ravni podataka

Ravan podataka

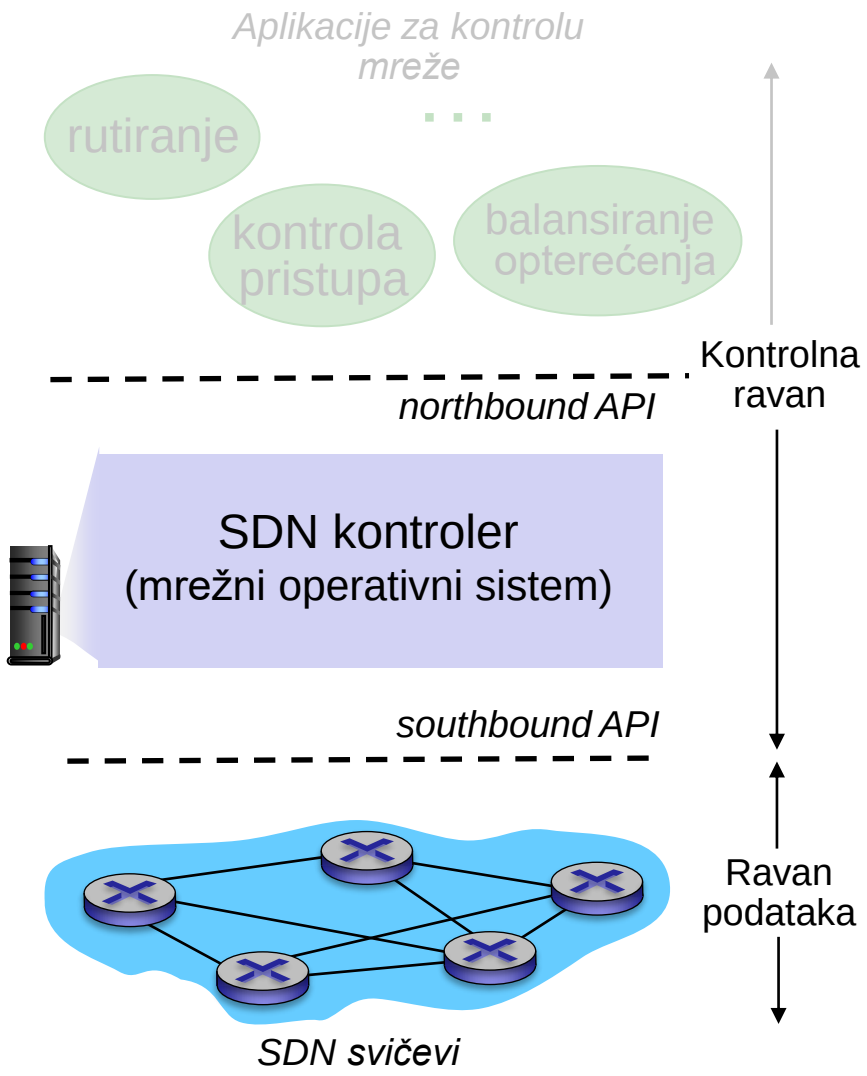
- Brzi, jednostavni svičevi, implementirani u hardveru
- Prosleđivanje se vrši na osnovu tabele tokova koju instalira kontroler
- Tabele tokova generalizuju tabele prosleđivanja
- API za kontrolu tabela tokova (npr. OpenFlow)
  - Definiše šta može da se kontroliše a šta ne može
- Protokol za komunikaciju sa kontrolerom (npr. OpenFlow)



SDN perspektiva: SDN kontroler

SDN kontroler (mrežni OS):

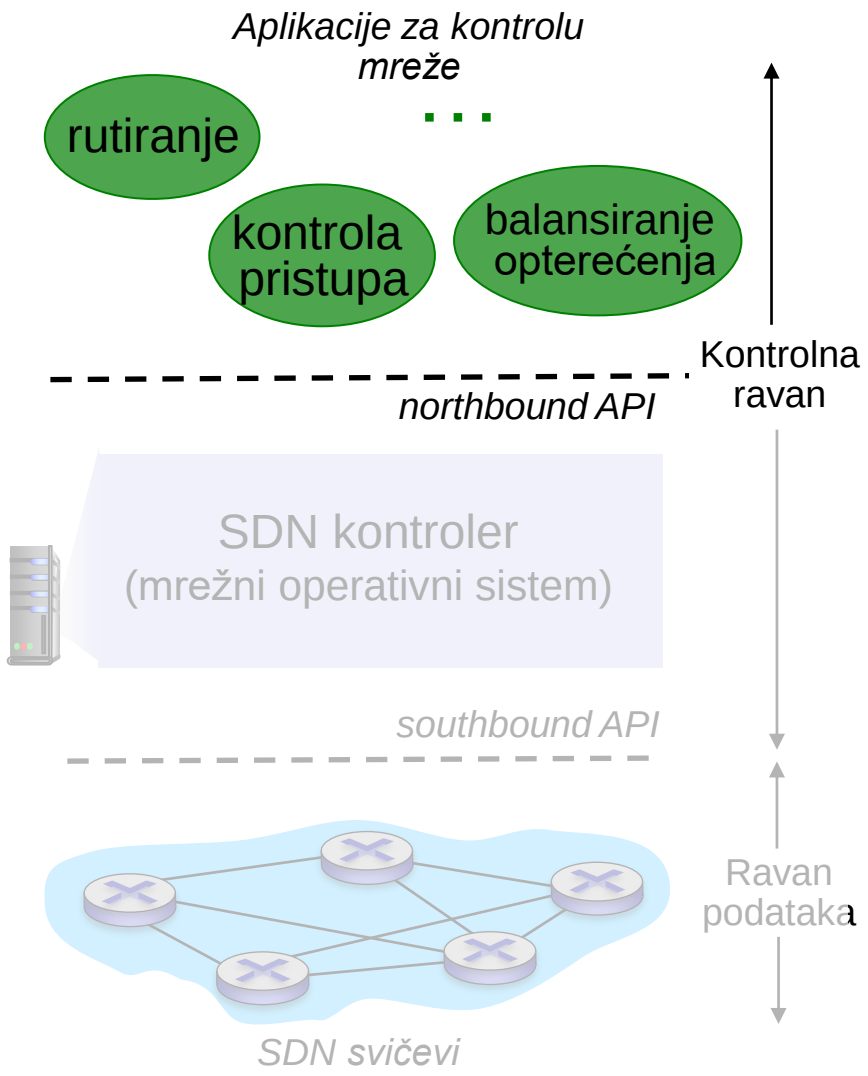
- Održava informacije o stanju mreže
- Interakcija sa mrežnim kontrolnim aplikacijama putem *northbound* API-ja
- Interakcija sa mrežnim svičevima preko *southbound* API-ja
- Implementiran kao distribuiran sistem zbog performansi, skalabilnosti, robustnosti na greške i kvarove



SDN perspektiva: kontrolne aplikacije

Mrežne kontrolne aplikacije:

- "Mozak" upravljanja: implementiraju kontrolne funkcije koristeći API koji pruža SDN kontroler
- Može ih kreirati i treća strana: različita od proizvođača SDN svičeva i kontrolera

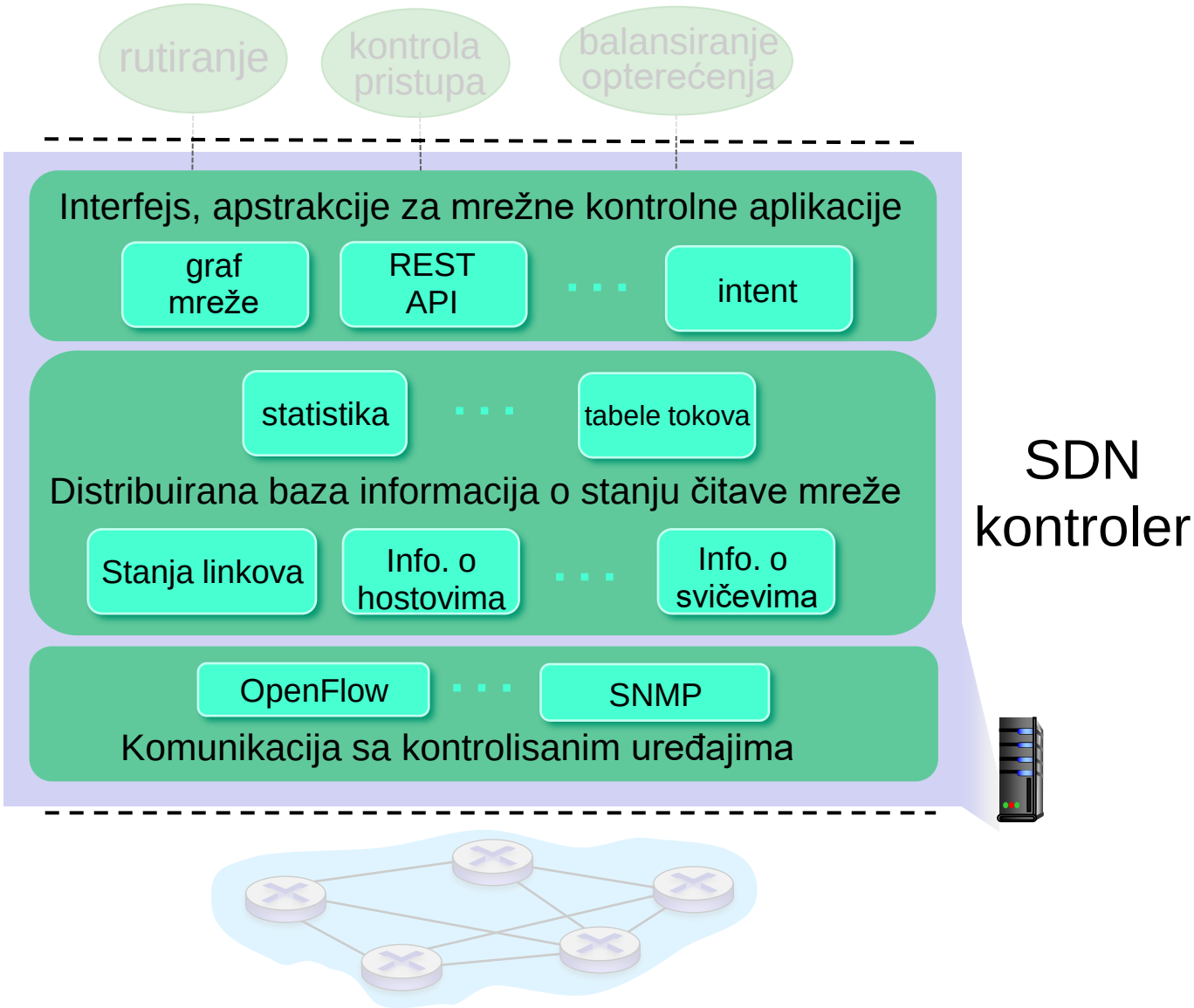


Komponente SDN kontrolera

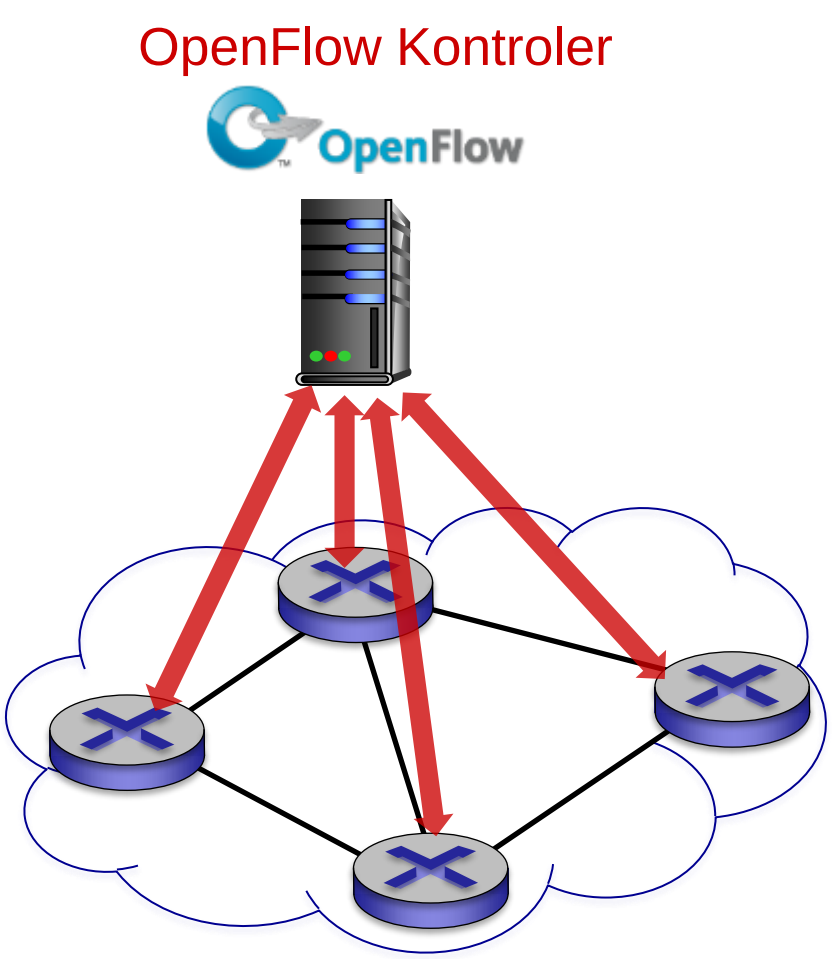
Interfejs prema mrežnim kontrolnim aplikacijama: API za apstrakciju

Sloj za upravljanje informacijama o stanju mreže: stanje mrežnih linkova, svičeva, servisa: *distribuirana baza podataka*

Nivo komunikacije: komunikacija između SDN kontrolera i kontrolisanih svičeva



OpenFlow protokol

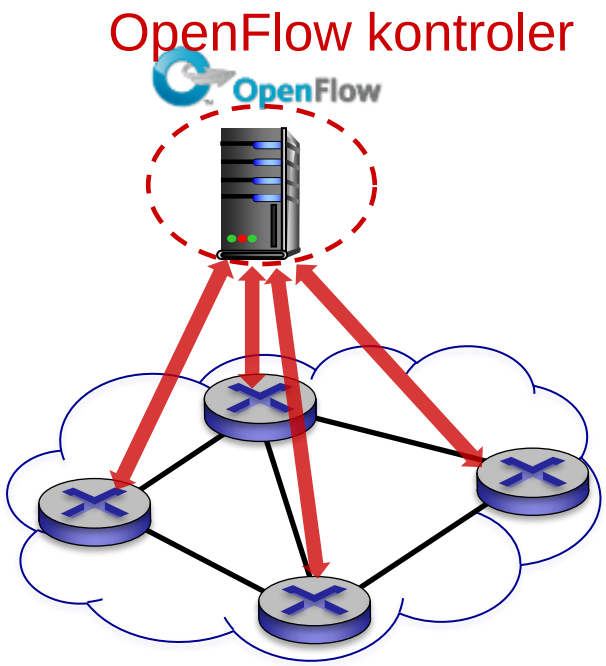


- Koristi se za komunikaciju između kontrolera i svičeva
- TCP se koristi za razmjenu poruka
  - opciona enkripcija
- Tri klase OpenFlow poruka:
  - kontroler-svič
  - Asinhrone (svič- kontroler)
  - simetrične

OpenFlow: kontroler-svič poruke

Ključne kontroler-svič poruke

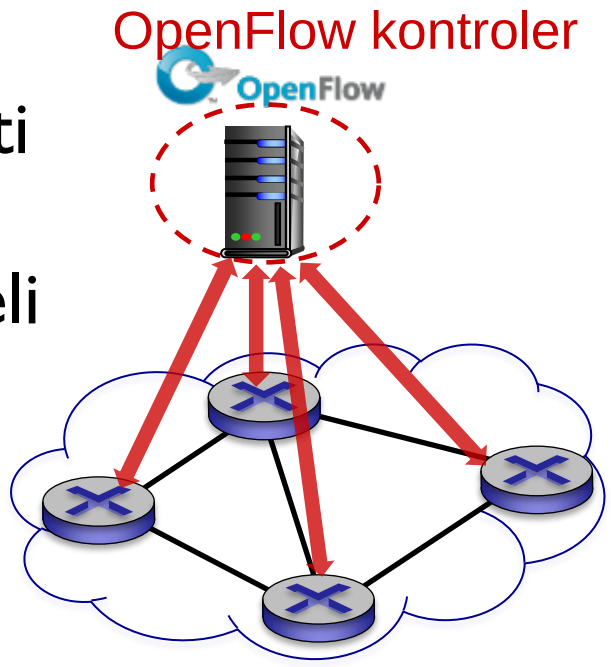
- *features:* kontroler provjerava svojstva sviča, svič odgovara
- *configure:* kontroler provjerava/podešava konfiguracione parametre sviča
- *modify-state:* dodavanje, brisanje i modifikovanje zapisa u OpenFlow tabeli tokova
- *packet-out:* kontroler može poslati paket preko određenog porta sviča



OpenFlow: svič-kontroler poruke

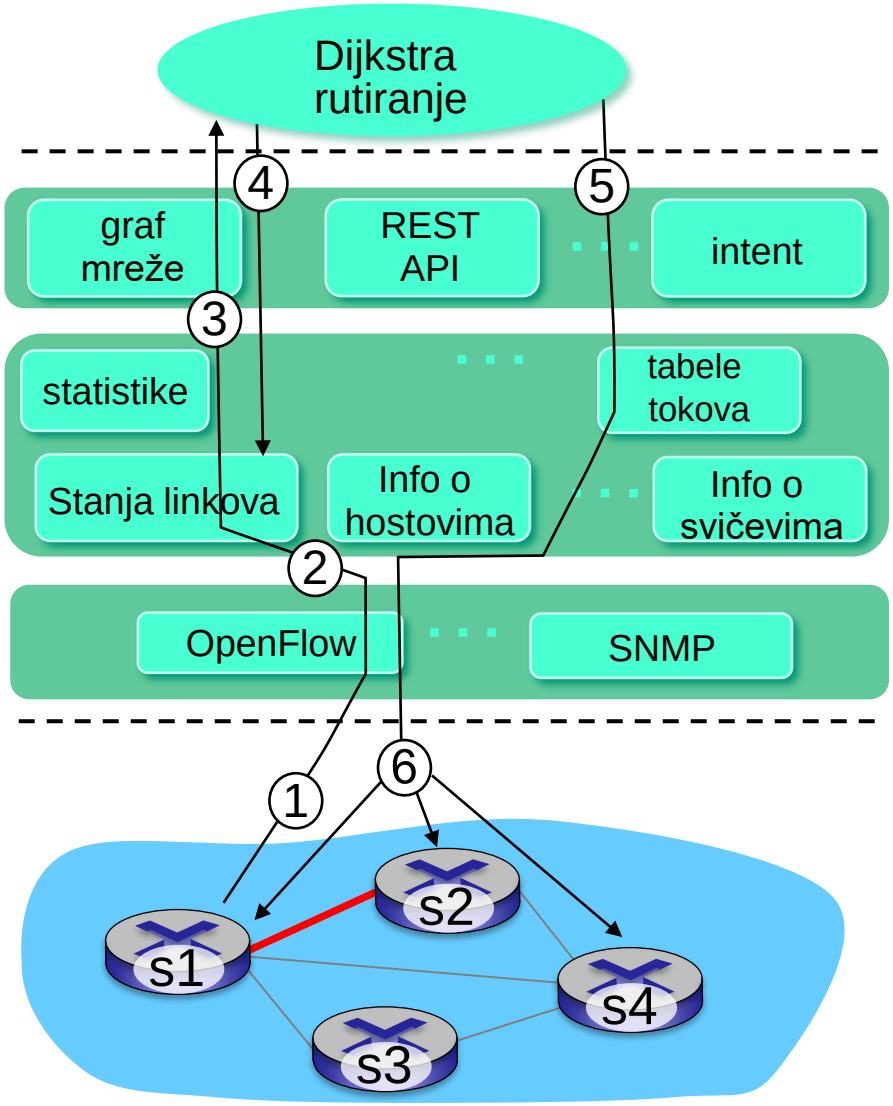
Ključne svič-kontroler poruke

- *packet-in:* prenos paketa do kontrolera. Kontroler može poslati *packet-out* poruku kao odgovor.
- *flow-removed:* brisanje zapisa u tabeli tokova sviča.
- *port status:* informiše kontroler o promjeni statusa porta.



Na sreću, mrežni operatori ne "programiraju" svičeve tako što kreiraju i šalju OpenFlow poruke. Umjesto toga koristi se apstraktni API kontrolera.

SDN: interakcija između kontrolne i ravni podataka



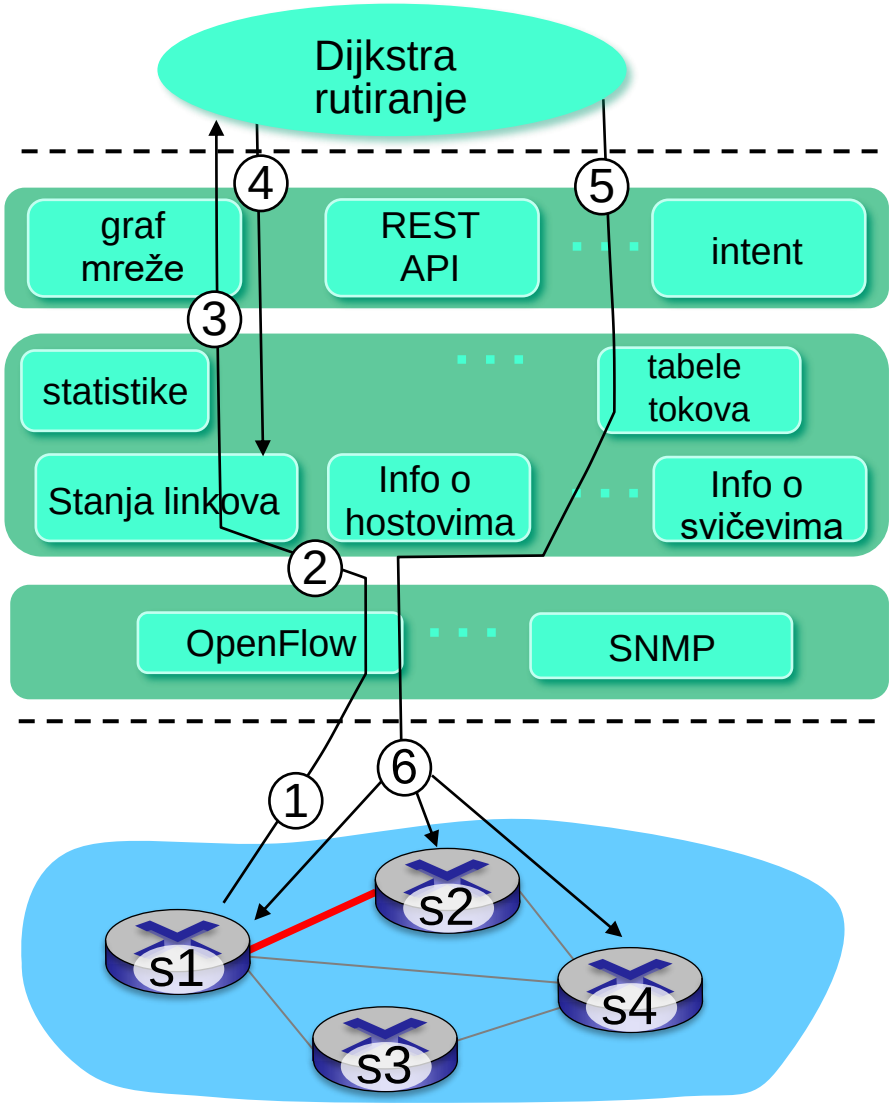
① S1, po detekciji otkaza linka, šalje OpenFlow *port-status* poruku kontroleru

② SDN kontroler prima OpenFlow poruku, ažurira informacije o stanju linkova

③ Dijkstra aplikacija za rutiranje prethodno se registrovala da prima notifikacije o promjenama statusa linka. Kontroler šalje notifikaciju.

④ Dijkstra algoritam rutiranja pristupa informacijama o mrežnom grafu i stanju linkova na kontroleru, računa nove rute.

SDN: interakcija između kontrolne i ravni podataka



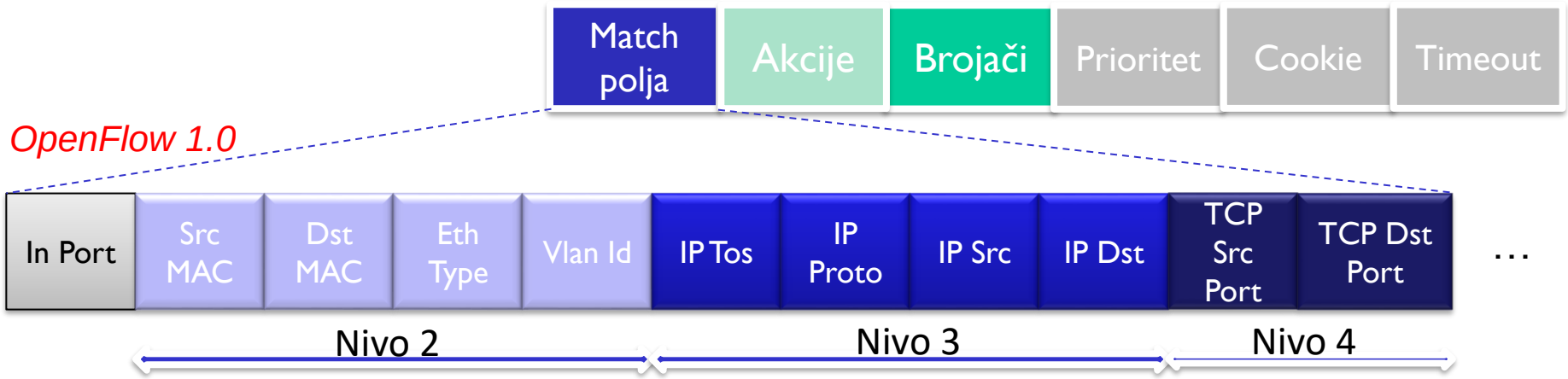
⑤ Aplikacija za rutiranje intereaguje sa komponentom kontrolera koja računa nove tabele tokova

⑥ Kontroler koristi OpenFlow protokol da instalira nove tabele tokova na svičevima

OpenFlow tabela tokova

- Svaki od zapisa sadrži:
  - *Match polja* - služe za identifikaciju toka
  - *Akcije* – definišu postupak obrade paketa
  - *Brojači* – statistike na nivou toka, porta itd.
  - *Prioritet* – relativni prioritet zapisa
  - *Cookie* – ne koristi se prilikom obrade paketa ali olakšava upravljanje i filtriranje *zapisa*
  - *Timeout* – rok važenja zapisa

Nove OpenFlow verzije omogućavaju korišćenje proizvoljnih match polja!



OpenFlow tabela tokova

- Podržane akcije:
  - *Output* – prosleđivanje na izlazni interfejs sviča
  - *Set-queue* – određuje bafer u koji će se paket smjestiti
  - *Drop* – odbacuje paket
  - *Group* – upućuje paket u *group* tabelu
  - *Go-To* – upućuje paket u sledeću tabelu tokova
  - *Push-Tag/Pop-Tag* – dodaje ili uklanja polje iz zaglavlja (npr. VLAN, MPLS)
  - *Set-Field* – modifikovanje polja u zaglavlju paketa
  - *Change-TTL* – modifikuje *TTL* polje u IPv4 i *HopLimit* polje u IPv6 paketima
- Svič može da ima više tabela tokova koje su lančano uvezane
- OpenFlow omogućava definisanje seta akcija koje će se izvršiti nad paketom prilikom njegove obrade kroz lanac tabela tokova
- Pored tabela tokova, svič može da koristi *group* i *meter* tabele
  - *Group tabela* trigeruje jednu ili više akcija koje se izvršavaju nad više tokova
  - *Meter tabela* trigeruje različite akcije koje se odnose na performanse toka (npr. Oblikovanje saobrćaja)

OpenFlow kontroleri

Open-source rešenja

- Onos – Java
- OpenDaylight – Java
- Floodlight – Java
- Ruby – Python
- PoX – Python
- NoX – C++
- Trema – C/Ruby
- Maestro - Java

Komercijalne rešenja

- Uglavnom bazirana na *open-source* okruženjima

Virtuelizacija mrežnih funkcija (NFV)

Virtuelizacija

- Pretvara fizičke resurse u logičke, ili virtuelne, resurse
- Omogućava korisnicima i aplikacijama iznad nivoa apstrakcije da upravljaju i koriste "logičke" resurse bez ulaza u detalje o stvarnim fizičkim resursima

NFV

- Razdvajanje mrežnih funkcija od hardvera, i stvaranje uslova za izvršavanje mrežnih funkcija u formi softverskih komponenti na serverima opšte namjene, svičevima, storidž uređajima ili *cloud*-u
- Virtuelne mrežne funkcije (eng. *Virtual Network Functions*–*VNFs*) se mogu sastojati od jedne ili više virtuelnih mašina na kojima se izvršavaju različiti softveri i procesi koji zamjenjuju ulogu određenih hardverskih uređaja
- Obično se više VNF-ova se koristi u sekvenci kako bi se korisnicima pružili servisi od interesa

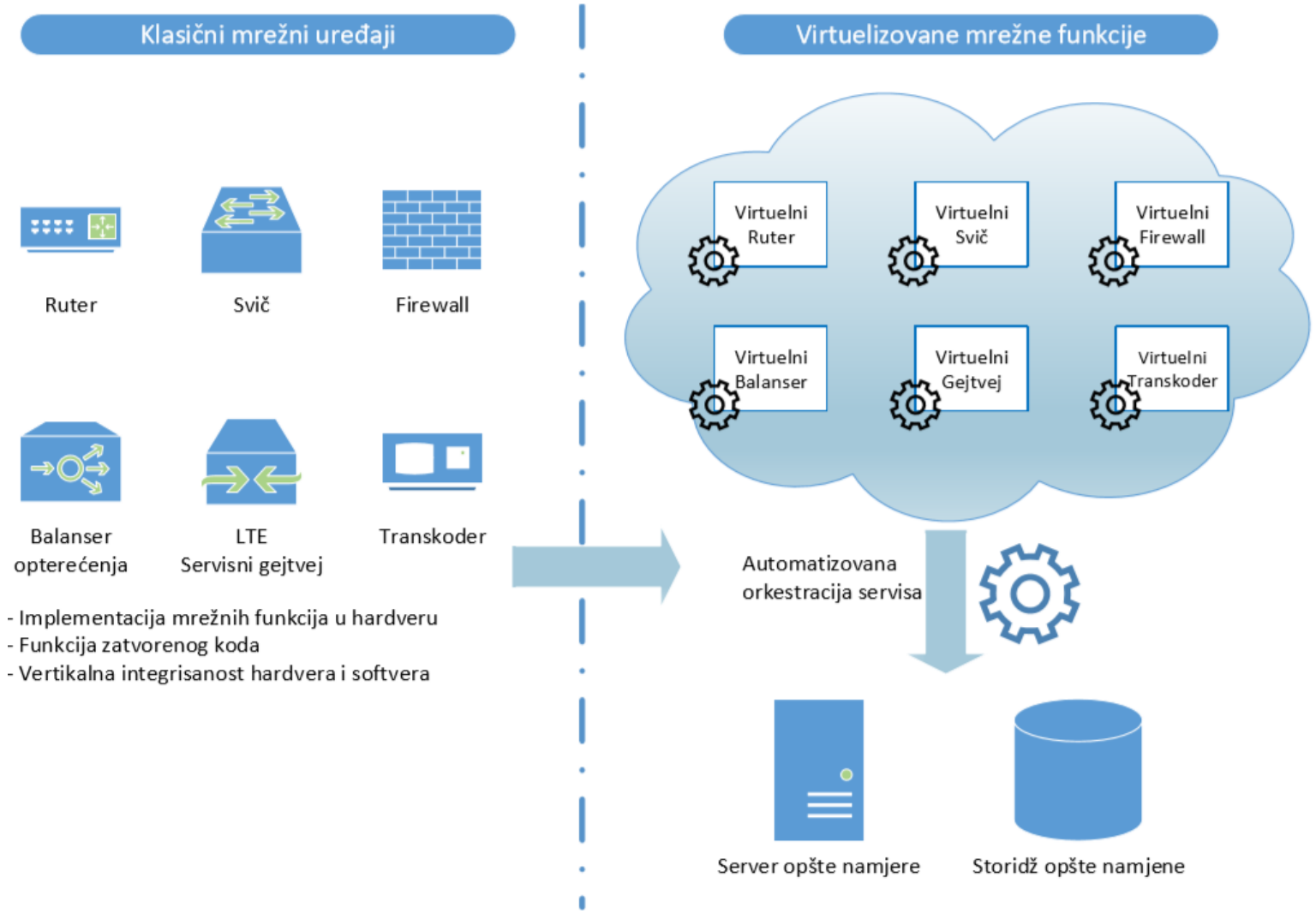
Mane hardverske implementacije mrežnih funkcija

- Današnje mreže sadrže veliki broj hardverskih uređaja na kojima su vertikalno integrisane različite mrežne funkcije (npr. rutiranje, balansiranje opterećenja, dubinska inspekcija paketa, *firewall*)

Mane tradicionalnog pristupa

- Softver koji je implementiran na ovim uređajima je zatvorenog koda i nudi vrlo ograničene mogućnosti za udaljenu rekonfiguraciju
- Novi servisi zahtijevaju nove tipove hardverskih uređaja
- Novi hardver znači nove kapitalne troškove
- Kompleksna integracija i upravljanje velikim brojem uređaja
- Jednom kada se saobraćajno opterećenje približi mrežnom kapacitetu (npr. 70% iskorišćenosti), vrši se *upgrade* opreme kako bi mreža mogla da odgovori na buduće zahtjeve
- Bilo kakva nepredviđena distorzija saobraćaja može izazvati probleme u funkcionisanju mreže

NFV transformacija



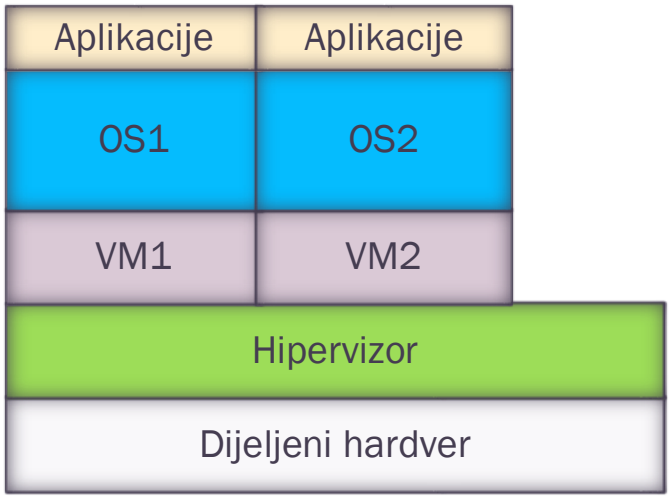
Virtuelne mašine (VM) - podsjećanje

- Tradicionalno, aplikacije su se izvršavale direktno na OS-u računaru.
- Svaki računar izvršavao je samo jedan OS.
- Virtualizacija je omogućila da se na jednom računaru izvršava više različitih OS-ova ili više instanci istog OS-a.
- Rešenje koje omogućava virtualizaciju je poznato pod nazivom *VM monitor* (VMM) ili *hipervizor.*
- Hipervizor upravlja fizičkim resursima i omogućava da više VM-ova koegzistira na jednom fizičkom serveru i dijeli njegove resurse.
- Broj VM-ova koji se mogu hostovati na jednom fizičkom serveru (ili računaru) definisan je odnosom konsolidacije.
  - Npr. ukoliko host može da podrži 6 VM-ova, onda je njegov odnos konsolidacije 6:1.
- Što je manje fizičkih servera u *data* centrima, to su manji zahtjevi u pogledu hlađenja i prostora, i manja je potrošnja energije.
- VM-ovi se mogu migrirati sa jedne fizičke mašine na drugu u slučaju kvarova ili za potrebe balansiranja opterećenja.

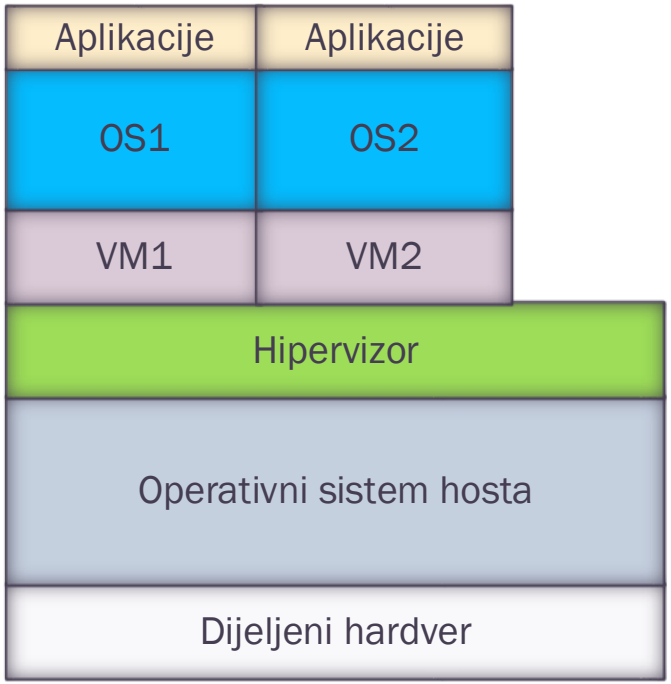
Virtuelne mašine (VM) - podsjećanje

- Dva tipa VM hipervizora: Tip1 i Tip 2
- *Hivervizor tipa 1*: softverki sloj instaliran direktno na hardveru (analogno OS-u koji se učitava prilikom startovanja računara)
  - Čim se instalira i konfiguriše, računar je u stranju da podrži više VM-ova
  - Primjer: VMware ESXi, Microsoft Hyper-V i razne otvorene Xen varijante
  - Direktno upravlja fizičkim resursima hosta
  - Troši manje resursa od hipervizora tipa II, pa može da podrži više VM-ova
- *Hipervizor tipa II*: instalira se na OS-u računara/servera
  - Jednostavniji za upotrebu ali slabijih performansi
  - Instalira se kao standardna aplikacija na Windows ili Unix/Linux OS-u
  - Primjer: VMware Workstation i Oracle VM Virtual Box
  - OS vrši interakciju sa hardverom u ime hipervizora

Virtuelne mašine (VM) - podsjećanje

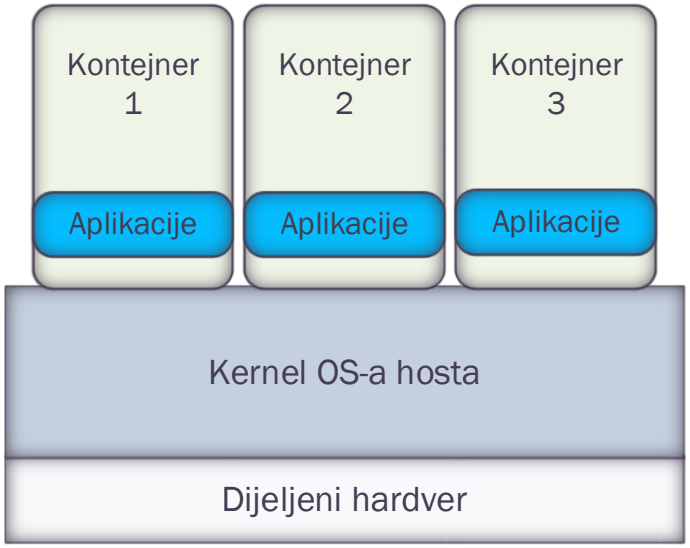


Hipervizor tip 1



Hipervizor tip 2

Virtuelizacija kontejnera

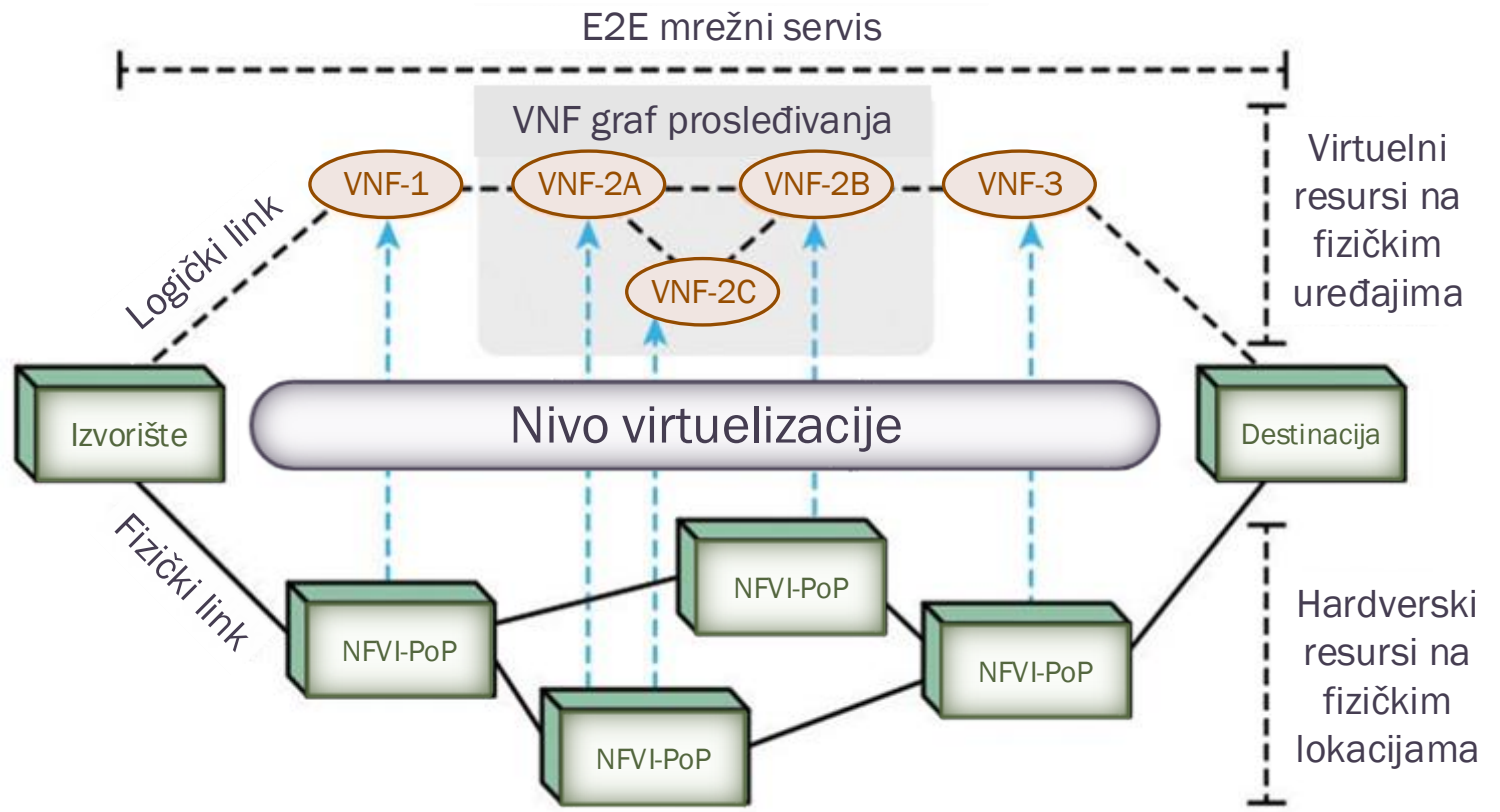


- Novi pristup kada je virtuelizacija u pitanju
- Kontejneri se izvršavaju na operativnom sistemu hosta
- Za razliku od VM-ova, kontejneri ne emuliraju fizičke servere
- Kontejnerizovane aplikacije na hostu dijele isti OS kernel
  - Ovo eliminiše potrebu da se za svaku aplikaciju instalira posebni OS
  - *Velika ušteda resursa!*
- Host može podržati veoma veliki broj kontejnera

NFV koncepti

- Vodeću ulogu u NFV standardizaciji ima NFV ISG
- Ključne definicije i termini:
  - *N-PoP (Network Point of Presence)* – lokacija na kojoj je mrežna funkcija implementirana bilo kao hardverski uređaj bilo kao softverski kontejner
  - *NFVI-PoP* – lokacija na kojoj je mrežna funkcija implementirana ili može biti implementirana kao VNF
  - *NFVI-Node* – hardverski uređaj koji podržava NFV tehnologiju
  - *VNF* – implementacija mrežne funkcije na NFV infrastrukturi
  - *PNF* (*Physical Network Function*) – hardverska implementacija mrežne funkcije
  - *Mrežni servis* – kompozicija mrežnih funkcija koja se koristi za pružanje nekog servisa
  - *Mrežni put prosleđivanja* – uređena lista konekcija koje čine lanac mrežnih funkcija, uz odgovarajuće administrativne politike
  - *VNF graf prosleđivanja* – Graf logičkih linkova koji povezuju VNF čvorišta

Primjer NFV konfiguracije



Kreiranje mrežnih servisa

Ključni principi:

1. *Uvezivanje servisa* (eng. *service chaining*) – VNF funkcije su modularne i svaka pruža ograničenu funkcionalnost. Da bi se nekom toku pružila željena fukcionalnost operator kreira lanac povezanih VNF funkcija kroz koji će tok biti obrađivan.
2. *Upravljanje i orkestracija* (*MANO - Management and Orchestration*) – Modul zadužen za instalaciju VNF funkcija i upravljanje njihovim životnim ciklusom (npr. kreiranje, uvezivanje, brisanje, migriranje, skaliranje, itd.).
3. *Distribuirana arhitektura* – VNF može da se sastoji od jedne ili više VNF komponenti, od kojih svaka implementira jedan dio njene funkcionalnosti. VNF komponente mogu biti impementirane na različitim hostovima u cilju skalabilnosti i redudanse.

NFV prednosti

- Smanjeni kapitalni troškovi – nema redudantnih resursa
- Smanjeni operativni troškovi – manje fizičkih uređaja, manja potrošnja energije, manji troškovi upravljanja
- Mogućnost brzog pokretanja novih servisa
- Veći nivo interoperabilnosti zbog standardizovanih i otvorenih interfejsa
- Fleksibilno skaliranje resursa
- Jedna platforma za više aplikacija i različite grupe korisnika
- Veća stopa inovacija - otvoreno tržište za softveraše, male kompanije i akademsku zajednicu

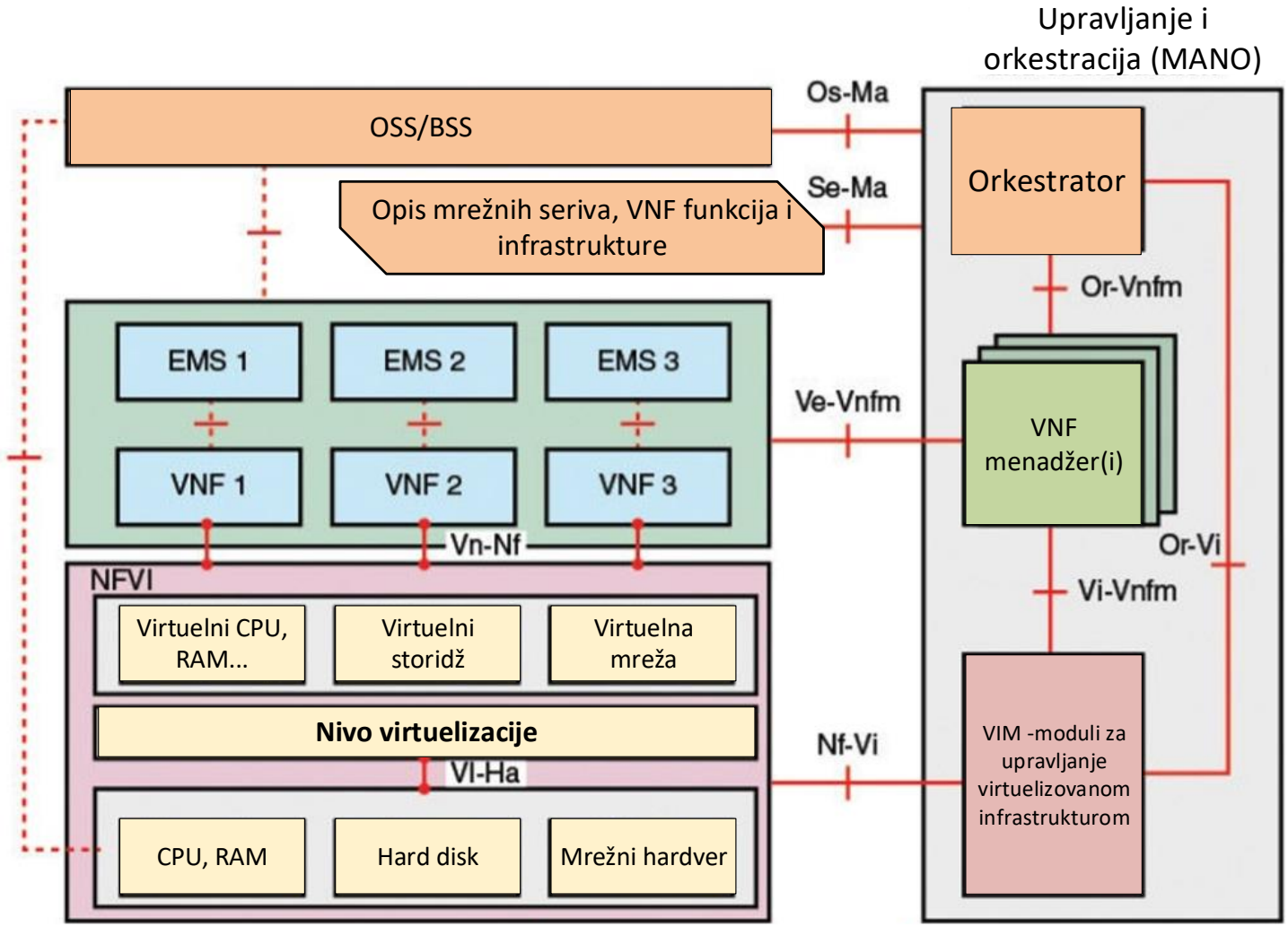
NFV zahtjevi

- *Portabilnost/interoperabilnost*: Mogućnost izvršavanja VNF modula različitih proizvođača na raznih standardizovanim hardverskim platformama
  - Potreban je *jedinstveni* interfejs koji jasno razdvaja softverske module od hardvera.
- *Kompromis u pogledu performansi*: S obzirom da se koristi hardver opšte namjene, potrebno je uzeti u obzir potencijalnu degradaciju performansi
  - Izazov je kako svesti degradaciju performansi na što manji nivo korišćenjem odgovarajućih hipervizora i softverskih tehnologija, tako da je uticaj na kašnjenje i propusnost minimalan.
- *Koegzistencija sa konvencionalnom opremom*
  - NFV mora raditi u hibridnim mrežama sa klasičnim fizičkim uređajima i virtuelnim uređajima.
  - Virtuelni uređaji moraju koristiti postojeće *northbound* interfejse za upravljanje i kontrolu, i biti u mogućnosti da "sarađuju" sa konvencionalnim uređajima na kojima su implementirane iste funkcije.

NFV zahtjevi

- *Upravljanje i orkestracija*
  - Potrebni su standardizovani *northbound* interfejsi i mehanizmi za apstrakciju resursa.
- *Automatizacija funkcija* kao preduslov za skalabilnost
- *Sigurnost i robusnost na kvarove*
- *Stabilnost mreže*
  - Migracija i rekonfiguracija mrežnih funkcija u kompleksnom NFV okruženju ne smije uticati na stabilnost mreže.
- *Jednostavnost*
  - Virtuelizovane mrežne platforme moraju biti jednostavne za korišćenje.
- *Integracija*
  - Mrežnim operatorima se mora pružiti mogućnost kombinovanja servera, hipervizora i softverskih VNF funkcija različitih proizvođača.

NFV arhitektura



NFV arhitektura

- *NFV infrastruktura (NFVI)* – Hardverski i softverski resursi koji čine okruženje na kojima se kreiraju VNF funkcije. Virtuelizuje fizičke računarske, memorijske i mrežne resurse.
- *VNF/EMS* – Skup mrežnih funkcija implementiranih u softveru koje se izvršavaju na virtuelizovanom hadrveru opšte namjene (VNF funkcije) i skup modula za upravljanje VNF funkcijama (*Element Management System – EMS*).
- *NFV MANO* – Okruženje za upravljanje i orkestraciju svih mrežnih resursa.
- *OSS/BSS* – Sistem operativne i poslovne podrške koje implementira provajder VNF servisa. Čuva informacije o korisničkim ugovorima, sprovodi tarifiranje, vodi računa o garanciji kvaliteta servisa, itd.

NFV – upravljanje i orkestracija

MANO funkcionalni blokovi:

1. *NFV orkestrator* – Odgovoran za instalaciju i konfigurisanje novih mrežnih servisa i VNF lanaca, globalno upravljanje mrežom, validaciju i autorizaciju servisnih zahtjeva.
2. *VNF menadžer* – Nadgleda i upravlja "životnim ciklusom" VNF instanci. Zadužen je za kreiranje, skaliranje, ažuriranje i brisanje VNF funkcija.
3. *VIM (Virtualized Infrastructure Manager)* – Odgovoran je za upravljanje i kontrolisanje računarskih, memorijskih i mrežnih resursa NFV infrastrukture, obično unutar jednog administrativnog domena.

SDN & NFV

- SDN i NFV su komplementarne i ključne tehnologije na kojima će biti bazirane mreže naredne generacije.
- SDN omogućava dinamičku rekonfiguraciju mrežnih resursa i konekcija između VFN funkcija.
  - Bez SDN-a, NFV bi zahtijevao kompleksne manuelne intervencije.
- SDN ima značajnu ulogu u orkestraciji fizičkih i virtuelnih NFV resursa.
  - Omogućava funkcionalnosti kao što su rezervacija propusnog opsega, konfigurisanje mrežnih konekcija, prikupljanje mrežnih statistika, kontrola pristupa, prevencija zlonamjernih napada, itd.
- SDN omogućava kreiranje virtuelne mreže preko infrastruktura različitih operatora.
- SDN kontroler se može izvršavati kao VNF funkcija. Isto važi i za aplikacije koje se izvršavaju na kontroleru.