
Mrežni sloj: upravljačka ravnina

Poglavlje 7 — Skripta kolegija *Računalne mreže*

doc. dr. sc. Sandi Baressi Šegota

Fakultet informatike Sveučilišta Jurja Dobrile u Puli

Poglavlje 7

Mrežni sloj: upravljačka ravnina

Dok podatkovna ravnina prosljeđuje pojedinačne pakete, **upravljačka ravnina** odlučuje *kojim putem* paketi trebaju putovati — ona računa *tablice prosljeđivanja* koje podatkovna ravnina zatim koristi. To se postiže **usmjeravanjem** (routing). U ovom poglavlju upoznajemo dva temeljna algoritma usmjeravanja (link-state i distance-vector), hijerarhijsku organizaciju Interneta u autonomne sustave s protokolima **OSPF** i **BGP**, model **SDN** te pomoćni protokol **ICMP**.

Ishodi učenja

Po završetku poglavlja student će moći: objasniti zadaću upravljačke ravnine; primijeniti Dijkstrin (link-state) algoritam i objasniti distance-vector pristup; razlikovati intra-AS (OSPF) i inter-AS (BGP) usmjeravanje te ulogu politike; opisati SDN upravljačku ravninu i ICMP.

7.1 Usmjeravanje i algoritmi

Mrežu prikazujemo kao *težinski graf*: čvorovi su ruteri, a bridovi su linkovi s pridruženom *cijenom*. Cijena može predstavljati broj skokova, biti obrnuto proporcionalna propusnosti ili odražavati kašnjenje i opterećenje; izbor metrike izravno oblikuje putove koje algoritam odabire. Cilj usmjeravanja jest pronaći *najjeftiniji put* od izvora do svakog odredišta.

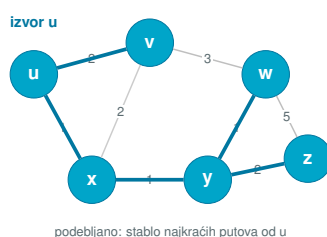


Figure 7.1: Težinski graf mreže i stablo najkraćih putova od izvora u (podebljano).

7.1.1 Link-state (Dijkstra)

Kod **link-state** pristupa svaki ruter zna *cijelu* topologiju mreže (dobiva je preplavljanjem informacija o stanju linkova). Nad tim potpunim grafom svaki ruter samostalno pokreće **Dijkstrin algoritam**: kreće od izvora, održava skup već „riješениh” čvorova te u svakom koraku odabire najbliži još neriješen čvor i ažurira procjene udaljenosti njegovih susjeda. Rezultat je stablo najkraćih putova, iz kojega ruter iščitava prvi skok prema svakom odredištu i tako popunjava tablicu prosljeđivanja. Link-state protokoli brzo konvergiraju, ali zahtijevaju razmjenu većeg broja poruka.

Za graf sa slike, izvođenje algoritma iz izvora u teče kako je prikazano u tablici: u svakom se retku u riješeni skup dodaje najbliži preostali čvor, a ostale se procjene po potrebi smanjuju.

korak	dodano u N'	ažurirane udaljenosti od u
0	u	$D(x)=1, D(v)=2$
1	x (najbliži, 1)	preko x : $D(y)=2$
2	v (2)	$D(w)=5$ preko v
3	y (2)	preko y : $D(w)=3, D(z)=4$
4	w (3)	—
5	z (4)	gotovo

Table 7.1: Tijek Dijkstrina algoritma iz izvora u .

7.1.2 Distance-vector (Bellman-Ford)

Kod **distance-vector** pristupa ruter *ne* poznaje cijelu topologiju — zna samo svoje izravne susjede i cijene do njih. Svaki čvor održava *vektor* procijenjenih udaljenosti do svih odredišta i povremeno ga šalje susjedima. Primivši susjedov vektor, čvor ažurira vlastite procjene prema Bellman-Fordovoj jednačini

$$D_x(y) = \min_v \{ c(x, v) + D_v(y) \},$$

gdje je $c(x, v)$ cijena do susjeda v , a $D_v(y)$ susjedova procjena udaljenosti do y . Promjene se tako postupno šire mrežom dok se procjene ne ustale (*konvergencija*). Nedostatak je **brojanje do beskonačnosti**: nakon prekida linka „loše vijesti putuju sporo“, pa cijena raste u malim koracima; ublažavaju ga tehnike poput *poisoned reverse*.

Kao primjer, neka čvor x ima izravne cijene $c(x, y) = 2$ i $c(x, z) = 7$. U početku x procjenjuje $D_x(z) = 7$ (izravno). Kada od susjeda y primi vektor s procjenom $D_y(z) = 1$, x ažurira svoju procjenu na $D_x(z) = \min\{7, 2 + 1\} = 3$, birajući put preko y . Ta poboljšana procjena postaje dio x -ovog vektora i u sljedećoj se razmjeni širi njegovim susjedima — upravo se tako korisne informacije postupno propagiraju kroz cijelu mrežu.

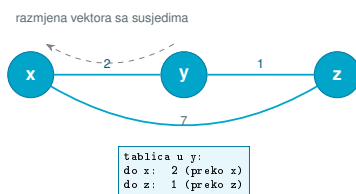


Figure 7.2: Distance-vector: čvorovi razmjenjuju vektore udaljenosti sa susjedima i ažuriraju procjene.

Link-state vs distance-vector

Link-state poznaje cijelu topologiju i koristi Dijkstru (brza konvergencija, više poruka); distance-vector poznaje samo susjede i koristi Bellman-Ford (manje poruka, ali moguć count-to-infinity). Oba se u praksi koriste — OSPF je link-state, a stariji RIP distance-vector.

7.2 Usmjeravanje na Internetu

Jedinstven algoritam ne može opsluživati stotine milijuna rutera, a k tome svaka organizacija želi *autonomiju* u odabiru politike. Zato je Internet podijeljen na **autonomne sustave** (AS) —

skupine rutera pod jednom upravom. Unutar AS-a koristi se *intra-AS* protokol, a između AS-ova *inter-AS* protokol.

Najčešći intra-AS protokol je **OSPF**, link-state protokol u kojem ruteri emitiraju stanje svojih linkova i lokalno računaju Dijkstru. OSPF podržava podjelu na *područja* (areas), čime se smanjuju opseg preplavlivanja i veličina tablica, kao i autentikaciju razmjene.

Između AS-ova djeluje **BGP** (*Border Gateway Protocol*), često nazvan „ljepilom“ Interneta. Sesije *eBGP* uspostavljaju se između rubnih rutera različitih AS-ova i služe za učenje vanjskih prefiksa, dok *iBGP* te informacije razdjeljuje unutar AS-a. BGP oglašava *dosežnost* prefiksa zajedno s atributima, ponajprije **AS-path** (nizom AS-ova kroz koje ruta prolazi) i sljedećim skokom.

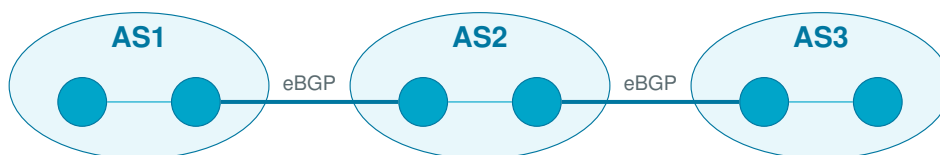


Figure 7.3: Autonomni sustavi povezani eBGP sesijama; unutar svakog AS-a djeluje intra-AS protokol.

Ključno je da BGP bira rute prema *poslovnoj politici*, a ne samo prema duljini puta. Odnosi su tipično *korisnik–pružatelj* (korisnik plaća tranzit) ili *peer–peer* (besplatna razmjena vlastitog prometa), pa AS oglašava rute tako da izbjegne plaćanje tuđeg tranzita. Kad više izlaza vodi do istog odredišta, primjenjuje se **hot-potato** usmjeravanje: odabire se izlaz s najmanjom *unutarnjom* cijenom, tj. paket se najbrže „izbaci“ iz vlastite mreže. Zbog utjecaja BGP-a na cijeli promet, sigurnost je važna: *BGP hijacking* (lažno oglašavanje tuđih prefiksa) suzbija se filtriranjem i mehanizmom **RPKI** koji provjerava podrijetlo ruta.

Zamislimo da AS primi dvije rute do istog prefiksa: jednu preko svog *korisnika* i jednu preko *pružatelja*. Iako je ruta preko pružatelja možda kraća po broju AS-ova, ekonomski je nepovoljna (znači plaćanje tranzita), pa će politika obično preferirati rutu preko korisnika. Tek ako se preferencije izjednače, u obzir dolaze tehnički kriteriji poput kraćeg AS-patha i, na kraju, hot-potato pravilo. Time se vidi da su BGP odluke slojevite: prvo politika, potom duljina puta, pa unutarnja cijena.

7.3 SDN, ICMP i upravljanje mrežom

U pristupu **SDN** (*software-defined networking*) upravljačka ravnina odvaja se od uređaja i seli u logički *centraliziran* kontroler. Kontroler ima globalni pogled na mrežu i putem *južnog* sučelja (npr. OpenFlow) puni tablice tokova u preklopnike, dok putem *sjevernog* sučelja komunicira s mrežnim aplikacijama (usmjeravanje, vatrozid, balansiranje opterećenja). Time mreža postaje programirljiva, a optimizacija jednostavnija jer se odluke donose iz jedne točke s potpunom slikom.

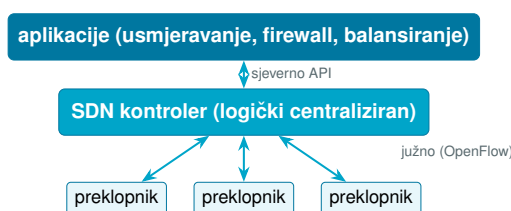


Figure 7.4: SDN arhitektura: aplikacije, centralizirani kontroler te preklopnici povezani OpenFlowom.

ICMP (*Internet Control Message Protocol*) prenosi kontrolne poruke i poruke o pogreškama; njegovi se paketi nose *unutar* IP datagrama. Tipične poruke uključuju *echo request/reply* (na kojima počiva *ping*), *destination unreachable* i *time exceeded*. Na potonjoj se temelji **traceroute**: slanjem paketa s postupno rastućim TTL-om svaki ruter na putu, kada TTL padne na nulu, vraća poruku o isteku vremena, čime se otkriva cijeli put do odredišta. Za nadzor uređaja koristi se **SNMP** (agent na uređaju i središnji upravitelj) uz bazu upravljanih objekata *MIB*.

Sažetak poglavlja

Upravljačka ravnina računa tablice prosljeđivanja usmjeravanjem. *Link-state* (Dijkstra) i *distance-vector* (Bellman-Ford) dva su temeljna algoritma. Internet je hijerarhijski organiziran u autonomne sustave: unutar njih djeluje **OSPF**, a između **BGP**, koji rute bira prema politici i AS-path atributu. **SDN** centralizira upravljanje putem kontrolera i OpenFlowa, a **ICMP** prenosi kontrolne poruke na kojima počivaju ping i traceroute.