SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

SEMINAR

Upravljački uređaji za programski upravljane komunikacijske mreže

Matija Šantl
Mentor: *prof. dr. sc. Maja Matijašević*
Voditelj: *dr. sc. Ognjen Dobrijević*

Zagreb, svibanj 2014.

SADRŽAJ

1. Uvod

Klasične komunikacijske mreže temeljene na prethodno programiranim uređajima čije karakteristike ovise od proizvođača do proizvođača onemogućuju provedbu ispitivanja novih komunikacijskih protokola u produkcijskim mrežama.

Kao rješenje tog problema, predložena je specifikacija *OpenFlow* kao izvedba programski upravljanih mreža (engl. *Software Defined Networks*). Osnovna ideja predloženoga pristupa je upravljanje prosljeđivanjem podataka u komutatorima pomoću programske podrške (engl. *Software*) smještene na zasebnom računalu, tzv. upravljačkom uređaju (engl. *Controller*) [11].

Apstrakcija elemenata mreže izdvajanjem funkcije upravljanja prosljeđivanjem i definiranjem komunikacijskih sučelja prema njima (npr. speficikacija *OpenFlow*) provodi se radi lakšeg i bržeg razvoja mrežnih usluga poput kontrole pristupa mreži, kontrole prihvata novih tokova, poboljšanja kvalitete usluge i mnogih drugih.

Cilj ovog seminarskog rada je proučiti i opisati specifikaciju *Openflow* te usporediti odabrane upravljačke uređaje (*OpenDaylight*, *POX* i *Floodlight*) za upravljanje komunikacijskim mrežama.

U drugom je poglavlju dan opis arhitekture programski upravljanih komunikacijskih mreža i specifikacije *OpenFlow*. Nakon toga, u trećem poglavlju opisani se upravljački uređaji koji podržavaju specifikaciju *OpenFlow*. Konačno, u četvrtom poglavlju dana je demonstracija rada odabranih upravljačkih uređaja. Na samom kraju nalazi se zaključak seminarskog rada.

2. Programski upravljane komunikacijske mreže

Osnovna ideja programski upravljanih mreža je razdvajanje upravljanja mrežom od samih podataka koji prolaze kroz mrežu [11].

Odvajanje upravljanja mrežom i usmjeravanja omogućuje komunikacijskoj mreži da postane izravno programirljiva, a pripadna se infrastruktura može prikazati apstraktno za aplikacije i mrežne usluge. Programski upravljane komunikacijske mreže (engl. *Software-Defined Networks, SDN*) time čine dinamičnu, upravljivu, prilagodljivu i isplativu arhitekturu koja pogoduje promjenjivim zahtjevima mrežnih usluga [8].

Za razliku od klasičnih unaprijed programiranih mrežnih uređaja, programski upravljane komunikacijske mreže rješavanje problema usmjeravanja paketa komunikacijskom mrežom obavljaju koristeći lako izmjenjivu programsku podršku. Upravo one donose mogućnost izvođenja praktičnih pokusa s novim mrežnim protokolima (npr. novi protokoli usmjeravanja) u dovoljno realističnim uvjetima [7].

Programski upravljane komunikacijske mreže također pogoduju razvoju mrežnih infrastruktura, posebice u javnom sektoru [6]. Automatizacija i centralizacija upravljanja komunikacijskom mrežom olakšava, a samim time i pojednostavljuje, operacije s kojima se komunikacijska mreža mora moći nositi.

Danas ne postoji standardizirano tijelo koje bi brinulo oko definiranja standarda za arhitekture programski upravljanih komunikacijskih mreža, sve arhitekture imaju neka osnovna svojstva koja su nužna za njihov rad.

Svojstva arhitekture programski upravljanih komunikacijskih mreža dana su u nastavku [9].

- **Izravno programirljiva:**
  Upravljanje mrežom je izravno programirljivo jer je odvojeno od usmjeravanja.
- **Agilna:**

Apstrakcija odvojenosti upravljanja i usmjeravanja dozvoljava administratorima dinamične prilagodbe promjene toka podataka duž mreže.

- **Centralizirana:**
  Logika upravljanja mrežom je centralizirana pomoću programirljivih upravljačkih uređaja koji znaju globalno stanje mreže.
- **Programski konfigurirana:**
  Moguće je konfigurirati, upravljati, zaštititi i optimirati mrežne resurse pomoću programa upravljačkih uređaja.
- **Neovisna o proizvođaču i zasnovana na otvorenim standardima:**
  Otvoreni standardi pojednostavljuju dizajn mreža i operacija zbog instrukcija koje pružaju upravljački uređaji koji nisu specifični za proizvođača uređaja ili protokola.

2.1. Arhitektura

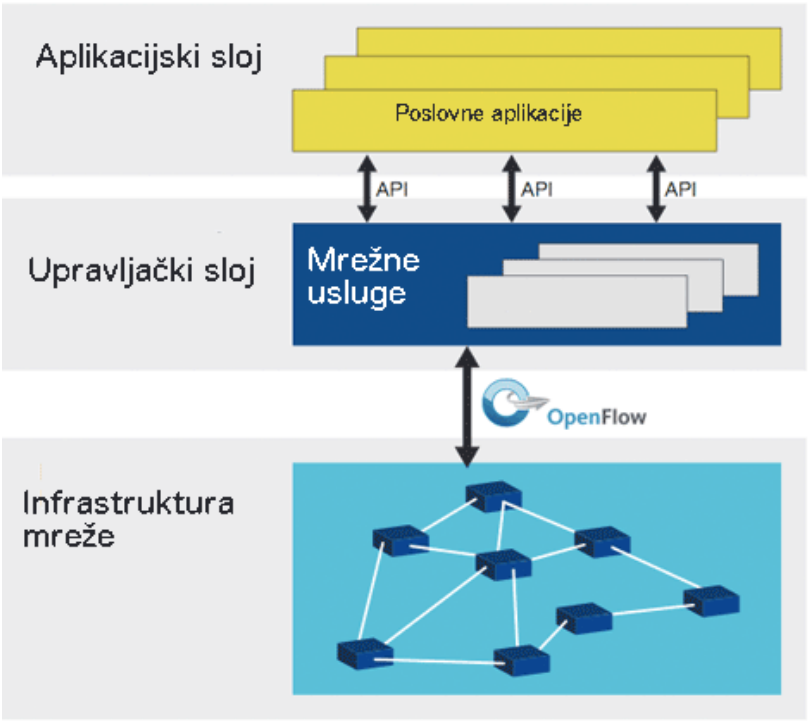
Prethodno navedena svojstva programski upravljanih komunikacijskih mreža ostvarena su različitim komponentama arhitekture.

Objasnimo najprije dva sučelja čiji se nazivi često pojavljuju kod opisa arhitektura računala, pa tako i komunikacijskih mreža. To su *northbound* i *southbound* sučelja. U nastavku su dana objašnjenja u konteksu programski upravljanih komunikacijskih mreža.

Northbound sučelje je sučelje koje određenoj komponenti programski upravljane komunikacijske mreže omogućuje komunikaciju s komponentama višeg sloja. Ono opisuje prostor protokolom podržane komunikacije između upravljačkog uređaja i aplikacije ili upravljačkih programa višeg sloja.

Southbound sučelje je sučelje koje određenoj komponenti programski upravljane komunikacijske mreže omogućuje komunikaciju s komponentama nižeg sloja.

Specifikacija *OpenFlow* protokola predstavlja *southbound* sučelje programski upravljanih komunikacijskih mreža. Glavna je zadaća omogućiti komunikaciju između upravljačkog uređaja komunikacijske mreže i komutatora ili usmjeritelja, tako da upravljački uređaji mogu otkrivati topologiju mreže, definirati tablice usmjeravanja i ostvariti zahtjeve koristeći *northbound* sučelje.



Slika 2.1: Arhitektura programski upravljanih komunikacijskih mreža, preuzeto iz [8]

U nastavku je dana lista komponenata SDN arhitekture [9].

- **SDN aplikacija:**
  SDN aplikacije su programi koji eksplicitno, izravno i programatski dojavljuju svoje mrežne potrebe i željeno ponašanje SDN upravljačkom uređaju koristeći *SDN northbound* sučelja.
- **SDN upravljački uređaj:**
  SDN upravljački uređaji su logički centralizirani entiteti koju su zaduženi za ispravan rad programski upravljanih mreža. Njima je posvećeno jedno od sljedećih poglavlja.
- **SDN *datapath*:**
  SDN *datapath* je logički mrežni uređaj koji izlaže vidljivost i omogućuje kontrolu nad prosljeđivanjem i obradom podataka.
- **SDN CDPI** [1] **sučelje:**
  SDN CPDI je sučelje definirano između SDN upravljačkog uređaja i SDN *datapatha* čija je minimalna funkcionalnost programirljiva kontrola nad prosljeđivanjem, obavještavanje o događajima, bilježenje statistike i obavještavanje o vlastitim sposobnostima.

[1] Control to Data-Plane Interface

- **SDN northbound sučelje:**
  SDN northbound sučelje je sučelje između SDN aplikacija i SDN upravljačkog uređaja koje tipično pruža apstraktni pogled na mrežu i omogućuje izravan utjecaj na ponašanje mreže.
- **Pogonski programi i agenti:**
  Svako je sučelje definirano kao par agent - pogonski program, pri čemu agent predstavlja vezu na *southbound* API dok pogonski program predstavlja vezu na *northbound* API.
- **Menadžment i administracija:**
  Menadžment pokriva statičke zadatke koje je bolje rješavati izvan aplikativne, kontrolne i podatkovne sfere. Komunikacija između entiteta menadžmenta izlazi iz okvira SDN arhitekture.

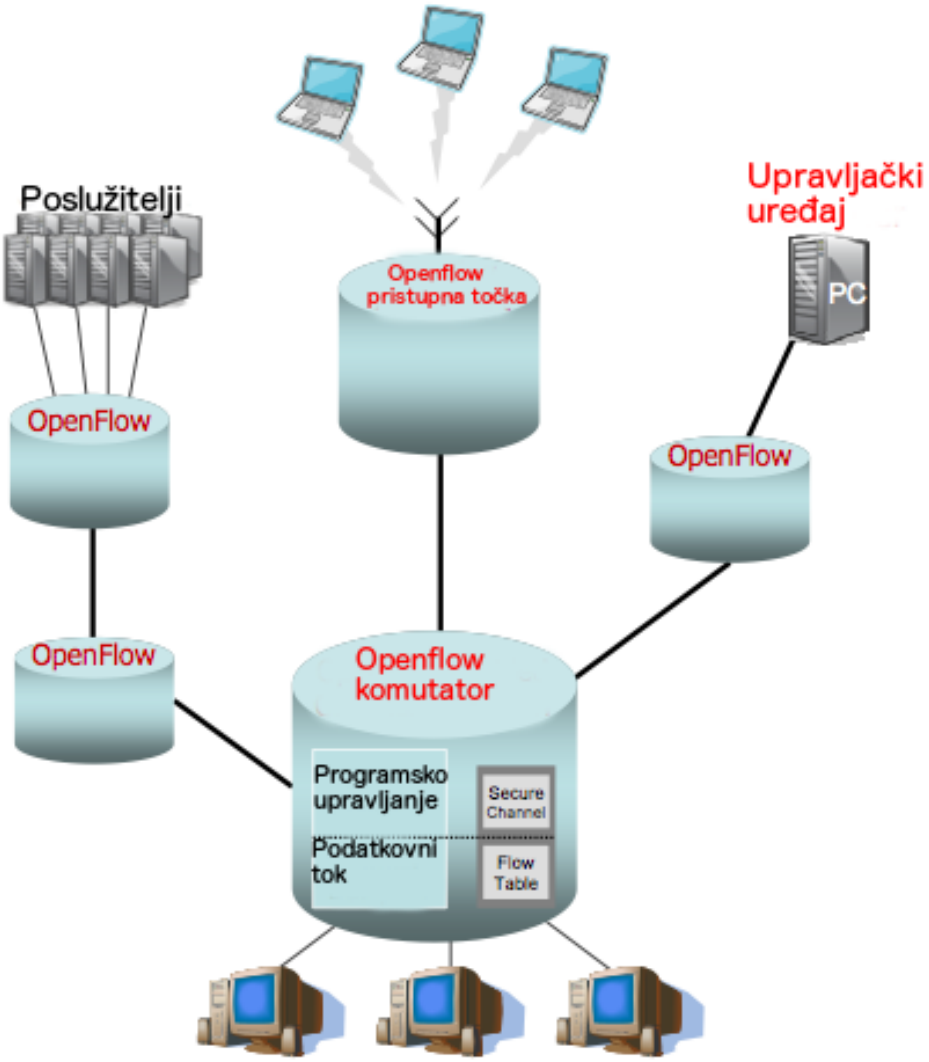
2.2. Specifikacija *OpenFlow*

Najpoznatiji protokol kojeg koriste programski upravljane komunikacijske mreže za komunikaciju s komutatorima i usmjeriteljima je definiran specifikacijom *OpenFlow*. Osim protokola, specifikacija *OpenFlow* definira i komponente i osnovni skup funkcionalnosti koje svaki *SDN datapath* element mora podržavati.

OpenFlow pruža otvoreni protokol za upravljanje tablicama tokova (engl. *Flow Table*) u različitim komutatorima i usmjeriteljima. U nekim od scenarija korištenja, mrežni administrator može odvojiti promet koji se koristi u produkciji i koji se koristi za istraživanje. Na taj se način dodatno olakšava ispitivanje novih algoritama za usmjeravanje, sigurnosnih modela, pa čak i alternativa IP[2]-u [7].

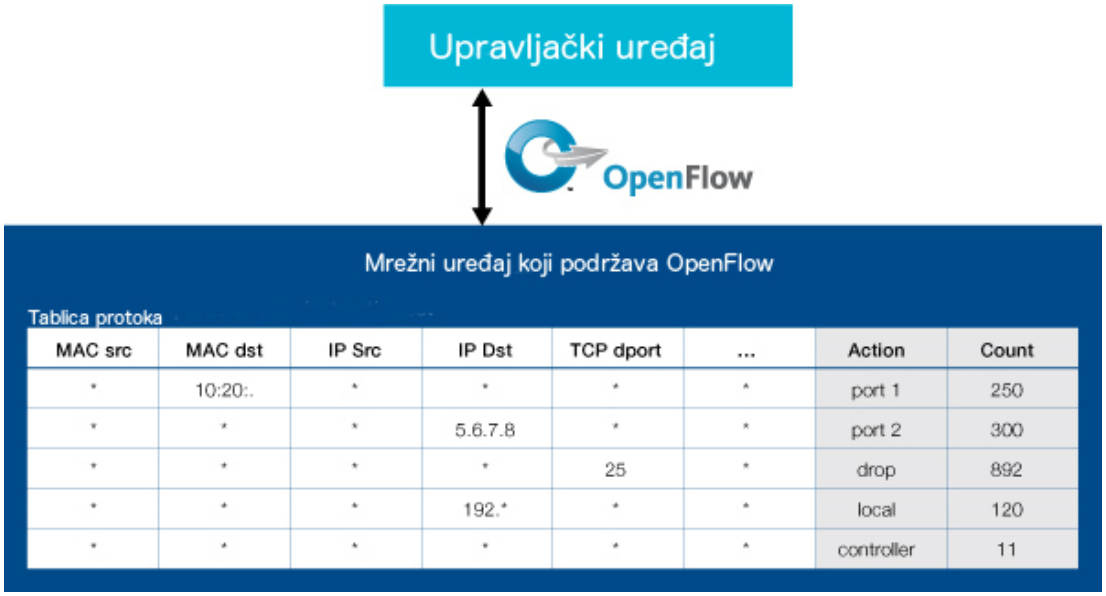
Svaki komutator koji podržava specifikaciju *OpenFlow* sadrži tablicu tokova (engl. *Flow Table*). *OpenFlow* komutatori iskorištavaju činjenicu da većina suvremenih komutatora i usmjeritelja koriste tablice tokova za ostvarenje vatrozida (engl. *Firewall*), *NAT* [3], kvalitetu usluge (engl. *QoS*) [4] i prikupljanje statistika. Iako je tablica tokova različita od proizvođača do proizvođača, *OpenFlow* koristi zajednički skup funkcionalnosti koji je prisutan kod većine komutatora i usmjeritelja.

[2]Internet Protocol
[3]Network Address Translation
[4]Quality of Service



Slika 2.2: Mreža *OpenFlow* komutatora i usmjeritelja, preuzeto iz [7]

Put kojim paket prolazi kroz *OpenFlow* komutator sastoji se od tablice usmjeravanja i jedne akcije koja je povezana sa svakim unosom u tablici usmjeravanja.



MAC src	MAC dst	IP Src	IP Dst	TCP dport	...	Action	Count
*	10:20:..	*	*	*	*	port 1	250
*	*	*	5.6.7.8	*	*	port 2	300
*	*	*	*	25	*	drop	892
*	*	*	192.*	*	*	local	120
*	*	*	*	*	*	controller	11

Slika 2.3: Tablica usmjeravanja, preuzeto iz [8]

Tri osnovna dijela *OpenFlow* komutatora su:

1. Tablica usmjeravanja (engl. *Flow Table*):
   Svakom unosu u tablici pridružena je akcija koja opisuje kako usmjeriti paket. Ukoliko se za neki dolazni paket ne pronađe odgovarajuća akcija, komutator šalje upit upravljačkom uređaju kako bi saznao koju akciju treba poduzeti.

2. Siguran kanal (engl. *Secure Channel*):
   Koristi se za povezivanje komutatora s upravljačkim uređajem. Riječ je o *SSL/TLS* [5] vezi koja preko transportnog protokola *TCP* [6] prenosi poruke protokola definiranog specifikacijom *OpenFlow* između upravljačkog uređaja i mrežnog uređaja.

3. *OpenFlow* protokol:
   Definira komunikaciju između komutatora i upravljačkog uređaja.

[5] Secure Sockets Layer/Transport Layer Security
[6] Transmission Control Protocol

3. Usporedba *OpenFlow* upravljačkih uređaja

Upravljački uređaj (engl. *Controller*) u programski upravljanoj komunikacijskoj mreži je "mozak" te mreže [4]. To je strateška kontrolna točka komunikacijske mreže, prenosi informacije LAN-komutatorima (engl. *switch*) i usmjeriteljima (engl. *router*) koristeći *southbound API* [1], ali i aplikacijama i poslovnoj logici koristeći *northbound API*.

Danas postoje brojni *OpenFlow* upravljački uređaji poput *NOX*, *POX*, *RYU*, *BEACON*, *IRIS*, *MUL*, *Jaxon*, *Maestro*, *Trema*, *OpenDaylight* i *Floodlight*. U nastavku rada naglasak je stavljen na *POX*, *OpenDaylight* i *Floodlight*.

Upravljački uređaj programski upravljane komunikacijske mreže uobičajeno sadrže kolekciju modula koji izvršavaju različite mrežne zadaće. Neke od tih zadaća su otkrivanje uređaja i njihovih mogućnosti unutar mreže, bilježenje mrežne statistike i sl. Također podržavaju dodavanje ekstenzija koje proširuju mogućnosti upravljačkih uređaja, kao npr. algoritam za analizu i upravljanje novim pravilima u komunikacijskoj mreži.

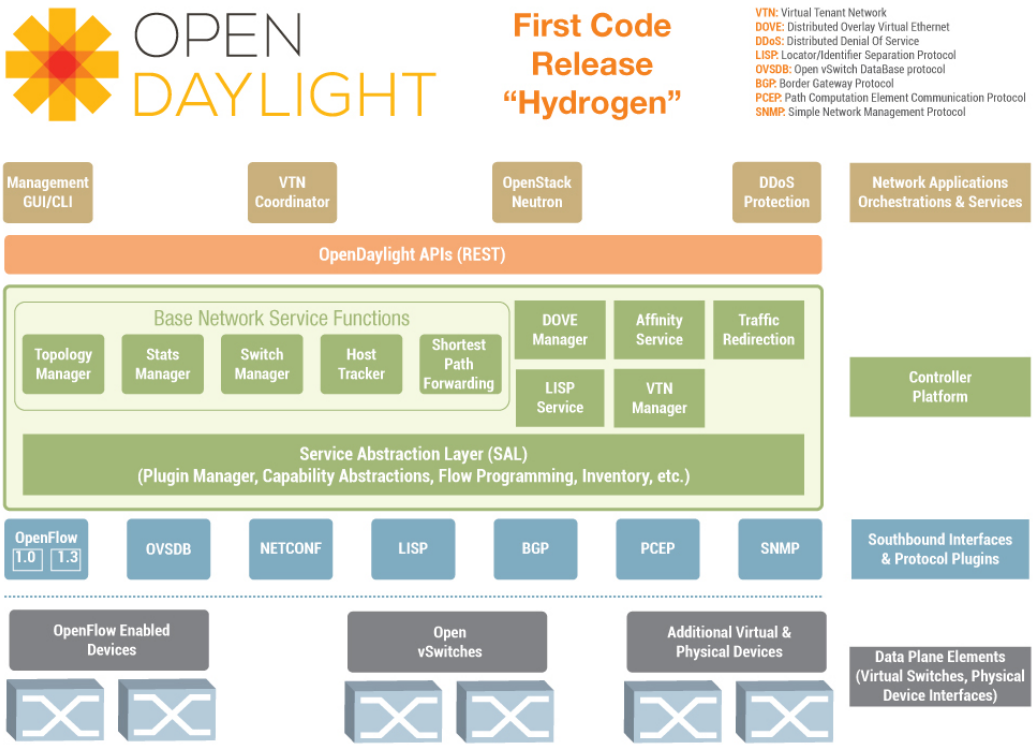
Glavna zadaća upravljačkih uređaja je dodavanje i brisanje unosa iz tablice usmjeravanja. Korisnik može koristeći upravljačke uređaje upravljati cijelom mrežom *OpenFlow* komutatora i tako imati svu slobodu nad upravljanjem usmjeravanja.

3.1. OpenDaylight

Kao prvi upravljački uređaj za upravljanje komunikacijskim mrežama je *OpenDaylight*. *OpenDaylight* je upravljački uređaj otvorenog koda koji je napisan u programskom jeziku *Java*. Kao takav, upravljački uređaj *OpenDaylight* se može koristiti na svim uređajima i operacijskim sustavima koji podržavaju *Javu* [1].

[1] Application Programming Interface

Platforma upravljačkog uređaja sadrži određene jezgrene pakete, gdje svaki od njih obavlja jednu od ključnih zadaća. Svaki se jezgreni paket može dinamički uključiti kako bi izvršavao određene mrežne zadaće. Postoji niz osnovnih mrežnih usluga za takve zadatke, kao npr. otkrivanje topologije mreže i mogućnosti uređaja koji se nalaze u toj mreži. Dodatno, usluge orijentirane prema platformi i druga proširenja također se mogu ugraditi u upravljački uređaj za poboljšanu funkcionalnost programski upravljanih komunikacijskih mreža. Jezgreni paketi ostvaruju se *Java* sučeljima.



Slika 3.1: Arhitektura upravljačkog uređaja *OpenDaylight*, preuzeto iz [1]

Upravljački uređaj izlaže (engl. *northbound*) API koji onda koriste aplikacije. *OpenDaylight* podržava *OSGi* [2] i obostrani *REST* [3]. *OSGi* okosnicu koriste aplikacije koje će se izvršavati u istom adresnom prostoru kao i upravljački uređaj, dok se *REST* okosnica temelji na mrežnom pristupu i koriste ga aplikacije koje se izvršavaju udaljeno (izvan adresnog prostora upravljačkog uređaja). Poslovna logika i ostvarenje algoritama leži u aplikacijama. Te aplikacije koriste upravljački uređaj kako bi prikupile informacije o komunikacijskoj mreži, izvršile algoritme za dobivanje statistika te u konačnici s dobivenim informacijama odredile nova pravila koja će upravljati podatkovnim tokovima

[2] Open Service Gateway initiative
[3] Representational state transfer

komunikacijske mreže.

OSGi specifikacija opisuje sustav modula i platformu servisa za programski jezik *Java* koja ostvaruje potpune i dinamičke komponente modela. Aplikacije ili komponente se, koristeći *OSGi* specifikaciju, mogu instalirati, pokrenuti, zaustaviti, ažurirati i obrisati s udaljenog računala bez da se zahtijeva ponovno pokretanje sustava.

REST je arhitekturni stil koji se koristi za izradu raspodijeljenih aplikacija, a temelji se na prijenosu prikaza stanja resursa. Tri su bitne stvari za shvaćanje rada *REST* arhitekture.

- klijent šalje usluzi zahtjev za resursom. U zahtjevu se šalje identifikator resursa te je poželjno da se pošalje i vrsta podataka koja se očekuje u odgovoru.
- odgovor sadrži prikaz stanja resursa u obliku niza okteta zajedno s metapodacima koji opisuju taj resurs.
- dohvaćanje prikaza može uzrokovati promjenu stanja resursa.

Southbound sučelje podržava više protokola (kao zasebnih proširenja) poput *OpenFlow* specifikacije verzije 1.0 i 1.3, BGP-LS (engl. *Border Gateway Protocol - Link State*) itd. Ta se proširenja dinamički povezuju u (engl. *Service Abstraction Layer, SAL*), apstraktni sloj usluga. Apstraktni sloj usluga izlaže usluge uređaja te određuje na koji će se način koja usluga ostvariti, bez obzira na protokol koji se koristi između upravljačkog uređaja i mrežnih uređaja.

Budući da je naglasak za daljnji rad predviđen za upravljački uređaj *OpenDaylight*, u nastavku je dan kratak pregled upravljačkih uređaja *POX* i *Floodlight*.

3.2. POX

Upravljački uređaj *POX* nastao je iz *NOX* upravljačkog uređaja, koji je okolina koja pruža podršku za programiranje i izvođenje *OpenFlow* upravljačkih uređaja. *NOX* je ujedno i prvi upravljački uređaj koji je implementirao *OpenFlow* specifikaciju. *POX* je unaprijeđenje *NOX* upravljačkog uređaja u pogledu primarnog programskog jezika kojeg koristi, tako se umjesto jezika *C++* koristi jezik *Python* [3].

POX se nalazi pod aktivnim razvojem te se primarno koristi u istraživačke svrhe. Neke od značajki upravljačkog uređaja *POX* su:

- jednostavno sučelje
- komponente koje su pogodne za ponovnu upotrebu (npr. odabir puta, otkrivanje topologije itd.)

- neovisnot o platformi
- podrška za grafičko korisničko sučelje

POX nije samo okosnica za ostvarivanje logike upravljačkog uređaja već se koristi i za otkrivanje i distribuciju prototipova, virtualizaciju mreža, dizajniranje upravljačkih uređaja i programskih modela.

3.3. Floodlight

Floodlight je upravljački uređaj otvorenog koda koji je napisan u programskom jeziku *Java*.

Floodlight je upravljački uređaj poslovne klase otvorenog koda čija se zajednica sastoji od velikog broja korisnika, uključujući i inženjere, te je time podrška od strane zajednice dosta jaka [2].

Neke od značajki upravljačkog uređaja *Floodlight* su:

- sustav učitavanja modula koji olakšava pisanje proširenja upravljačkog uređaja
- jednostavan za postaviti s minimalnim postavkama
- podrška za veliki broj *OpenFlow* komutatora
- mogućnost rada s mješovitim (*OpenFlow* i ne*OpenFlow*) mrežama
- visoke performanse rada

Funkcionalnost upravljačkog uređaja *Floodlight* je moguće proširiti modulima koji su ostvareni kao *Java* sučelja.

3.4. Usporedba upravljačkih uređaja

U nastavku je dana tablica koja sumira usporedbu opisanih upravljačkih uređaja po sljedećim stavkama: naziv upravljačkog uređaja, zadnja podržana verzija specifikacije *OpenFlow*, vrsta i verzija licence te programski jezik kojim je upravljački uređaj razvijen.

Upravljački uređaj	Verzija	OpenFlow podrška	Licenca	Programski jezik
OpenDaylight	v1.0	1.3	EPL v1	Java
POX	dart	1.0	Apache v2	Pyhton
Floodlight	v0.9	1.0	GPL v2	Java

Upravljački uređaji *OpenDaylight*, *POX* i *Floodlight* podržani su na operacijskim sustavima *Windows*, *OS X* te na distribucijama *GNU/Linux* operacijskog sustava, a njihove potpune dokumentacije dostupne su na pripadajućim web sjedištima [1, 2, 3].

4. Demonstracija rada odabranih upravljačkih uređaja

Upravljački uređaji pokretani su unutar konfiguriranog virtualnog stroja koji je preuzet sa [5]. Riječ je o 64-bitnom operacijskom sustavu *GNU/Linux Ubuntu* na kojem su postavljeni potrebni alati za početak korištenja upravljačkih uređaja. Za virtualizaciju mreže nad kojom su pokretani upravljačku uređaji korišten je alat *Mininet*.

Za pokretanje virtualnog stroja korišten je program *VirtualBox*.

4.1. Stvaranje virtualne mreže koristeći alat *Mininet*

Mininet je alat naredbenog retka koji omogućuje brzo stvaranje prototipova programski upravljanih komunikacijskih mreža. Virtualnu mrežu koja je korištena prilikom ispitivanja rada svakog upravljačkog uređaja stvaramo sljedećom naredbom .

```
1 $ sudo mn --topo single,3 --mac --controller remote --switch ovsk
```

Parametrom *–topo single, 3* naznačujemo da želimo mrežnu topologiju koja se sastoji od jednog komutatora na kojeg su spojena tri računala.

Parametrom *–mac* naznačujemo da se automatski postave MAC adrese svih uređaja u mreži.

Parametrom *–controller remote* naznačujemo da komutatorom upravlja udaljeni upravljački uređaj.

Parametrom *–switch ovsk* naznačujemo da komutator koristi *Open vSwitch Kernel* način rada. *Open vSwitch* je izvedba *OpenFlow* komutatora koji ima najopsežniju dokumentaciju i najveću zajednicu korisnika [11].

4.2. Pokretanje upravljačkog uređaja *OpenDaylight*

Najprije je potrebno stvoriti mrežu nad kojom ćemo demonstrirati rad upravljačkog uređaja kako je navedeno u 4.1.

Nakon toga, kako bismo uključili *OpenDaylight* u rad komunikacijske mreže, potrebno je izgraditi *Java* aplikaciju kojom ćemo upravljati komunikacijskom mrežom.

```
1 $ ~/SDNHub_Opendaylight_Tutorial/commons/parent$ mvn clean install
```

Pokretanje upravljačkog uređaja radimo pokretanjem skripte *run.sh*.

```
1 $ ~/SDNHub_Opendaylight_Tutorial/distribution/opendaylight/target/
    distribution.tutorial_L2_forwarding-1.0.0-SNAPSHOT-osgipackage/
    opendaylight$ ./run.sh
```

4.3. Pokretanje upravljačkog uređaja *POX*

Najprije je potrebno stvoriti mrežu nad kojom ćemo demonstrirati rad upravljačkog uređaja kako je navedeno u 4.1.

Budući da *POX* za sad ne podržava specifikaciju *OpenFlow* verzije 1.3, moramo ručno naznačiti da ćemo koristiti *OpenFlow* specifikaciju verzije 1.0.

```
1 $ sudo ovs-vsctl set bridge s1 protocols=OpenFlow10
```

POX upravljački uređaj dolazi s nekoliko gotovih primjera koji se mogu odmah pokrenuti. Kako bismo vidjeli upravljački uređaj u akciji, možemo pokrenuti jedan od primjera.

```
1 $ cd /home/ubuntu/pox && ./pox.py log.level --DEBUG forwarding.
    tutorial_l2_hub
```

Prethodnom smo naredbom pokrenuli upravljački uređaj *POX* koji se povezao s usmeriteljem iz virtualne mreže, te je nakon toga moguća komunikacija između računala koja se nalaze u toj mreži.

4.4. Pokretanje upravljačkog uređaja *Floodlight*

Pokretanje upravljačkog uređaja *Floodlight* radi se u dva koraka. Prvo se koristeći sljedeći blok naredbi izgradi izvršna *Java jar* datoteka.

```
1 $ cd floodlight && ant;
```

Nakon čega se pokrene upravljački uređaj *Floodlight*.

```
1 $ java -jar target/floodlight.jar
```

Nakon što smo pokrenuli upravljački uređaj, potrebno stvoriti mrežu nad kojom ćemo demonstrirati rad upravljačkog uređaja, a to možemo napraviti prema uputama u 4.1. Kad stvorimo mrežu moramo promijenti verziju specifikacije *Openflow* koju komutator koristi jer *Floodlight* ne podržava verziju 1.3. To radimo na isti način opisan u 4.3.

4.5. Rezultati

Ostvarena je funkcionalnost svakog od odabranih upravljačkih uređaja. Obavlja li pojedini upravljački uređaj svoju zadaću ili ne ,ispitivano je na sljedeći način.

1. Stvaranje virtualne mreže koristeći alat *Mininet*.
2. Pokretanje naredbe *ping* između dva računala, pri čemu je očekivani ishod nemogućnost isporuke paketa.
3. Pokretanje upravljačkog uređaja.
4. Pokretanje naredbe *ping* između dva računala, pri čemu je očekivani ishod isporuka paketa.

U nastavku je dan isječak naredbenog retka koji prikazuje gornje korake za upravljački uređaj *OpenDaylight*.

Pokretanje alata *Mininet*.

```
$ sudo mn --topo single,3 --mac --switch ovsk --controller remote
*** Creating network
*** Adding controller
Unable to contact the remote controller at 127.0.0.1:6633
*** Adding hosts:
h1 h2 h3
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1) (h3, s1)
*** Configuring hosts
h1 h2 h3
*** Starting controller
*** Starting 1 switches
s1
*** Starting CLI:
mininet> h1 ping -c1 h2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
From 10.0.0.1 icmp_seq=1 Destination Host Unreachable

--- 10.0.0.2 ping statistics ---
1 packets transmitted, 0 received, +1 errors, 100% packet loss, time
    0ms
```

Pokretanje upravljačkog uređaja *OpenDaylight*.

```
$ ~/SDNHub_Opendaylight_Tutorial/distribution/opendaylight/target/
    distribution.tutorial_L2_forwarding-1.0.0-SNAPSHOT-osgipackage/
    opendaylight $ ./run.sh
```

Provjera mogućnosti komunikacije između računala.

```
mininet> h1 ping -c1 h2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_req=1 ttl=64 time=31.9 ms

--- 10.0.0.2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 31.981/31.981/31.981/0.000 ms
```

5. Zaključak

Cilj ovog seminarskog rada bio je proučiti i opisati specifikaciju (engl. *Openflow*) te usporediti odabrane upravljačke uređaje (*OpenDaylight*, *POX* i *Floodlight*) za upravljanje komunikacijskim mrežama.

U radu su navedene bitne značajke koncepata programski upravljanih komunikacijskih mreža i specifikacije *OpenFlow* koje se mogu naći u literaturi. Također, dan je pregled arhitekture koju predlaže *Open Network Foundation* čije su članice organizacije poput *Deutsche Telekom*, *Facebook*, *Google*, *Microsoft*, *Verizon* i *Yahoo!*. Upravljački uređaji odabrani su prema dostupnoj dokumentaciji i zajednice ljudi koji ih koriste i održavaju.

Sva tri upravljačka uređaja uspješno su instalirana i uspješno upogonjena. Iako su sva tri upravljačka uređaja dobro dokumentirana i iza sebe imaju veliku zajednicu ljudi koji poboljšavaju njihova svojstva, neki nedostaci ipak postoje. Upravljački uređaji *Floodlight* i *POX* još ne podržavaju specifikaciju *OpenFlow* verzije 1.3 . *OpenDaylight* i *Floodlight*, koji koriste programski jezik *Java*, imaju više mogućnosti, što ih čini pogodnijima za korištenje, ali upravo zbog toga zahtjevaju puno više prostora i računalnih resursa od, primjerice, upravljačkig uređaja *POX*.

Naglasak u daljnjem radu stavit će se na upravljački uređaj *OpenDaylight* te će se proučiti konkretne primjene u praksi.

6. Literatura

[1] Open daylight. http://www.opendaylight.org/.

[2] Project floodlight. http://www.projectfloodlight.org/.

[3] Pox. http://www.noxrepo.org/pox/about-pox/.

[4] Sdn controllers. http://www.sdncentral.com/sdn-controllers/.

[5] Sdn tutorial. http://sdnhub.org/tutorials/sdn-tutorial-vm-64-bit/.

[6] Cisco. Software-defined networking. http://www.cisco.com/web/strategy/docs/gov/cis13090_sdn_sled_white_paper.pdf, 2013. Cisco White Paper.

[7] Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, i Jonathan Turner. Openflow: Enabling innovation in campus networks. http://www.sigcomm.org/sites/default/files/ccr/papers/2008/April/1355734-1355746.pdf, 2008. ACM Communications Review.

[8] OpenNetworkFoundation. Software-defined networking: The new norm for networks. https://www.opennetworking.org/images/stories/downloads/sdn-resources/white-papers/wp-sdn-newnorm.pdf, Travanj 2012. ONF White Paper.

[9] OpenNetworkFoundation. Sdn architecture overview v.1.0. https://www.opennetworking.org/images/stories/downloads/sdn-resources/technical-reports/SDN-architecture-overview-1.0.pdf, 2013. ONF White Paper.

[10] OpenNetworkFoundation. Openflow switch specification v.1.4.0. https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-spec-v1.4.0.pdf, 2013. ONF White Paper.

[11] Ivan Vuk. Programski upravljano prosljedivanje medijskih tokova optimalnim mreznim putevima, 2013. FER.

7. Sažetak

U programski upravljanim komunikacijskim mrežama (engl. *Software-Defined Network, SDN*), funkcija prosljeđivanja prometa u komutatorima i usmjeriteljima odvaja se od funkcije upravljanja prosljeđivanjem, koja se izvodi na tzv. upravljačkom uređaju (engl. *controller*).

Odgovornost upravljačkog uređaja u takvim mrežama je definirati pravila prosljeđivanja za pojedine tokove podataka, koja ne ovise o informacijama dobivenim, primjerice, konvencionalnim protokolima komutiranja i usmjeravanja. U svrhu zadavanja pravila prosljeđivanja, specifikacija *OpenFlow* definira komunikacijski protokol između upravljačkog uređaja i mrežnih uređaja.