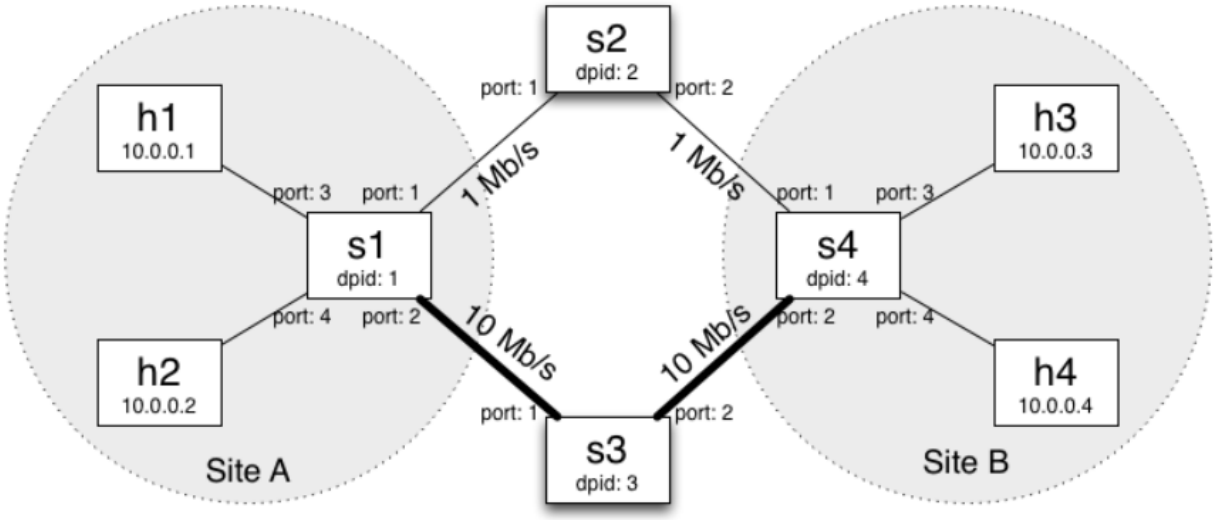
MREŽNA VIRTUELIZACIJA

UVODNE INSTRUKCIJE

Zadatak vježbe je slajsovanje OpenFlow mreže pomoću POX kontrolera. Kreiraćemo u Mininetu WAN mrežu sa Slike 1 koja povezuje dvije udaljene lokacije. Postoje dvije rute između lokacija:

- Ruta malog kapaciteta preko switch-a S2;
- Ruta većeg kapciteta preko switch-a S3.



Slika 1. Topologija mreže.

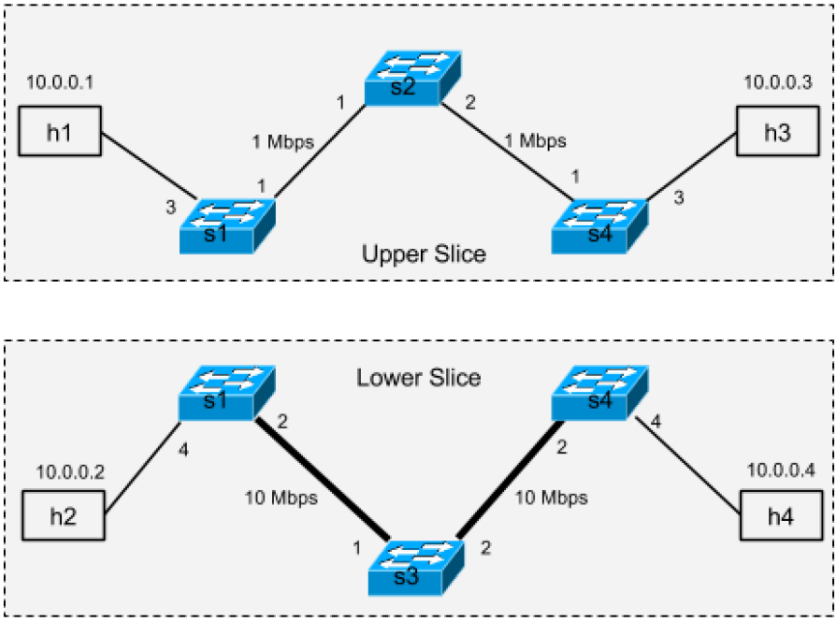
Zadatak se sastoji iz dva dijela: (1) jednostavno slajsovanje na osnovu topologije, i (2) slajsovanje na nivou tokova (tzv. flowspace slajsovanje). Prilikom izrade koristićemo pomoćni material sa linka: https://drive.google.com/open?id=1-5IMJRo-2sJgOPzc-gyn3B_1FTqCmdrf, koji se sastoji od sledećih fajlova:

- *mininetSlice.py*: Mininet skripte za kreiranje topologije;
- *topologySlice.py*: Skripta u okviru koje ćemo implementirat logiku za slajsovanje na nivou topologije;
- *videoSlice.py*: Skripta u okviru koje ćemo implementirati logiku za slajsovanje na nivou saobraćajnih tokova;

SLAJSOVANJE NA NIVOU TOPOLOGIJE

Zadatak ovog dijela mreže je kreiranje dva slajsa u mreži: "upper" i "lower", kao što je označeno na Slici 2. Korisnici u različitim slajsovima ne bi trebalo da mogu komunicirati jedan sa drugim. U praksi, operator

možda želi izdijeliti mrežu na ovaj način kako bi izolovano pružao servise različitim korisnicima. Funkcionalnost pružena na ovaj način slična je standardnim VLAN-ovima (Virtual Local Area Networks).



Slika 2. Slajsovanje na nivou topologije.

Da bi realizovali izolaciju između slajsova potrebno je blokirati saobraćaj između hostova u različitim slajsovima. Implementiraćemo ovu funkcionalnost na jednostavan način, slanjem `drop` instrukcija OpenFlow *switch*-evima. Na primjer, host h1 ne treba da komunicira sa hostom h2. Stoga, u tabeli tokova *switch*-a S1 potrebno je dodati nove zapise koji sprečavaju ovu komunikaciju.

Fajl topologySlice.py pruža okosnicu za implementaciju potrebne kontrolne logike. Kao i u prethodnoj vježbi, unaprijed je implementirana `launch()` funkcija, koja registruje novi kontrolni modul POX kontrolera. Funkcija `__handle__ConnectionUp()` poziva se svaki put kada se *switch* poveže na kontroler. U realizaciji koristićemo servise *discovery* i *spanning_tree* POX modula, koji su zaduženi za detekciju topologije I sprečavanja petlji prilkom prenosa saobraćaja.

TESTIRANJE KODA

Kada završite sa kreiranjem koda, pratite sledeće instrukcije:

Prekopirajte fajlove u POX direktorijum: `~/pox/pox/misc.`

```
$ mv topologySlice.py ~/pox/pox/misc/
$ mv mininetSlice.py ~/pox/pox/misc/
```

Pokrenite POX kontroler:

```
~/pox/pox/misc$ pox.py log.level --DEBUG misc.topologySlice &
```

U posebnom terminalu pokrenite Mininet skriptu:

```
~/pox/pox/misc$ sudo python mininetSlice.py
```

Sačekajte dok aplikacija na POX kontroleru ne potvrdi da su svi OpenFlow *switch*-evi povezani i da je *spanning tree* algoritam izvršen.

Provjerite da li hostovi smješteni u različitim slajsovima mogu da komuniciraju:

```
mininet> pingall
```

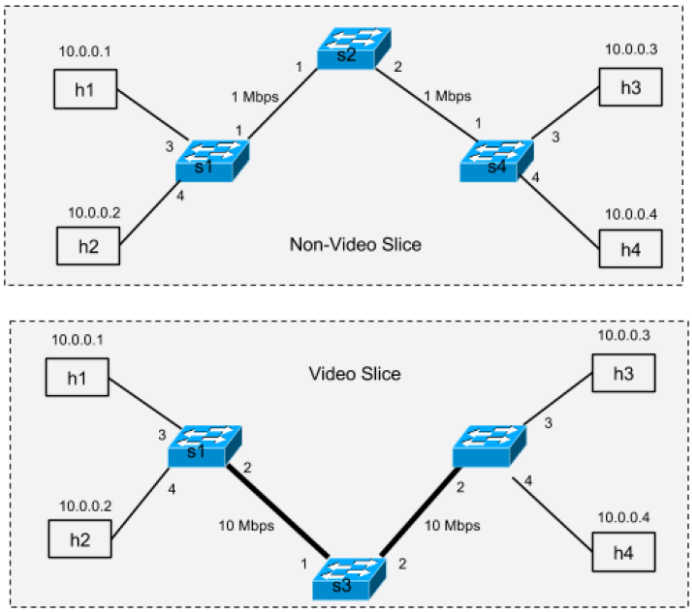
Trebalo bi da dobijete sledeći rezultat:

```
h1 -> X h3 X
h2 -> X X h4
h3 -> h1 X X
h4 -> X h2 X
*** Results: 66% dropped (8/12 lost)
```

SLAJSOVANJE NA NIVOU TOKOVA

U prethodnom dijelu vježbe kreirali smo dva fizički odvojena slajsa u mreži. Slajsovanje je moguće izvršiti na interesantniji način - na osnovu aplikacija koje generišu saobraćaj. Pretpostavimo da želimo prioritizovati video saobraćaj u mreži sa Slike 1 tako što ćemo ga slati rutom većeg kapaciteta, dok ćemo ostali saobraćaj slati rutom manjeg kapaciteta. Na prenos ostalog saobraćaja neće uticati prenos video saobraćaja, i obratno. Radi jednostavnosti, pretpostavićemo da je sav video saobraćaj usmjeren na TCP port 80.

Potrebno je kreirati kontrolnu logiku koja će omogućiti izolaciju između "video" i "ne-video" slajsa, kao što je prikazano na Slici 3.



Slika 3. Primjer FlowSpace slajsovanja.

U fajlu *videoSlice.py* definisana je klasa `VideoSlice`. Kao pomoć, unaprijed je dat kod za određene djelove logike slajsovanja. Potrebno je dopuniti strukturu **`portmap`** i ostale nedostajuće dijelove **`forward`** funkcije. Struktura **`portmap`** treba da mapira *switch* i odgovarajuće tokove na DPID drugog *switch*-a. Kreiranje ove strukture je već započeto u fajlu videoSlice.py, tako da se može relativno jednostavno i intuitivno dovršiti.

TESTIRANJE KODA

Prekopirajte fajlove *videSlice.py* i *mininetSlice.py* u .~/pox/pox/misc direktorijum.

Pokrenite kontroler:

```
~/pox/pox/misc$ pox.py log.level --DEBUG misc.videoSlice
```

U posebnom terminalu pokenite Mininet:

```
~/pox/pox/misc$ sudo python mininetSlice.py
```

Sačekajte dok se svi OpenFlow *switch*-evi povežu.

Provjerite da li hostovi mogu da komuniciraju međusobno.

```
mininet> pingall
```

Rezultat bi trebao da bude:

```
*** Ping: testing ping reachability
h1 -> h2 h3 h4
h2 -> h1 h3 h4
h3 -> h1 h2 h4
h4 -> h1 h2 h3
*** Results: 0% dropped (12/12 received)
```

Sada testirajte da li slajsovi funkcionišu ispravno, tj. provjerite da li video saobraćaj (saobraćaj sa destinacionim TCP portom 80 u ovom slučaju) koristi rutu kapaciteta 10 Mb/s a ostali saobraćaj rutu kapaciteta 1 Mb/s. Na primjer, dvije rute između hostova h2 I h3 mogu se testirati na sledeći način.

```
mininet > h3 iperf -s -p 80 &
mininet > h3 iperf -s -p 22 &
mininet> h2 iperf -c h3 -p 80 -t 2 -i 1
------------------------------------------------------------
Client connecting to 10.0.0.3, TCP port 80
TCP window size: 85.3 KByte (default)
------------------------------------------------------------
[  3] local 10.0.0.2 port 52154 connected with 10.0.0.3 port 80
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0- 2.1 sec  2.50 MBytes  9.77 Mbits/sec
mininet> h2 iperf -c h3 -p 22 -t 2 -i 1
------------------------------------------------------------
Client connecting to 10.0.0.3, TCP port 22
TCP window size: 85.3 KByte (default)
------------------------------------------------------------
[  3] local 10.0.0.2 port 53695 connected with 10.0.0.3 port 22

[ ID] Interval       Transfer     Bandwidth
[  3]  0.0- 4.0 sec   512 KBytes  1.05 Mbits/sec
```