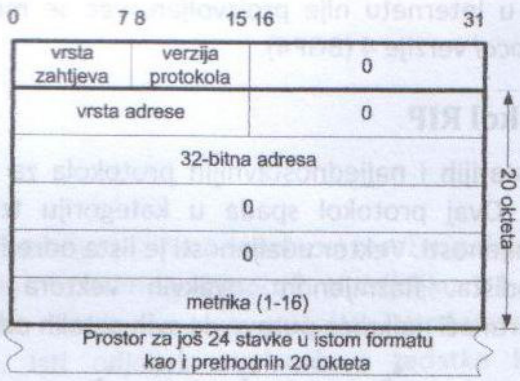


Slika 2: Zaglavlje protokola RIP

IMUNES

Zadatak 1. U simulatoru IMUNES, pomoću alata *Wireshark*, uhvatite paket koji pripada protokolu RIP, te proučite njegov sadržaj.

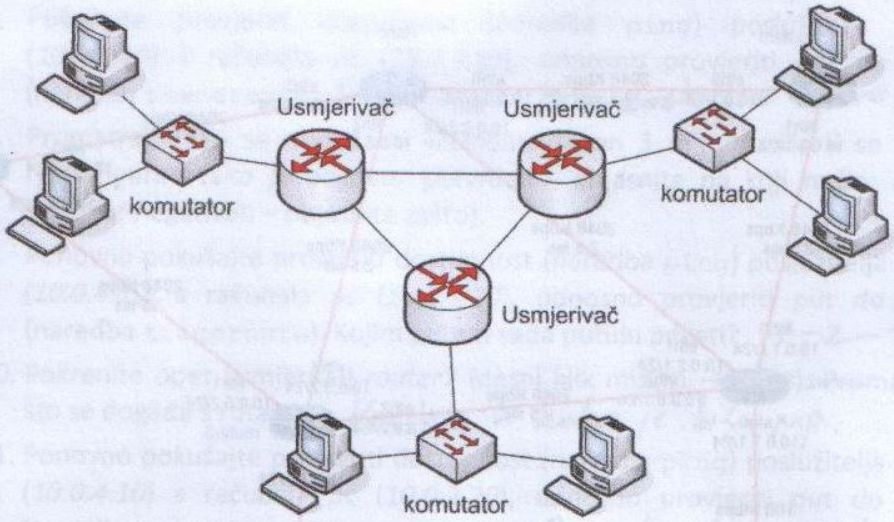
Prilikom pinga uhvaćen paket s protokolom RIPv2
Vrsta zahtjeva: 2 (odgovor)
9 metrika

IMUNES

Zadatak 2. U emulatoru/simulatoru IMUNES konstruirajte mrežu koja sadrži 3 podmreže povezane usmjerivačima, kao što je prikazano na Slici 3. Konfigurirajte usmjeravanje tako da dođe do petlje u usmjeravanju (bez korištenja protokola za usmjeravanje)[1]. Pomoću alata *Wireshark* utvrdite što se tada događa s paketima koji "uđu" u petlju.

Paketi se odbacuju nakon što se TTL toliko smanji da postane 0.

[1] Uputu o tome kako konfigurirati usmjeravanje iz grafičkog sučelja emulatora/simulatora IMUNES možete pronaći u posebnom dokumentu *Vodič za IMUNES*.



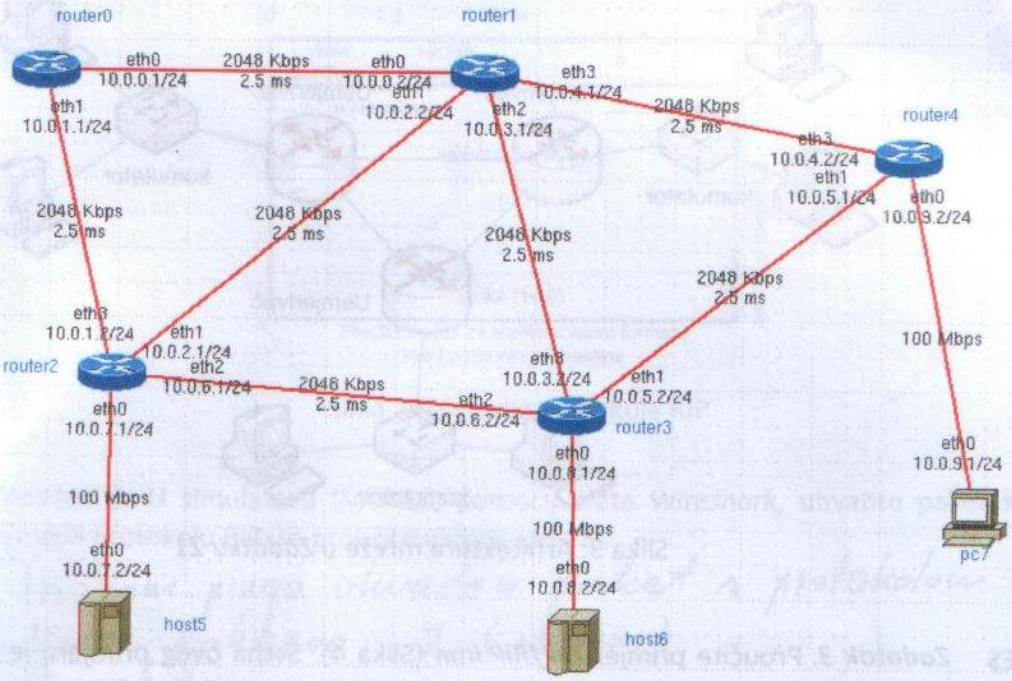
Slika 3: Arhitektura mreže u *Zadatku 21*

IMUNES

Zadatak 3. Proučite primjer *RIP/RIP.imn* (Slika 4). Svrha ovog primjera je pokazati što se događa u "tihoj" mreži – kako usmjeritelji razmjenjuju informacije o svojim susjedima. Scenarij vježbe:

1. Započnite simulaciju.
2. Pokrenite alat *Wireshark* na portu *eth3* usmjeritelja *router3*.
3. Otvorite *capture dialog* alata *Wireshark*. Označite opciju `Update list of packets in real time`.
4. Započnite snimanje prometa.
5. Nakon 30 sekundi trebali biste imati zabilježena barem 2 UDP datagrama.
6. Zaustavite snimanje prometa nakon otprilike 1 minute.
7. U alatu *Wireshark* otvorite jedan paket.
   a. Navedite izvorišnu i odredišnu adresu. 10.0.3.2, 224.0.0.9
   b. Navedite koliko često se RIP paketi šalju. 5.8, 27.1 pa 2.8 sekundi je razmak
8. Prikažite sadržaj prvog paketa i provjerite RIP podatke.
9. Svaki datagram sadrži rute sakupljene od strane usmjeritelja koji je poslao datagram, zajedno s pripadajućom metrikom. Komentirajte metriku (broj skokova).

Prikazuje međusobnu udaljenost usmjerivača.



Slika 4: Topologija *RIP/RIP.imn*

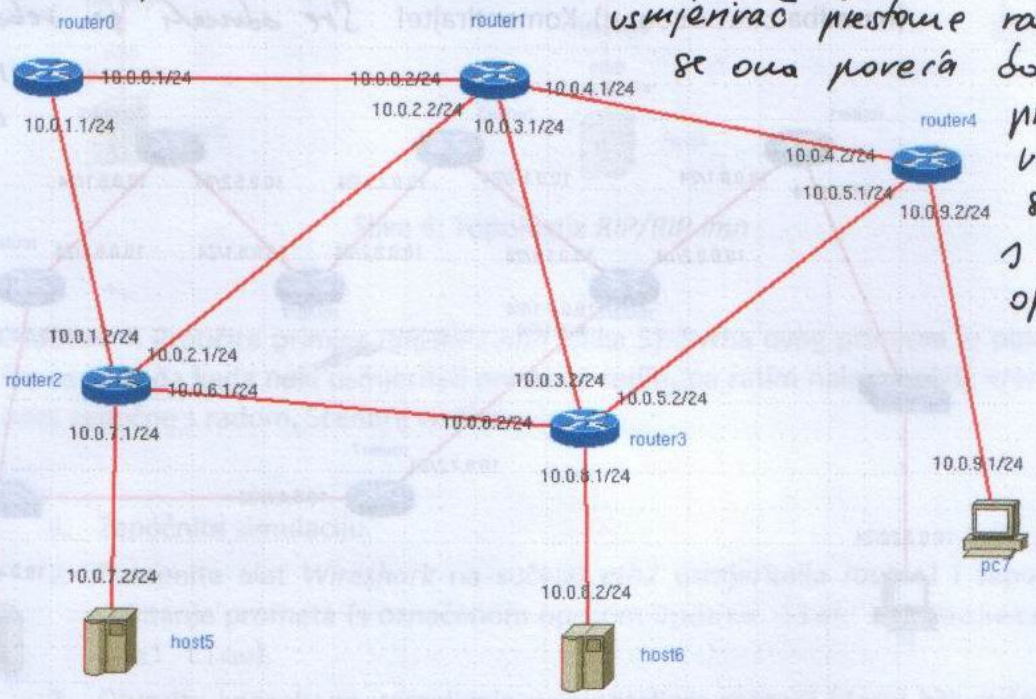
IMUNES

Zadatak 4. Proučite primjer *RIP/RIP1.imn* (Slika 5). Svrha ovog primjera je pokazati što se događa kada neki usmjeritelj prestane raditi, pa zatim nakon nekog vremena opet započne s radom. Scenarij vježbe:

1. Započnite simulaciju.
2. Pokrenite alat *Wireshark* na sučelju *eth2* usmjeritelja *router2* i započnite snimanje prometa (s označenom opcijom `Update list of packets in real time`).
3. Otvorite konzolu za upravljanje usmjeriteljem *router2* (desni klik mišem → *shell* prozor → `vtysh`) i upišite `show ip rip`. Ova naredba će prikazati rute koje su zapisane u usmjeritelju *router2*. Analizirajte tablicu usmjeravanja: dostupne mreže, te sljedeće skokove (*hopove*) i pripadajuću metriku.
4. Pokušajte provjeriti dostupnost (naredba `ping`) poslužitelja *server* (*10.0.4.10*) s računala *pc* (*10.0.3.20*), odnosno provjeriti put do njega (naredba `traceroute`). Što zapažate? Put ide preko usmjeritelja 7
5. Zaustavite usmjeritelj *router7* (desni klik mišem → *stop*).
6. Promatrajte na konzoli usmjeritelja *router2* što se događa (koristeći naredbe `show ip rip` i `show ip rip status`). Promatrajte vrijeme posljednjih osvježavanja. Koju će vrijednost dosegnuti parametar *Last Update* u trenutku kada naredba `show ip rip` prikaže metriku 16 za mrežu *10.0.4.0*.

2 : ∞

5. Zaustavite snimanje prometa nakon otprilike 1 minute.
6. U alatu *Wireshark* otvorite jedan paket.
   a. Navedite izvorišnu i odredišnu adresu. 10.0.6.1, 224.0.0.5
   b. Navedite koliko često se OSPF paketi šalju. Otprilike svaku sekundu. Negdje i 10
7. Prikažite sadržaj prvog paketa i provjerite OSPF podatke.
8. Da li OSPF, kao i RIP, koristi UDP kao transportni protokol? Ne.
9. Objasnite kako usmjeritelji pomoću protokola OSPF razmjenjuju informacije o metrici, te napravite usporedbu s protokolom RIP. Koristeći "Hello Packet"
10. Komentirajte značenje metrike kod protokola OSPF, te je usporedite s metrikom kod protokola RIP. Ne spominje se u pakelima. Vjerojatno jer se pretpostavlja da je prvi susjed 1, drugi 2, itd... Metrika kod RIP-a ima ulogu da ako neki usmjerivač prestane raditi i kad se ona poveća da kad ovaj proradi i veća metrika se usporedi s manjom i opet se rekonfigurira mreža ☺



Slika 6: Topologija *OSPF/OSPF.imn*

IMUNES

Zadatak 6. Proučite primjer *OSPF/OSPF1.imn* (Slika 7). Svrha ovog primjera je prikazati što se događa kad neki usmjeritelj prestane raditi, pa zatim nakon nekog vremena opet započne s radom. Scenarij vježbe:

1. Započnite simulaciju.
2. Pokrenite alat *Wireshark* na sučelju *eth2* usmjeritelja *router2* i započnite snimanje prometa (s označenom opcijom `Update list of packets in real time`).
3. Otvorite konzolu za upravljanje usmjeriteljem *router2* (desni klik mišem → *shell* prozor → `vtysh`) i upišite `show ip route`. Ova naredba će prikazati

rute koje su zapisane u usmjeritelju *router2*. Analizirajte tablicu usmjeravanja: dostupne mreže, te sljedeće skokove (*hopove*) i pripadajuću metriku.

4. Pokušajte provjeriti dostupnost (naredba `ping`) poslužitelja *server* (*10.0.4.10*) s računala *pc* (*10.0.3.20*), odnosno provjeriti put do njega (naredba `traceroute`). Što zapažate? Ništa posebno, ide najkraćim putem preko 7ice.
5. Zaustavite usmjeritelj *router7* (desni klik mišem → *stop*).
6. Promatrajte na konzoli usmjeritelja *router2* što se događa (koristeći naredbe navedene u tablici ispod opisa scenarija). Putevi se osvježeni i mreža rekonfi
7. Pokušajte provjeriti dostupnost (naredba `ping`) poslužitelja *server* (*10.0.4.10*) s računala *pc* (*10.0.3.20*), odnosno provjeriti put do njega (naredba `traceroute`). Što zapažate? Nije dostupan ?!? Opet zapinje na 6ici
8. Promatrajte što se događa u vremenu nakon 3 minute. Da li se mreža rekonfigurira (ako je odgovor potvrdan - objasnite na koji način; ako je odgovor negativan – objasnite zašto). Rekonfigurirala se tako da
9. Ponovno pokušajte provjeriti dostupnost (naredba `ping`) poslužitelja *server* (*10.0.4.10*) s računala *pc* (*10.0.3.20*), odnosno provjeriti put do njega (naredba `traceroute`). Kojim putem sada putuju paketi? Preko 6ice
10. Pokrenite opet usmjeritelj *router7* (desni klik mišem → *start*). Promatrajte što se događa s rutama. Rekonfigurirale se nazad
11. Ponovno pokušajte provjeriti dostupnost (naredba `ping`) poslužitelja *server* (*10.0.4.10*) s računala *pc* (*10.0.3.20*), odnosno provjeriti put do njega (naredba `traceroute`). Komentirajte! Sve se vratilo na staro i radi normalno, ne zapinje na 6ici

U konzoli usmjeritelja možete koristiti sljedeće naredbe:

Naredba	Značenje naredbe
`show ip route`	prikazuje sve rute
`show ip ospf route`	prikazuje OSPF rute
`show ip ospf interface`	prikazuje informacije o sučeljima usmjeritelja
`show ip ospf neighbor`	prikazuje informacije o susjedima usmjeritelja

Primjetite vrijednost *Dead Time* prilikom izvršavanja naredbe `show ip ospf neighbor` i pratite što se s njom događa nakon što usmjeritelj *router7* prestane s radom. Vrijednost se smanjuje i kad dođe do nule izbaci ju iz tablice na toj mreži. Valjda služi kao neki timer.

IMUNES ***Zadatak 8.*** Učitajte u simulator mrežu *Ping/ping.imn*. Iskoristite alat `netcat` kako bi uspostavili TCP vezu između dva proizvoljna računala u mreži[3]. Utvrdite što se događa nakon uspostavljanja veze između klijenta i poslužitelja[4].

Sve što se upiše na klijentu odvija se i na serveru.

Alat `netcat` očekuje podatke na svom *standardnom ulazu*, i onda te podatke šalje po uspostavljenoj vezi. Nadalje, sve što alat primi putem uspostavljene veze, ispisuje na *standardni izlaz*. Standardni ulaz i izlaz se obično poistovjećuju s tipkovnicom i zaslonom, te se čitanje sa standardnog ulaza svodi na upisivanje podataka putem tipkovnice, dok se ispisivanje na standardni izlaz svodi na ispisivanje na zaslon. Međutim, korištenjem operatora "<" i ">" moguće je ulaz i izlaz preusmjeriti iz datoteke, odnosno u datoteku. Na primjer,

```
nc 10.0.0.20 80 < README
```

će se spojiti na računalo IP adrese *10.0.0.20* na TCP vrata (*port*) 80 te sadržaj datoteke *README* (koju možete ispisati izvršavanjem naredbe: `cat README`) poslati kao da je upisan s tipkovnice. Nadalje, izvođenjem

```
nc -l 80 > file.txt
```

sve što klijent pošalje, umjesto na zaslon, upisat će se u datoteku *file.txt*.

IMUNES ***Zadatak 9.*** Pomoću alata `netcat` kopirati proizvoljnu datoteku s jednog računala na drugo. ☺ Radi. Na serveru se pokreni da se po gore navedenim postavkama.

SERVER: nc -l 100 > test.txt

KLIJENT: nc 10.0.0.20 100 < test.txt

Prije toga mu upravo: tekst kreirali na klijentu datoteku "test.txt

IMUNES ***Zadatak 10.*** Utvrdite mogu li se na jednom računalu pokrenuti dva procesa koji slušaju na istim vratima[5].

Mogu se pokrenuti, ne javlja grešku, ali samo onaj koji je prvi pokrenut radi.

---

[3] *Uputa:* na jednom računalu prvo treba pokrenuti `netcat` u tzv. *listen* načinu rada, uz navođenje odgovarajućih vrata (*port*) na kojima `nc` "sluša" zahtjeve za uspostavom veze; na drugom računalu treba pokrenuti alat `netcat` i od nje zahtijevati uspostavu veze prema prvom računalu i vratima na kojima `nc` sluša. Uspostavljena veza se prekida korištenjem kombinacije tipki *Ctrl* i *C*.

[4] *Uputa:* Upišite nešto putem tipkovnice na računalu s `netcat` klijentom i pritisnite *"Return"*.

[5] *Uputa:* pokušajte dvaput pokrenuti `nc` (istovremeno), iz dvije konzole istog računala.

IMUNES **Zadatak 11.** Ukoliko alatu `netcat` ne specificirate protokol koji će koristiti, on podrazumijeva protokol TCP. Ponovite pokus iz prethodnog zadatka uz korištenje protokola UDP.

SVE ISTO ...

2.2 Protokol UDP

Već je utvrđeno da protokol IP omogućava razmjenu paketa između računala u mreži. Ukoliko neko računalo A želi poslati podatke na drugo računalo B, formira se IP paket, kao izvorišna adresa paketa upisuje se adresa računala A, za odredišnu adresu stavlja se adresa računala B i paket se prosljeđuje prvom čvoru na putu prema B.

Međutim, s obzirom da se na jednom računalu može odvijati više međusobno neovisnih procesa (na primjer, različite korisničke aplikacije) koji imaju potrebu komunicirati s odgovarajućim procesima na drugom računalu, sama IP adresa u paketima koji pristižu na računalo nije dovoljna da bi se odredilo kojem procesu je potrebno dostaviti sadržaj paketa. IP adresa je paket "dovela" do računala, ali unutar računala ima više potencijalnih primatelja, odnosno, procesa.

Stoga je, osim IP adrese, u komunikaciju između procesa potrebno uvesti dodatnu oznaku koja određuje proces unutar računala kojem je potrebno dostavljati sadržaj IP paketa. Ova oznaka naziva se vrata (*port*).

IMUNES **Zadatak 12.** Prikupite i analizirajte promet koji generira `nc` prilikom prijenosa podataka s jednog računala na drugo, u slučaju kad se koristi protokol UDP. Što možete zaključiti o vratima i adresama; što se mijenja, a što ostaje isto?

Source i destination vrata ostaju ista, a ~~[illegible]~~ također i IP adrese. Neznam, il' bi se trebalo često mijenjati ??? Mislim da ne kad PC 1 samo šalje, a PC 2 samo prima

2.3 Protokol TCP

Za razliku od protokola UDP, TCP pruža bitno složeniju uslugu. Prije svega, TCP osigurava pouzdan prijenos niza okteta čuvajući njihov redoslijed. Pojam *pouzdan prijenos* znači da će, u slučaju gubitka paketa u mreži, TCP ponavljati slanje izgubljenih paketa sve dok se svi ne prenesu na odredište. Pojam *prijenos niza okteta* odnosi se na činjenicu da TCP procesima isporučuje jedan po jedan oktet, ne pružajući informaciju o načinu na koji su skupine okteta prilikom prijenosa kroz mrežu grupirane u IP pakete. Nadalje, TCP osigurava isporuku okteta u istom redoslijedu kojim su i poslani. Bez obzira na činjenicu da se u IP mrežama relativno često događa da paketi na odredište stižu različitim redoslijedom od onoga u kojem

su poslani, TCP "ispravlja" redoslijed IP paketa i procesu isporučuje "izvorni" slijed okteta.

Da bi osigurao pouzdan prijenos okteta, TCP koristi mehanizam potvrđivanja. Ispravan primitak svakog poslanog okteta mora biti potvrđen od strane primatelja. Ukoliko pošiljatelj ne dobije potvrdu o primitku okteta koje je već poslao, ponavlja se slanje istih okteta sve dok se ne dobije potvrda da ih je odredište primilo.

Prije slanja okteta između udaljenih procesa, TCP mora uspostaviti vezu. Da bi razlikovao procese kojima mora dostavljati pakete, koriste se vrata. Kao i kod UDP-a, TCP veza je jednoznačno određena četvorkom:

{*izvorišna IP adresa, izvorišna vrata, odredišna IP adresa, odredišna vrata*}.

IMUNES ***Zadatak 13.*** Pomoću analizatora *Wireshark* uhvatite proizvoljan TCP promet koji pripada jednoj vezi te odredite datagrame koji se razmjenjuju u fazama uspostave veze i raskida veze[6]

Uspostava : 3 datagrama - upit, odgovor, potvrda

Raskid : 4 datagrama - upit, odgovor, potvrda, potvrda

IMUNES ***Zadatak 14.*** Za uhvaćeni promet odredite koje se adrese i vrata koriste. Imaju li svi segmenti istu četvorku {*izvorišna IP adresa, izvorišna vrata, odredišna IP adresa, odredišna vrata*}?

[illegible] ~~[illegible]~~ logični, samo ovisi da li je primatelj ili pošiljatelj

Ne

Da bi osigurao pouzdan prijenos okteta između procesa, protokol TCP koristi mehanizam potvrđivanja. Za svaki poslani oktet primatelj mora potvrditi da ga je primio. U TCP zaglavlju za to se koristi posebno polje ACK.

IMUNES ***Zadatak 15.*** U prethodno uhvaćenom prometu, utvrdite na koji se način koriste potvrde.

Potvrda ne prenosi podatak i u zastavici je druga vrijednost (0x10)

[6] *Uputa*: za generiranje TCP prometa iskoristite alat `netcat`.

Jedna od bitnih funkcija koju obavlja TCP je i tzv. *kontrola toka*. Radi se o sljedećem. Pretpostavimo da jedan proces šalje određenu količinu podataka nekom drugom procesu, te da se proces koji šalje podatke (pošiljatelj) nalazi na računalu koje je u mrežu spojeno vezom kapaciteta 100 Mbit/s, dok se proces koji prima podatke (primatelj) nalazi na računalu koje je spojeno vezom kapaciteta 10 Mbit/s. S obzirom da je pošiljatelj spojen vezom od 100 Mbit/s, on bi mogao slati podatke tom brzinom, ali ti podaci ne bi mogli biti isporučeni primatelju (istom brzinom), jer je pristup do primatelja brzine 10 Mbit/s. Kako pošiljatelj zna kojom brzinom smije slati podatke, a da ih primatelj stigne obraditi? Kontrola toka koju obavlja TCP upravo rješava ovaj problem.

Protokol TCP koristi mehanizam tzv. klizećeg prozora (*sliding window*) za kontrolu toka. Primatelj je zadužen za kontinuirano obavještavanje pošiljatelja o količini podataka koju je on trenutno u stanju obraditi. Pošiljatelj nikad neće poslati više podataka od ove količine, bez da mu primatelj potvrdi primitak podataka. Na primjer, pri uspostavi TCP veze obje strane objavljuju jedna drugoj koliko su podataka spremne primiti u danom trenutku. Ova veličina naziva se *prozor*. Svaka strana smije odjednom poslati samo toliko podataka koliko je druga strana objavila u prozoru. Nakon što je poslala tu količinu podataka, strana pošiljatelja mora čekati potvrdu da je barem jedan dio podataka primljen na odredištu. Kad dobije potvrdu, pošiljatelj može poslati dodatnu količinu podataka, ali samo onoliko novih koliko je okteta potvrđeno. Na taj se način u mreži, u svakom trenutku, nalazi najviše onoliko nepotvrđenih podataka kolika je veličina prozora. S obzirom da pošiljatelj ne smije slati nove podatke dok primatelj ne potvrdi primitak starih, primatelj diktira brzinu kojom mu (brži) pošiljatelj šalje podatke.

IMUNES

Zadatak 16. Uhvatite proizvoljan TCP promet (od jedne veze) i utvrdite veličine prozora. Mijenja li se veličina prozora često u toku trajanja TCP veze?

Ne.

Osim pouzdanog i slijednog prijenosa, protokol TCP vodi računa i o kontroli zagušenja u mreži izazvanog TCP prometom te o ravnopravnosti podjele mrežnih kapaciteta između konkurentnih TCP veza. Tematika kontrole zagušenja je vrlo složena, te neće ovdje biti razmatrana.

Zadatak 17. Primijetite da, iako su bitno različiti po svojstvima, UDP i TCP po funkcionalnosti spadaju u transportni sloj referentnog modela OSI. Objasnite zašto.

I jedan i drugi služe za transport samo za različite usluge, jedan za audio/video za koji nije potrebna potvrda, a drugi za podatke s potvrdom

IMUNES

Zadatak 18. Pokušajte prouzročiti pojavu gubitka TCP segmenata. Na koji način možete utvrditi da je došlo do gubitaka? Možete li izazvati gubitke bez podešavanja simulatora na načina da namjerno ispušta pakete?

Ne pojavljuje se segment s potvrdom u Wireshark-u i isti prozor se šalje više puta. Možemo, treba opteretiti server ☺ Bez umišljaja

IMUNES

Zadatak 19. Pokušajte identificirati promet koji pripada jednoj TCP vezi za vrijeme u kojem dolazi do gubitka paketa. Utvrdite što se tada događa s potvrdama i veličinom prozora.

Veličina prozora skače jako.