دانشکده مهندسی کامپیوتر

آزمایشگاه شبکه‌های کامپیوتری

گزارش کار آزمایش ۵

گروه ۴

علی صداقی ۹۷۵۲۱۳۷۸

دانیال بازمانده ۹۷۵۲۱۱۳۵

۱ بخش الف

در ابتدای آزمایش و به عنوان اولین قدم، پس از دانلود فایل‌های موردنظر از LMS و انتقال به ماشین مجازی با scp، ابتدا با اجرای دستور زیر، برنامه‌ها را به حالت اجرایی درمی‌آوریم:

```
chmod +x tcp/tcpclient tcp/tcpserver udp/udpclient udp/udpserver
```

سپس، با استفاده از دستور زیر، مکانیزم کنترل ازدحام TCP را به مکانیزم reno تغییر می‌دهیم.

```
sudo bash -c 'echo reno >/proc/sys/net/ipv4/tcp_congestion_control'
```

۲ بخش ب

در مرحله‌ی بعد، با تغییر موارد خواسته‌شده ازجمله تعریف هاست‌ها و روترها در فایل lab5_network.py اقدام به ساخت توپولوژی موردنظر می‌کنیم که همانطور که مشاهده می‌شود، خروجی دستور pingall موفقیت‌آمیز بودن ساخت این توپولوژی را به اثبات می‌رساند.

۳ بخش ب-۱

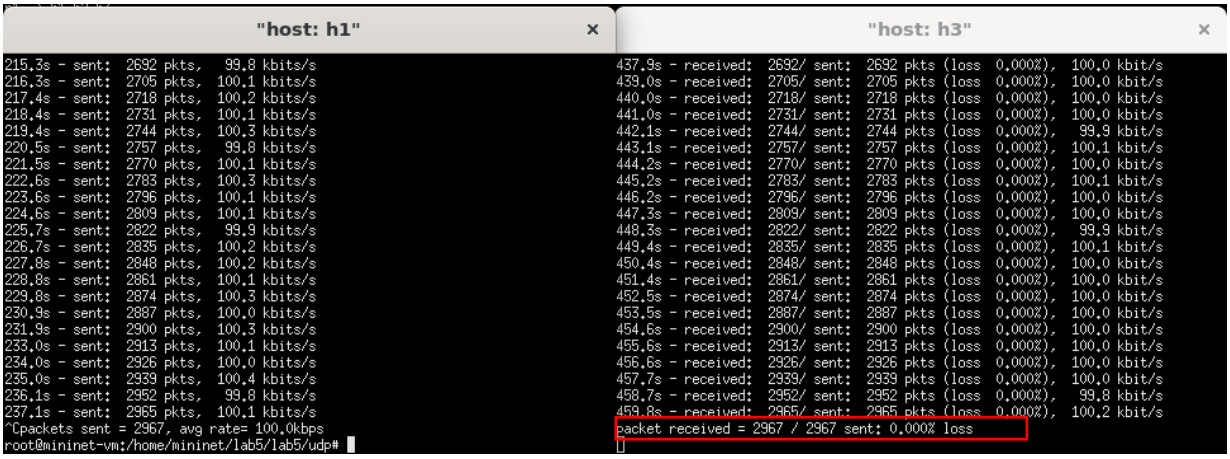
در مرحله‌ی بعد، با استفاده از دستورات زیر در دایرکتوری مربوط به UDP، یک سرور از جنس UDP در h3 بالا می‌آوریم که روی پورت 10000 گوش کند.

```
./udpserver 10000
```

۴ سوال ۱

با استفاده از دستور زیر، روی ماشین h1 یک کلاینت UDP اجرا می‌کنیم که برای سرور h3 داده‌ها را با نرخ 100Kbps ارسال می‌کند.

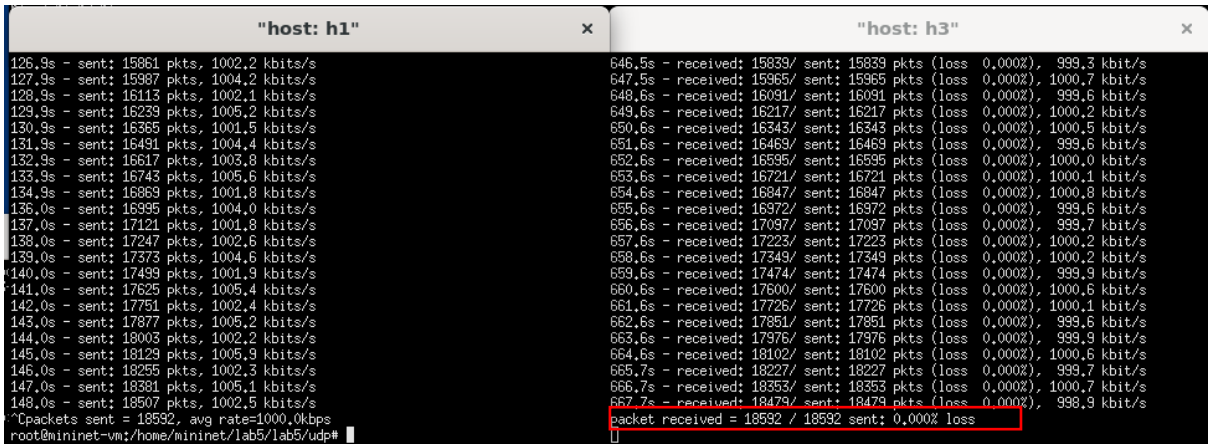
./udpclient 10.10.1.0 10000 100



همانطور که مشاهده می‌شود، احتمال loss با این rate صفر درصد و مقدار goodput حدود 100Kbps است.

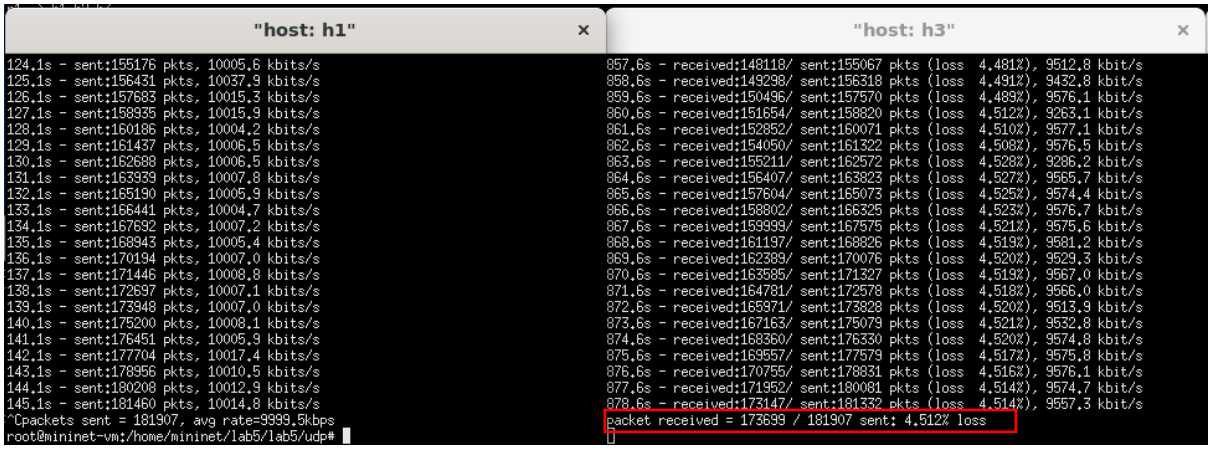
۵ سوال ۲

آزمایش قبلی را ۴ بار دیگر به ترتیب با نرخ‌های 1Mbps، 10Mbps، 100MBps و 1Gbps اجرا می‌کنیم.



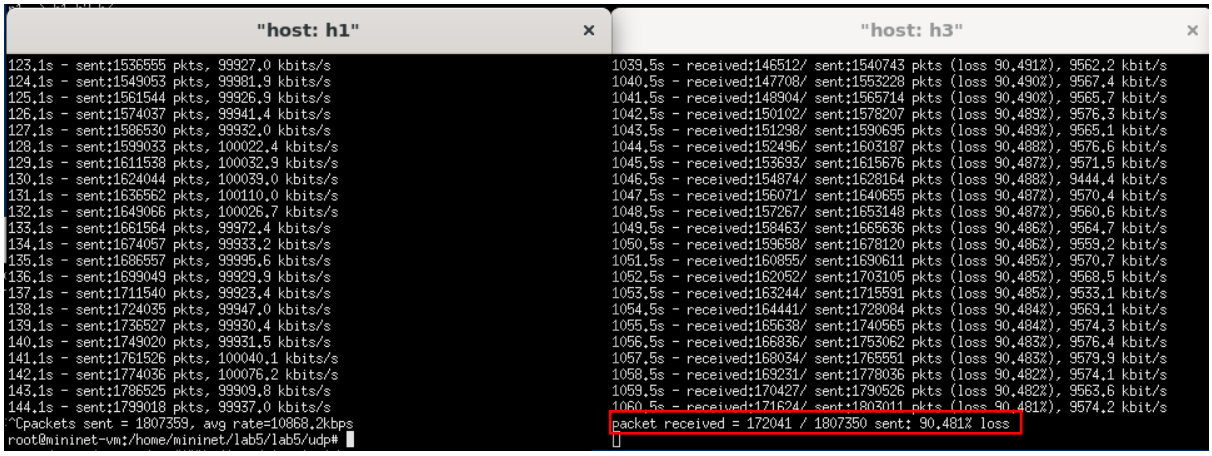
اجرای آزمایش با نرخ 1Mbps

مجددا احتمال loss صفر درصد به دست آمده است.



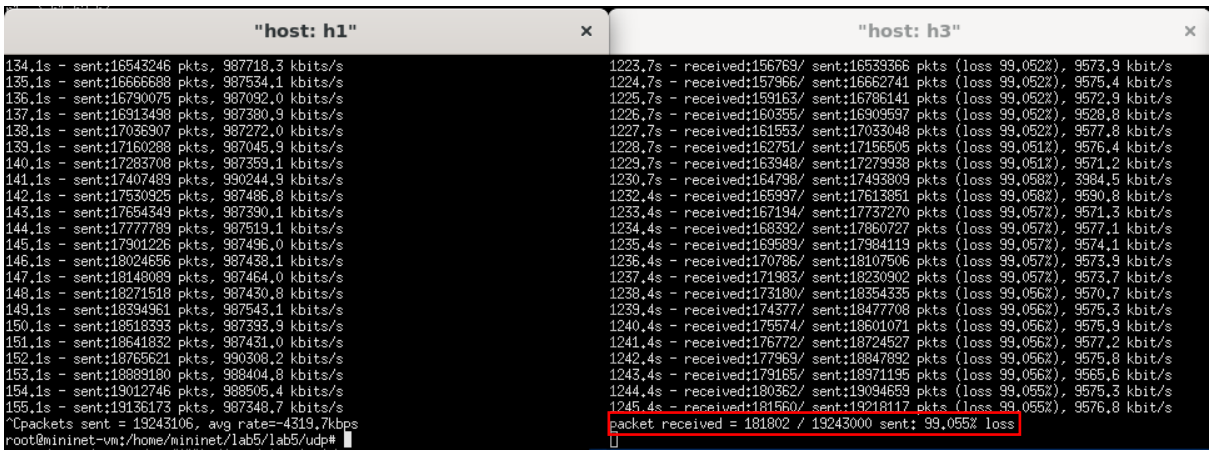
اجرای آزمایش با نرخ 10Mbps

احتمال loss افزایش پیدا کرده و به 4.5 درصد رسیده است.



اجرای آزمایش با نرخ 100Mbps

احتمال loss افزایش پیدا کرده و به 90 درصد رسیده است.



اجرای آزمایش با نرخ 1Gbps

احتمال loss افزایش پیدا کرده و به 99 درصد رسیده است.

همانطور که در تصاویر مشهود است، به ازای نرخ 10Mbps و بیشتر، احتمال loss بالای یک درصد به دست آمده است.

دلیل اصلی این قضیه را در کد پایتون می‌توان در دستور زیر دانست.

link_r1sw2.intf1.config(bw=10, enable_red=True , enable_ecn=True)

پارامتر bw مخفف عبارت bandwitch به معنای پهنای باند است که برحسب Mbps داده می‌شود. با توجه به اینکه مقدار 10Mbps را به عنوان پهنای باند داریم، پس ارسال داده‌هایی با نرخ بالاتر از این مقدار باعث packet loss خواهند شد.

۶ سوال ۳

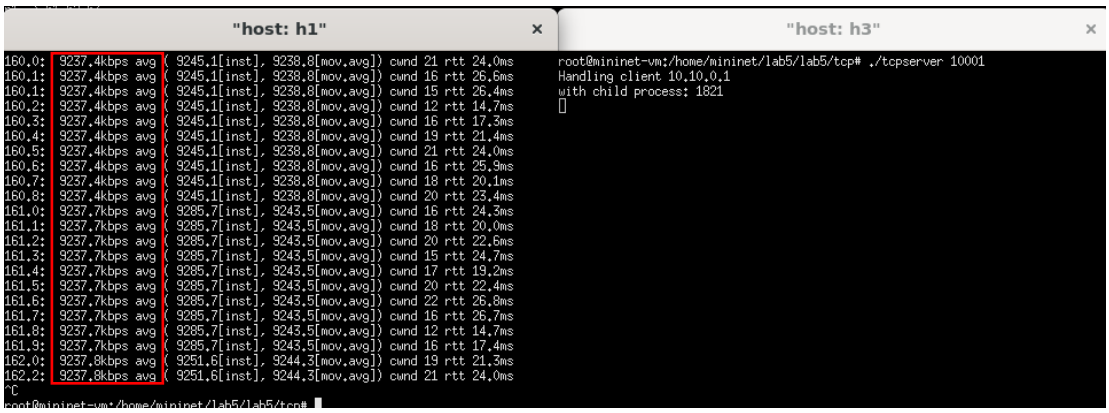
در مرحله‌ی بعد، با استفاده از دستورات زیر در دایرکتوری مربوط به TCP، یک سرور از جنس UDP در h3 بالا می‌آوریم که روی پورت 10001 گوش کند.

./tcpserver 10001

با استفاده از دستور زیر، روی ماشین h1 یک کلاینت UDP اجرا می‌کنیم که برای سرور h3 داده‌ها را ارسال می‌کند.

./udpclient 10.10.1.0 10001

مقداری صبر می‌کنیم تا به حالت نسبتا پایداری برسیم.

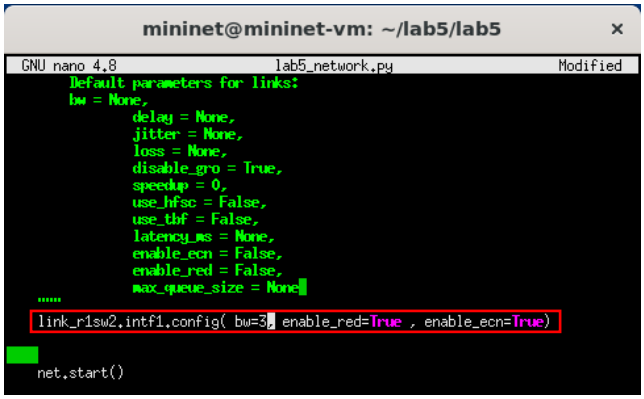


مقدار goodput برای این ارتباط به صورت میانگین 9237Kbps می‌باشد.

۷ بخش ب-۲

در این مرحله اقدام به محدودسازی پهنای باند اینترفیس eth1 به 3Mbps می‌کنیم. این کار را با استفاده از دستور زیر انجام می‌دهیم.

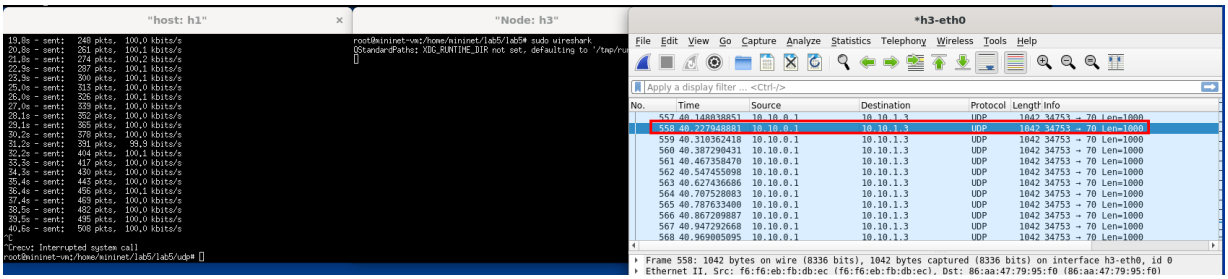
link.r1sw2.intf1.config(bw=3)



توپولوژی قبلی را clean-up کرده و مجددا فایل پایتون را اجرا می‌کنیم تا تغییرات اعمال شود.

۸ سوال ۴

همانطور که مشاهده می‌شود، در ستون length مقدار فریم‌های Ethernet مقدار 1042 بایت است که با مقدار تئوری همخوانی دارد.



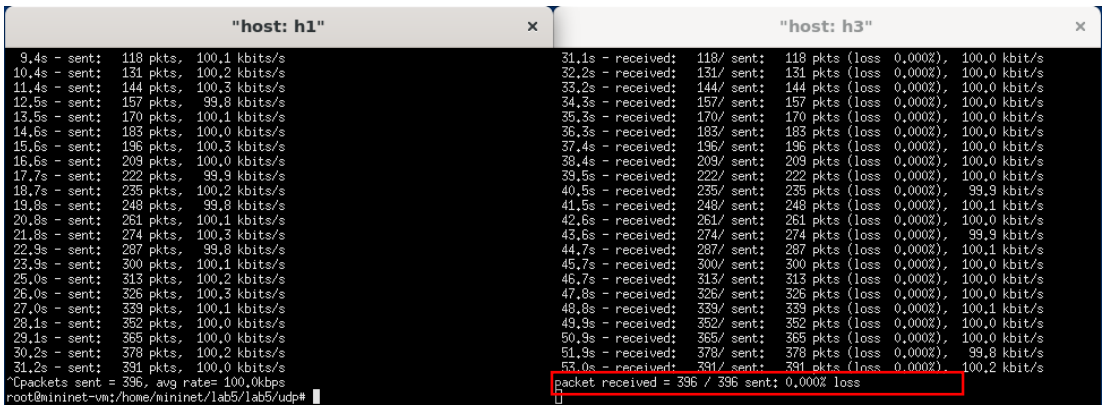
۹ سوال ۵

برای محاسبه‌ی حداکثر مقدار قابل دستیابی برای گذردهی داده‌های کاربردی (goodput) داریم:

$$goodput = bw \times \frac{1000}{1042} = 3 \times 0.96 = 2.88 \ \ Mbps$$

۱۰ سوال ۶

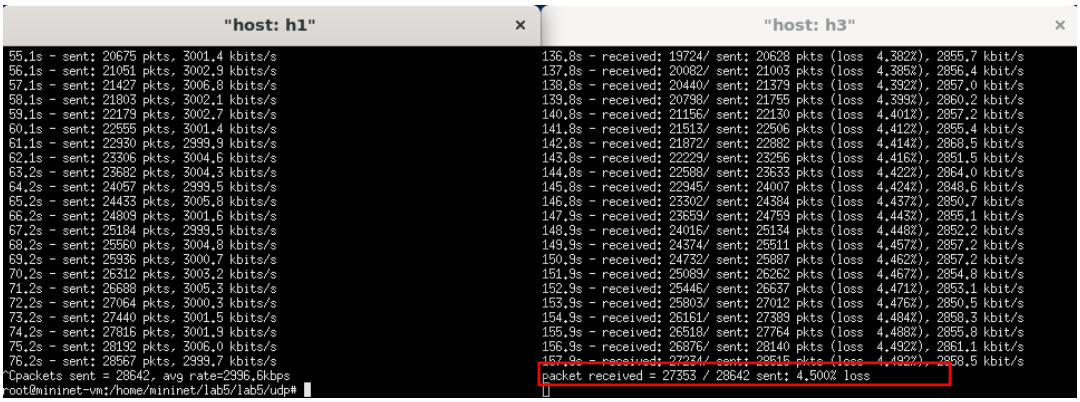
پس از راه‌اندازی مجدد سرور UDP با پورت 10000 روی h3 و کلاینت UDP روی h1 با نرخ 100Kbps داریم:



همانطور که مشاهده می‌شود، مقدار loss صفر و مقدار goodput حدود 100Kbps می‌باشد.

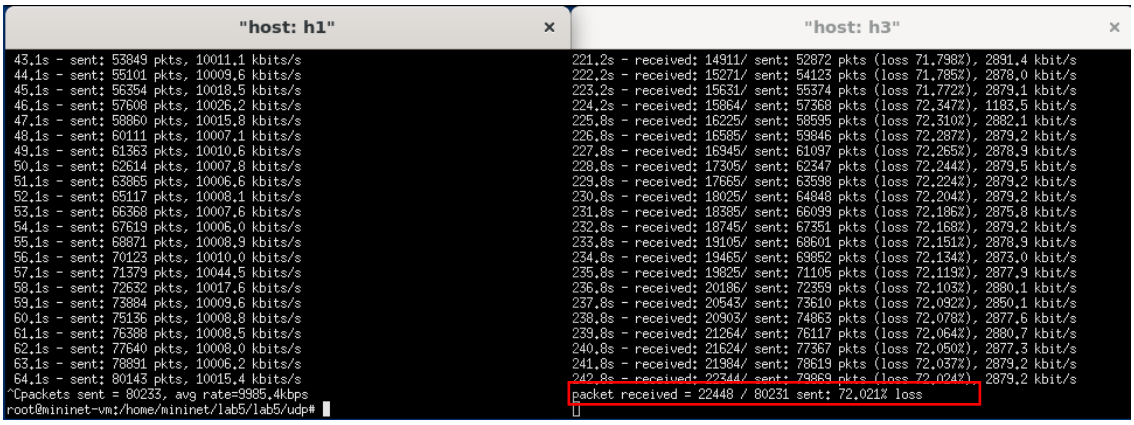
۱۱ سوال ۷

پس از اجرای عملیات بالا در نرخ‌های 3Mbps و 10Mbps داریم:



اجرای آزمایش با نرخ 3Mbps

همانطور که مشاهده می‌شود، مقدار goodput حدود بازه‌ی 2850-2860 است که بسیار به عددی که به دست آوردیم (2.88 Mbps) نزدیک است و تقریبا همخوانی دارد چون مقدار packet loss کم است.



اجرای آزمایش با نرخ 10Mbps

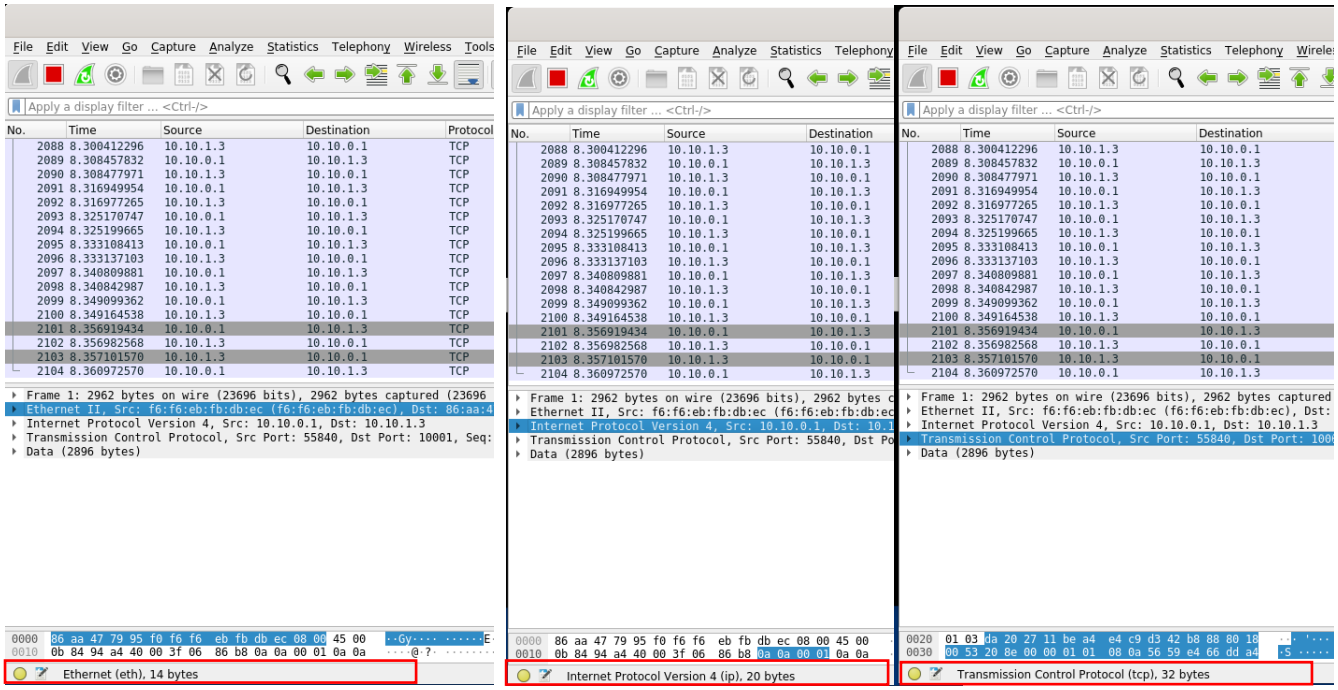
همانطور که مشاهده می‌شود، مقدار goodput حدود بازه‌ی 2870-2880 است که اگر از نظر تئوری بخواهیم محاسبه کنیم، داریم:

$$goodput = bw \times \frac{1000}{1042} = 10 \times 0.96 = 9.6 \ \ Mbps$$

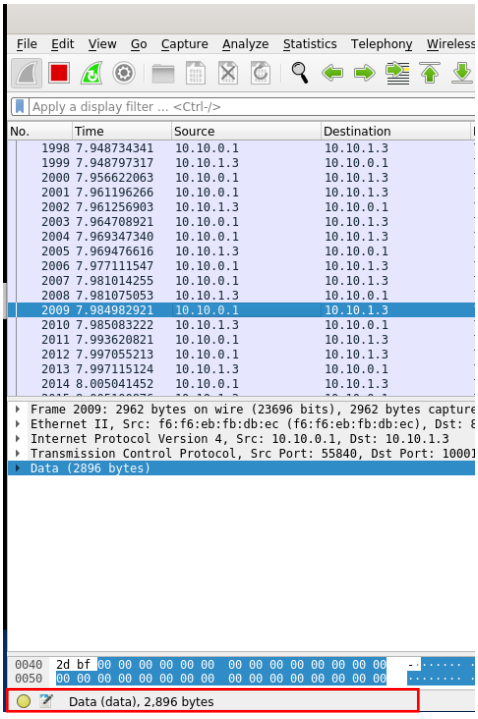
عددی که به طور تئوری به دست آمده است، بسیار با چیزی که در عمل می‌بینیم فاصله دارد که علت آن این است که حجم بسیار زیادی از داده‌ها و بسته‌ها loss می‌شوند. (نرخ packet loss حدود ۷۲ درصد است.)

۱۲ سوال ۸

با گوش دادن به بسته‌ها در سمت سرور درستی فرضیه را به طور عملی اطمینان پیدا می‌کنیم.
همانطور که در تصاویر زیر مشاهده می‌شود، مقدار هدر IP برابر 20 بایت، مقدار هدر اترنت برابر 14 بایت و مقدار هدر TCP برابر 32 بایت است.



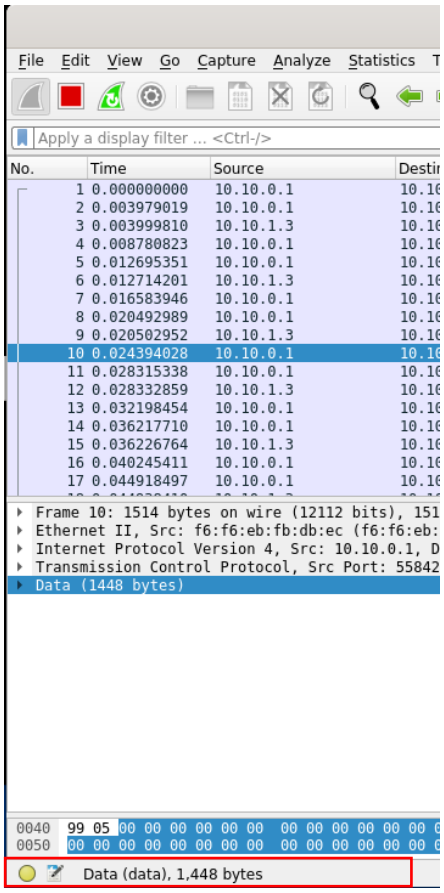
اما این قضیه درمورد data صدق نمی‌کند و مشاهده می‌کنیم مقدار Data برابر 2896 بایت می‌باشد.



با توجه به توضیحات داخل گزارش کار، این موضوع به دلیل فعال بودن TCP Large Segment Offload است که برای غیرفعال کردن آن از دستور زیر استفاده می‌کنیم.

```
ethtool -K h1-eth0 tx off sg off tso off
```

پس از اجرای دستور زیر، مجددا در Wireshark به بررسی بسته‌ها می‌پردازیم. همانطور که مشاهده می‌شود، مشکل مربوطه برطرف شده است و هدر Data هم مقدار درست یعنی بایت را نمایش می‌دهد.



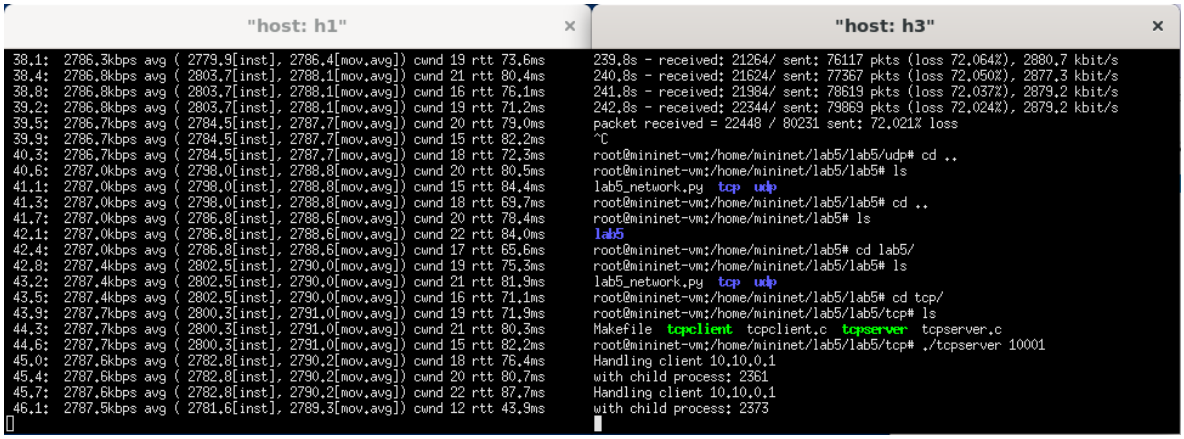
۱۳ سوال ۹

برای محاسبه‌ی مقدار goodput به صورت تئوری در پروتکل TCP به این صورت باید عمل کنیم که مقدار پهنای باند را در حاصل تقسیم اندازه‌ی داده‌های کاربردی بر اندازه‌ی کل فریم اترنتی به دست آوریم. همانطور که می‌دانیم، اندازه‌ی داده‌های کاربردی 1448 و اندازه‌ی کل فریم اترنتی 1514 بایت به دست آمده است. با درنظر گرفتن 3Mbps به عنوان پهنای باند داریم:

$$goodput = bw \times \frac{1448}{1514} = 3 \times 0.956 = 2.86 \ \ Mbps$$

۱۴ سوال ۱۰

پس از راه‌اندازی مجدد سرور TCP با پورت 10001 روی h3 و کلاینت TCP روی h1 داریم:



همانطور که مشاهده می‌شود، مقدار Goodput در این ارتباط حدود 2787Kbps به دست آمده است که بسیار نزدیک به مقدار تئوری (2.86 Mbps) است. دلیل آن هم این است که مقدار packet loss در آزمایش انجام‌شده بسیار کم است و حداکثر بسته‌ها ارسال می‌شوند.