

Homework 4

Team 23

Lamacchia Anna Lucia, Concari Laura, Di Chiara Alessandro, Longiarù Gianmarco

6 Novembre, 2022

Assunzioni

1) Per risolvere l'errore

```
"cond=\"m_ecmpRootExists.size()<=1\", msg=\"Assumed there is at most  
one exit from the root to this vertex\"."
```

abbiamo commentato l'assert a riga 325 e 326 nel file con percorso

```
-/ns-3-dev-git/src/internet/model/global-route-manager-impl.cc
```

2) (A2) Per ricostruire il percorso dei pacchetti attraverso la rete, per tutte le configurazioni, abbiamo abilitato il packet tracing PCAP utilizzando il metodo EnablePcapAll della classe PointToPointHelper su tutti i nodi, in modo da tracciare i nodi traversati dai pacchetti.

3) (C01) Per il calcolo del throughput istantaneo ci siamo basati sui dati forniti da Wireshark. Come consigliato dalla prof.ssa Cuomo in aula, per la *soluzione 1* abbiamo scelto arbitrariamente una coppia di pacchetti da analizzare e confrontare, in modo da verificare l'andamento variabile nel tempo del throughput. Per la *soluzione 2* abbiamo usato la formula data a lezione dal prof. Polverini che calcola l'efficienza di una connessione TCP nell'ipotesi di overhead nullo e di assenza di ritrasmissioni.

A1: Individuare le varie topologie note che compongono la rete

La rete è composta da due lan collegate da una serie di nodi interconnessi tra loro tramite link Point-to-Point.

Le LAN hanno un tipologia di struttura a bus, in cui tutti i nodi della rete sono collegati per mezzo di una trasmissione comune che ha esattamente due endpoint. In particolare tutti i dati trasmessi tra i nodi della rete vengono propagati su questo mezzo comune e possono essere ricevuti contemporaneamente da tutti i nodi.

Nella struttura che collega le due reti LAN riconosciamo una topologia mesh, o a maglia parzialmente connessa, in cui, dati N nodi, viene utilizzato solo un sottoinsieme di tutti i collegamenti diretti definibili tra i nodi.

A2: Ricostruzione del percorso dei pacchetti attraverso la rete di tutti i flussi simulati

Configurazione 0

TCP n8 ↔ n0

Vengono inviati 1498 pacchetti dal nodo n8 al nodo n0, evidenziati applicando il filtro `tcp and ip.src==192.148.2.3`, per selezionare solo i pacchetti basati sul protocollo TCP ed inviati dal nodo n8 con indirizzo IP 192.148.2.3.

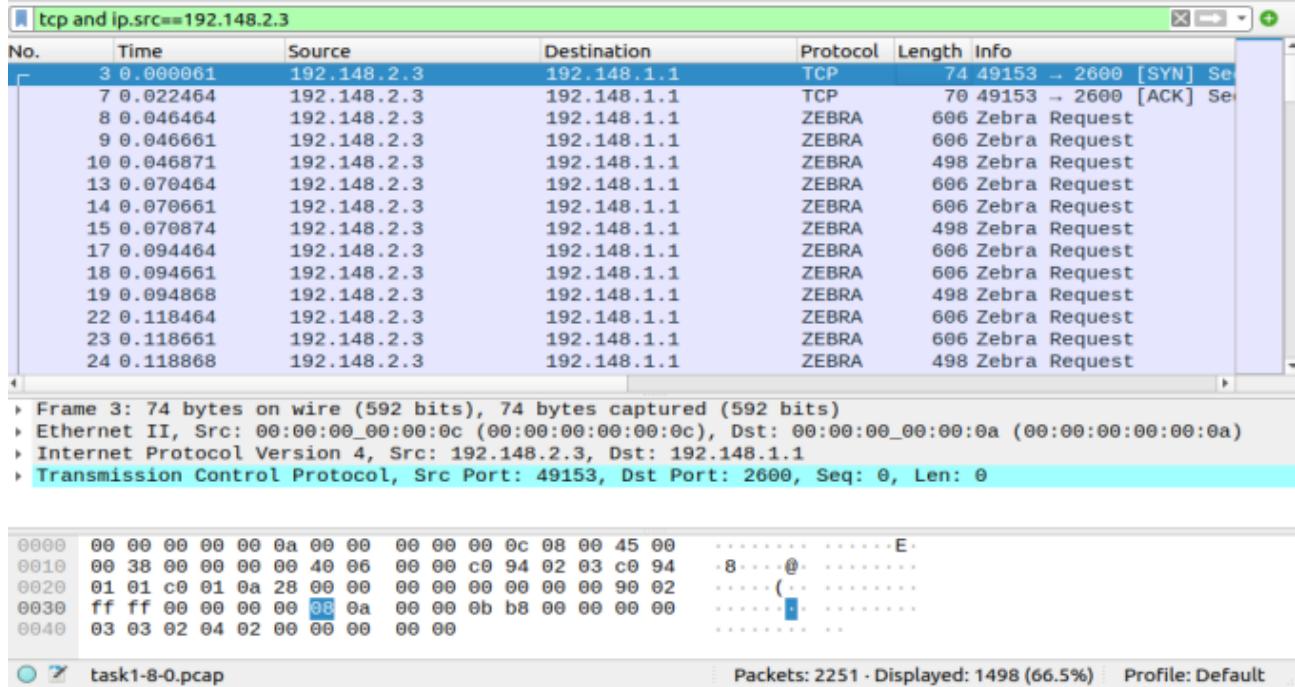


Fig. 2.1 - File task1-8-0.pcap, notare Packets totali e Displayed

Tutti questi pacchetti attraversano il canale LAN per raggiungere il nodo n6, per poi essere inviati sul nodo n4 o n5. Su n4 vengono inviati 0 pacchetti, su n5 ne vengono inviati 1498.



Fig. 2.2 - File task1-4-0.pcap, notare Packets totali e Displayed

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.148.2.3	192.148.1.1	TCP	58	49153 → 2600 [SYN] Seq: 0
2	0.022400	192.148.2.3	192.148.1.1	TCP	54	49153 → 2600 [ACK] Seq: 1
3	0.046624	192.148.2.3	192.148.1.1	ZEBRA	590	Zebra Request
4	0.046834	192.148.2.3	192.148.1.1	ZEBRA	590	Zebra Request
5	0.046997	192.148.2.3	192.148.1.1	ZEBRA	482	Zebra Request
6	0.070624	192.148.2.3	192.148.1.1	ZEBRA	590	Zebra Request
7	0.070837	192.148.2.3	192.148.1.1	ZEBRA	590	Zebra Request
8	0.071001	192.148.2.3	192.148.1.1	ZEBRA	482	Zebra Request
9	0.094624	192.148.2.3	192.148.1.1	ZEBRA	590	Zebra Request
10	0.094831	192.148.2.3	192.148.1.1	ZEBRA	590	Zebra Request
11	0.094995	192.148.2.3	192.148.1.1	ZEBRA	482	Zebra Request
12	0.118624	192.148.2.3	192.148.1.1	ZEBRA	590	Zebra Request
13	0.118831	192.148.2.3	192.148.1.1	ZEBRA	590	Zebra Request
14	0.118999	192.148.2.3	192.148.1.1	ZEBRA	482	Zebra Request

```

Frame 1: 58 bytes on wire (464 bits), 58 bytes captured (464 bits)
Point-to-Point Protocol
Internet Protocol Version 4, Src: 192.148.2.3, Dst: 192.148.1.1
Transmission Control Protocol, Src Port: 49153, Dst Port: 2600, Seq: 0, Len: 0

0000  00 21 45 00 00 38 00 00  00 00 3e 06 00 00 c0 94  -!E..8...>.....
0010  02 03 c0 94 01 01 c0 01  0a 28 00 00 00 00 00 00  ....(.....
0020  00 00 90 02 ff ff 00 00  00 00 08 0a 00 00 0b b8  .....
0030  00 00 00 00 03 03 02 04  02 00  .....

```

Fig. 2.3 - File task1-5-0.pcap, notare Packets totali e Displayed

I pacchetti raggiungono poi il nodo n3, passano ad n2 e attraversano il canale LAN per arrivare a destinazione nel nodo n0.

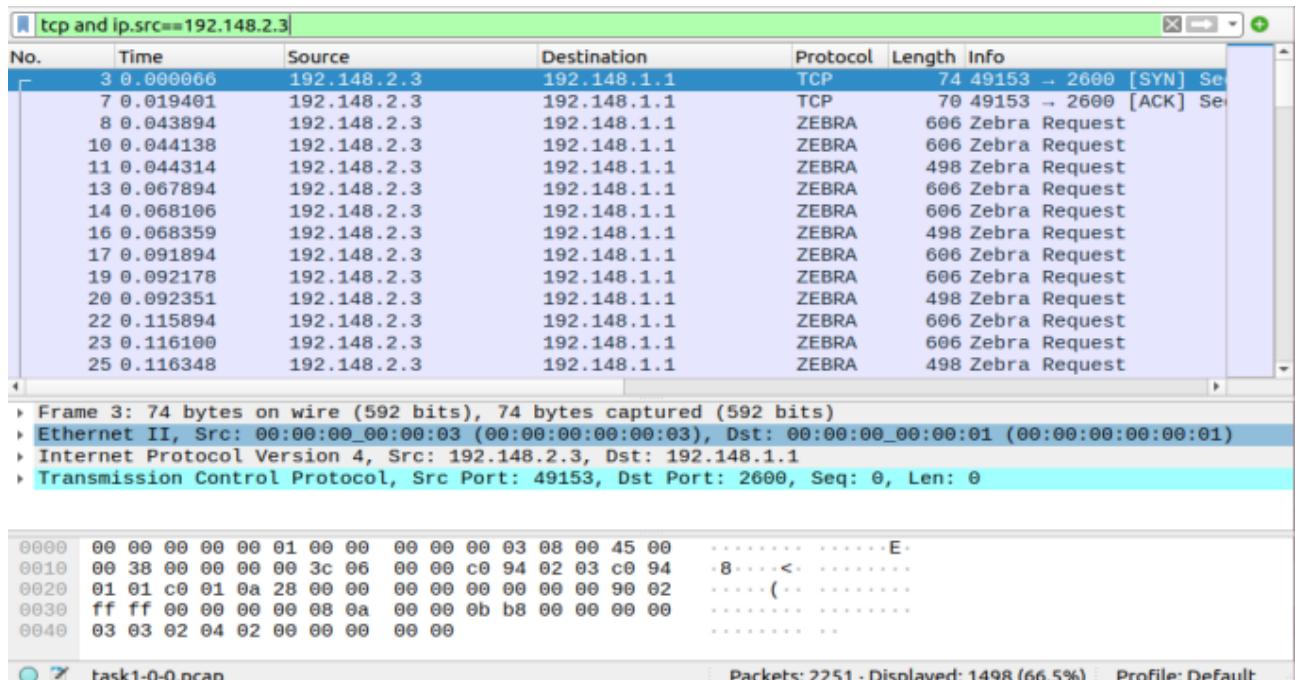


Fig. 2.4 - File task1-0-0.pcap, notare Packets totali e Displayed

Configurazione 1

TCP n8 ↔ n0

Vengono inviati 1249 pacchetti dal nodo n8 al nodo n0, evidenziati applicando il filtro `tcp and ip.src==192.148.2.3`, per selezionare solo i pacchetti basati sul protocollo TCP inviati dal nodo n8 con indirizzo IP 192.168.2.3.

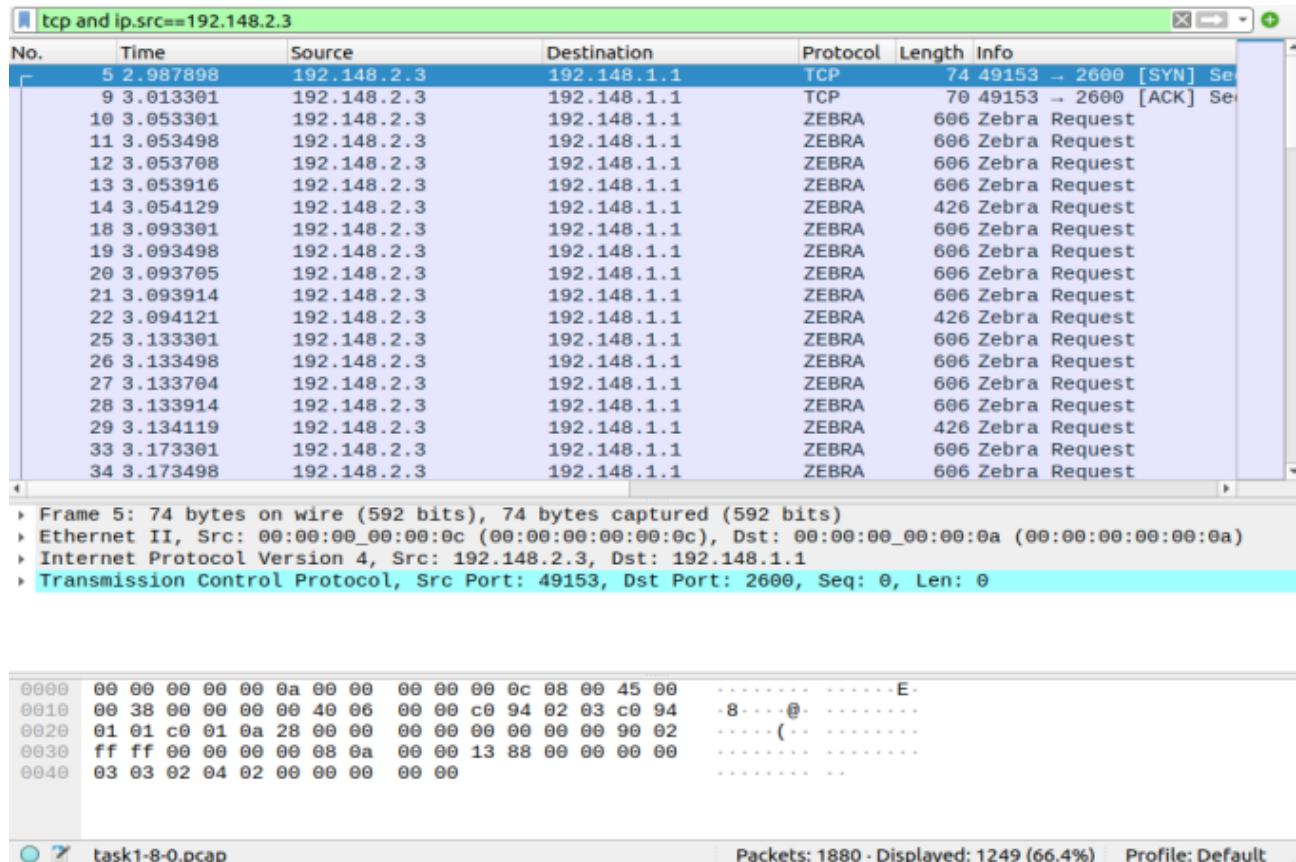


Fig. 2.5 - File task1-8-0.pcap, notare Packets totali e Displayed

I pacchetti attraversano il canale LAN per raggiungere il nodo n6, per poi essere inviati sul nodo n4 o n5. Su n4 vengono inviati 0 pacchetti, su n5 ne vengono inviati 1249.

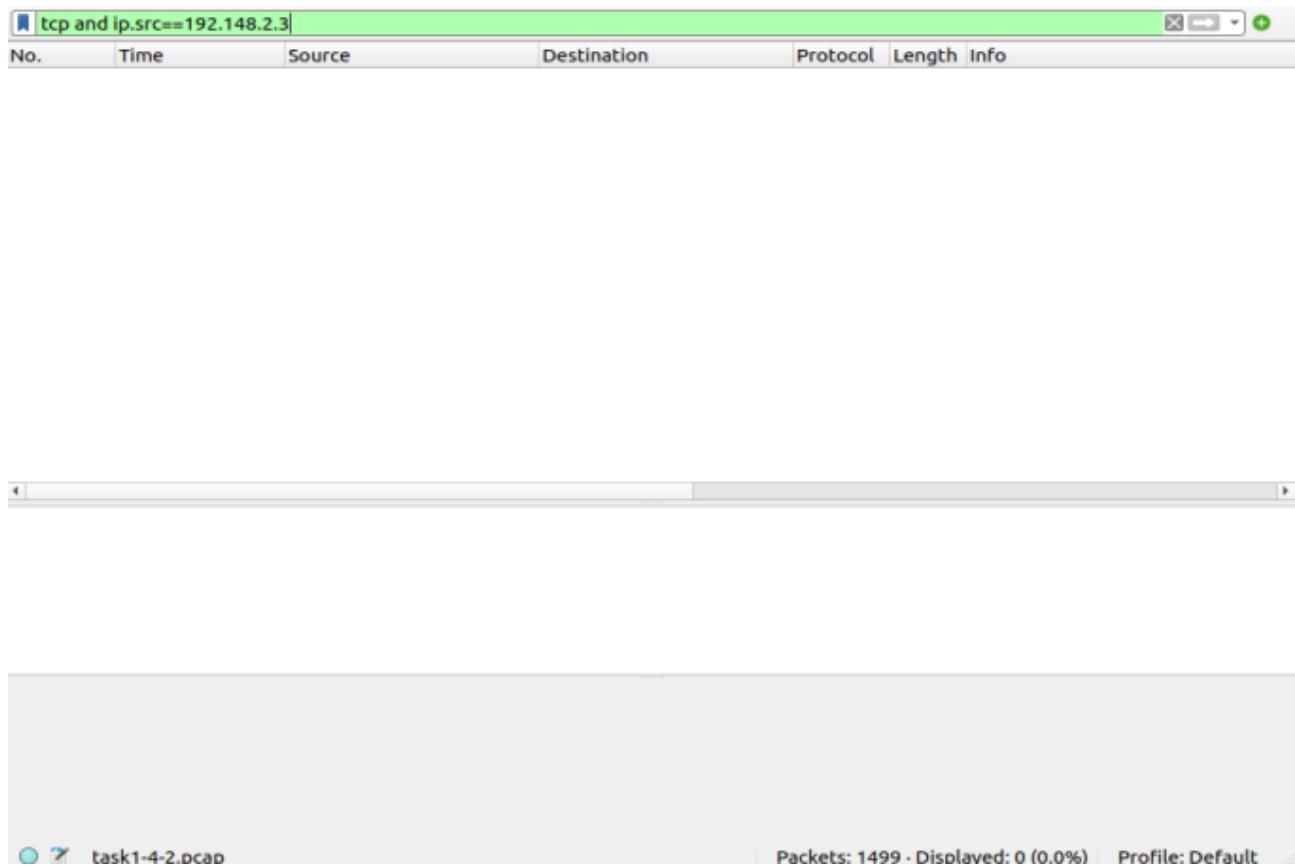


Fig. 2.6 - File task1-4-2.pcap, notare Packets totali e Displayed

No.	Time	Source	Destination	Protocol	Length	Info
187	2.985771	192.148.2.3	192.148.1.1	TCP	58	49153 → 2600 [SYN] Seq: 1
188	3.011171	192.148.2.3	192.148.1.1	TCP	54	49153 → 2600 [ACK] Seq: 2
194	3.051395	192.148.2.3	192.148.1.1	ZEBRA	590	Zebra Request
195	3.051605	192.148.2.3	192.148.1.1	ZEBRA	590	Zebra Request
196	3.051813	192.148.2.3	192.148.1.1	ZEBRA	590	Zebra Request
197	3.052026	192.148.2.3	192.148.1.1	ZEBRA	590	Zebra Request
198	3.052159	192.148.2.3	192.148.1.1	ZEBRA	410	Zebra Request
199	3.091395	192.148.2.3	192.148.1.1	ZEBRA	590	Zebra Request
200	3.091602	192.148.2.3	192.148.1.1	ZEBRA	590	Zebra Request
201	3.091811	192.148.2.3	192.148.1.1	ZEBRA	590	Zebra Request
202	3.092018	192.148.2.3	192.148.1.1	ZEBRA	590	Zebra Request
203	3.092155	192.148.2.3	192.148.1.1	ZEBRA	410	Zebra Request
209	3.131395	192.148.2.3	192.148.1.1	ZEBRA	590	Zebra Request
210	3.131601	192.148.2.3	192.148.1.1	ZEBRA	590	Zebra Request
211	3.131811	192.148.2.3	192.148.1.1	ZEBRA	590	Zebra Request
212	3.132016	192.148.2.3	192.148.1.1	ZEBRA	590	Zebra Request
213	3.132146	192.148.2.3	192.148.1.1	ZEBRA	410	Zebra Request
214	3.171395	192.148.2.3	192.148.1.1	ZEBRA	590	Zebra Request
215	3.171600	192.148.2.3	192.148.1.1	ZEBRA	590	Zebra Request

Frame 187: 58 bytes on wire (464 bits), 58 bytes captured (464 bits)
 ▶ Point-to-Point Protocol
 ▶ Internet Protocol Version 4, Src: 192.148.2.3, Dst: 192.148.1.1
 ▶ Transmission Control Protocol, Src Port: 49153, Dst Port: 2600, Seq: 0, Len: 0

```

0000  00 21 45 00 00 38 00 00  00 00 3e 06 00 00 c0 94  .!E.8...>...
0010  02 03 c0 94 01 01 c0 01  0a 28 00 00 00 00 00 00  ....(.....
0020  00 00 90 02 ff ff 00 00  00 00 08 0a 00 00 13 88  .....
0030  00 00 00 00 03 03 02 04  02 00  ..... .

```

Fig. 2.7 - File task1-5-0.pcap, notare Packets totali e Displayed

I pacchetti raggiungono poi il nodo n3, passano ad n2 e attraversano il canale LAN per arrivare a destinazione nel nodo n0.

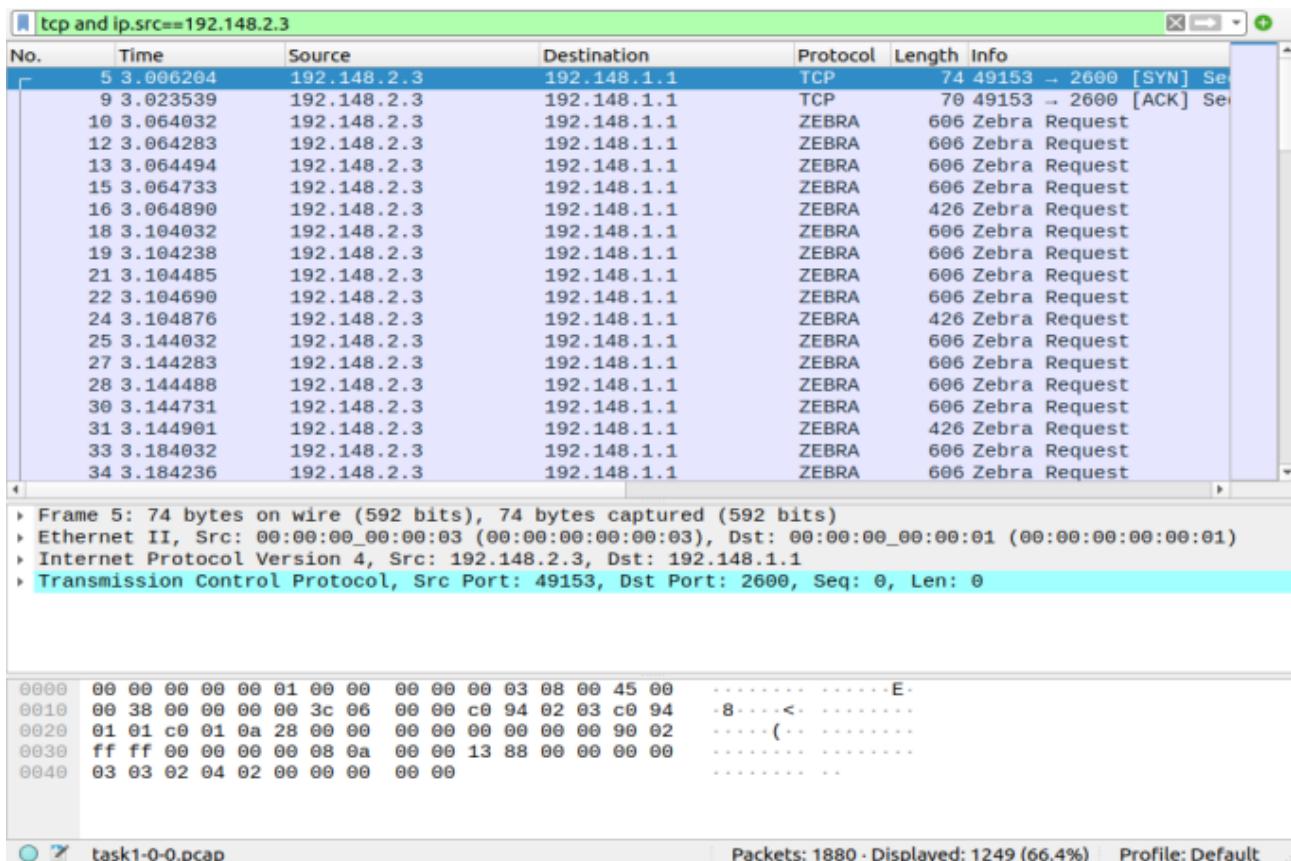


Fig. 2.8 - File task1-0-0.pcap, notare Packets totali e Displayed

TCP n1 ↔ n7

Vengono inviati 874 pacchetti dal nodo n1 al nodo n7, evidenziati applicando il filtro `tcp and ip.src==192.148.1.2`, per selezionare solo i pacchetti basati sul protocollo TCP inviati dal nodo n1 con indirizzo IP 192.168.1.2.

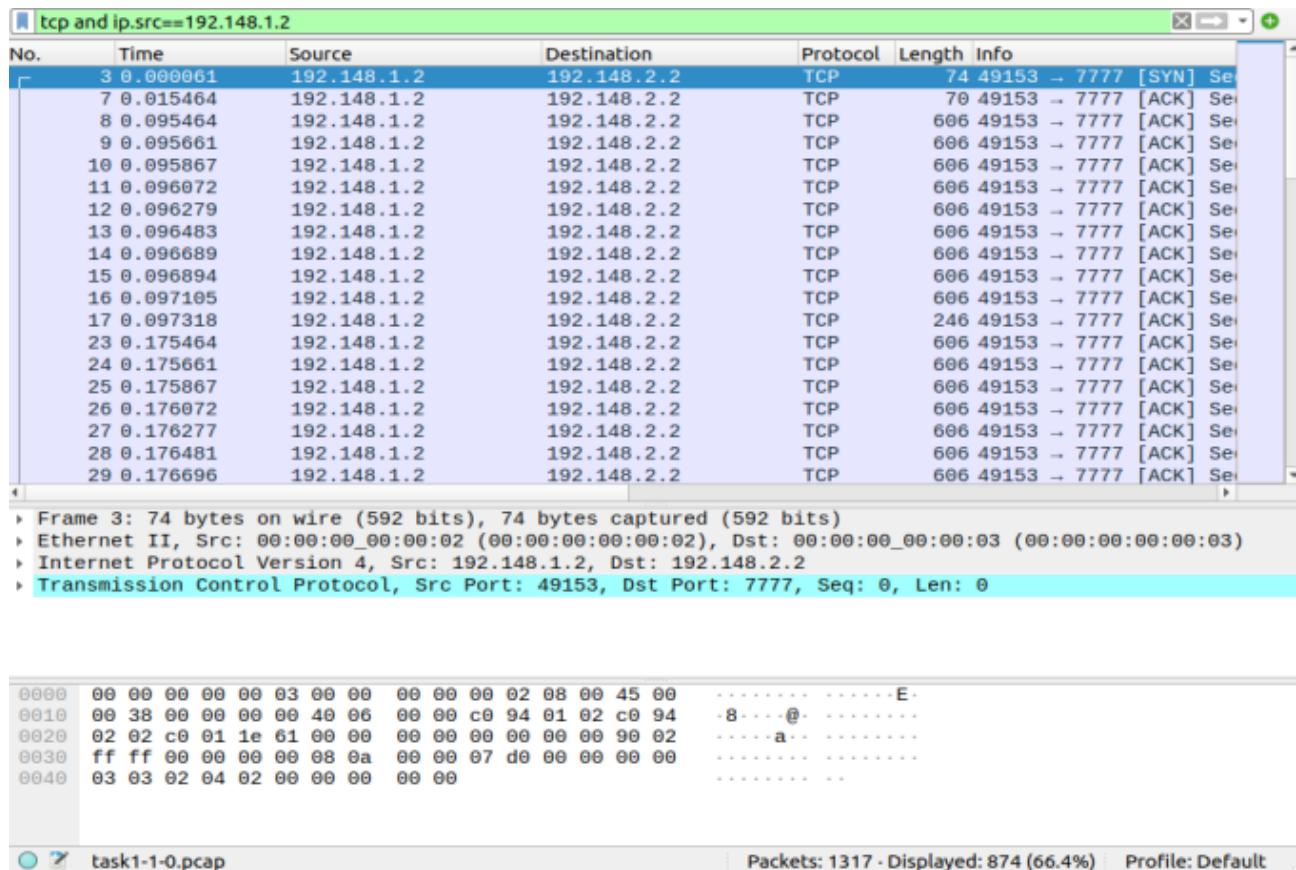


Fig. 2.9 - File task 1-1-0.pcap, notare Packets totali e Displayed

I pacchetti attraversano il canale LAN per raggiungere il nodo n2, passano ad n3 per poi essere inviati sul nodo n4 o n5. Su n4 vengono inviati 874 pacchetti, su n5 ne vengono inviati 0.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.148.1.2	192.148.2.2	TCP	58	49153 → 7777 [SYN] Seq: 0 Len: 0
2	0.015400	192.148.1.2	192.148.2.2	TCP	54	49153 → 7777 [ACK] Seq: 1 Len: 0
3	0.095667	192.148.1.2	192.148.2.2	TCP	590	49153 → 7777 [ACK] Seq: 2 Len: 0
4	0.095872	192.148.1.2	192.148.2.2	TCP	590	49153 → 7777 [ACK] Seq: 3 Len: 0
5	0.096077	192.148.1.2	192.148.2.2	TCP	590	49153 → 7777 [ACK] Seq: 4 Len: 0
6	0.096284	192.148.1.2	192.148.2.2	TCP	590	49153 → 7777 [ACK] Seq: 5 Len: 0
7	0.096489	192.148.1.2	192.148.2.2	TCP	590	49153 → 7777 [ACK] Seq: 6 Len: 0
8	0.096694	192.148.1.2	192.148.2.2	TCP	590	49153 → 7777 [ACK] Seq: 7 Len: 0
9	0.096899	192.148.1.2	192.148.2.2	TCP	590	49153 → 7777 [ACK] Seq: 8 Len: 0
10	0.097110	192.148.1.2	192.148.2.2	TCP	590	49153 → 7777 [ACK] Seq: 9 Len: 0
11	0.097324	192.148.1.2	192.148.2.2	TCP	590	49153 → 7777 [ACK] Seq: 10 Len: 0
12	0.097349	192.148.1.2	192.148.2.2	TCP	230	49153 → 7777 [ACK] Seq: 11 Len: 0
13	0.175667	192.148.1.2	192.148.2.2	TCP	590	49153 → 7777 [ACK] Seq: 12 Len: 0
14	0.175872	192.148.1.2	192.148.2.2	TCP	590	49153 → 7777 [ACK] Seq: 13 Len: 0
15	0.176077	192.148.1.2	192.148.2.2	TCP	590	49153 → 7777 [ACK] Seq: 14 Len: 0
16	0.176282	192.148.1.2	192.148.2.2	TCP	590	49153 → 7777 [ACK] Seq: 15 Len: 0
17	0.176487	192.148.1.2	192.148.2.2	TCP	590	49153 → 7777 [ACK] Seq: 16 Len: 0
18	0.176701	192.148.1.2	192.148.2.2	TCP	590	49153 → 7777 [ACK] Seq: 17 Len: 0
19	0.176916	192.148.1.2	192.148.2.2	TCP	590	49153 → 7777 [ACK] Seq: 18 Len: 0

> Frame 1: 58 bytes on wire (464 bits), 58 bytes captured (464 bits)
> Point-to-Point Protocol
> Internet Protocol Version 4, Src: 192.148.1.2, Dst: 192.148.2.2
> Transmission Control Protocol, Src Port: 49153, Dst Port: 7777, Seq: 0, Len: 0

```
0000  00 21 45 00 00 38 00 00  00 00 3e 06 00 00 c0 94  .!E..8...->....  

0010  01 02 c0 94 02 02 c0 01  1e 61 00 00 00 00 00 00  .....a.....  

0020  00 00 90 02 ff ff 00 00  00 00 08 0a 00 00 07 d0  .....  

0030  00 00 00 00 03 03 02 04  02 00  .....  


```

task1-4-0.pcap Packets: 1499 - Displayed: 874 (58.3%) Profile: Default

Fig. 2.10 - File task1-4-0.pcap, notare Packets totali e Displayed

No.	Time	Source	Destination	Protocol	Length	Info

task1-5-0.pcap Packets: 1686 - Displayed: 0 (0.0%) Profile: Default

Fig. 2.11 - File task1-5-0.pcap, notare Packets totali e Displayed

I pacchetti raggiungono poi il nodo n6 e attraversano il canale LAN per arrivare a destinazione nel nodo n7.

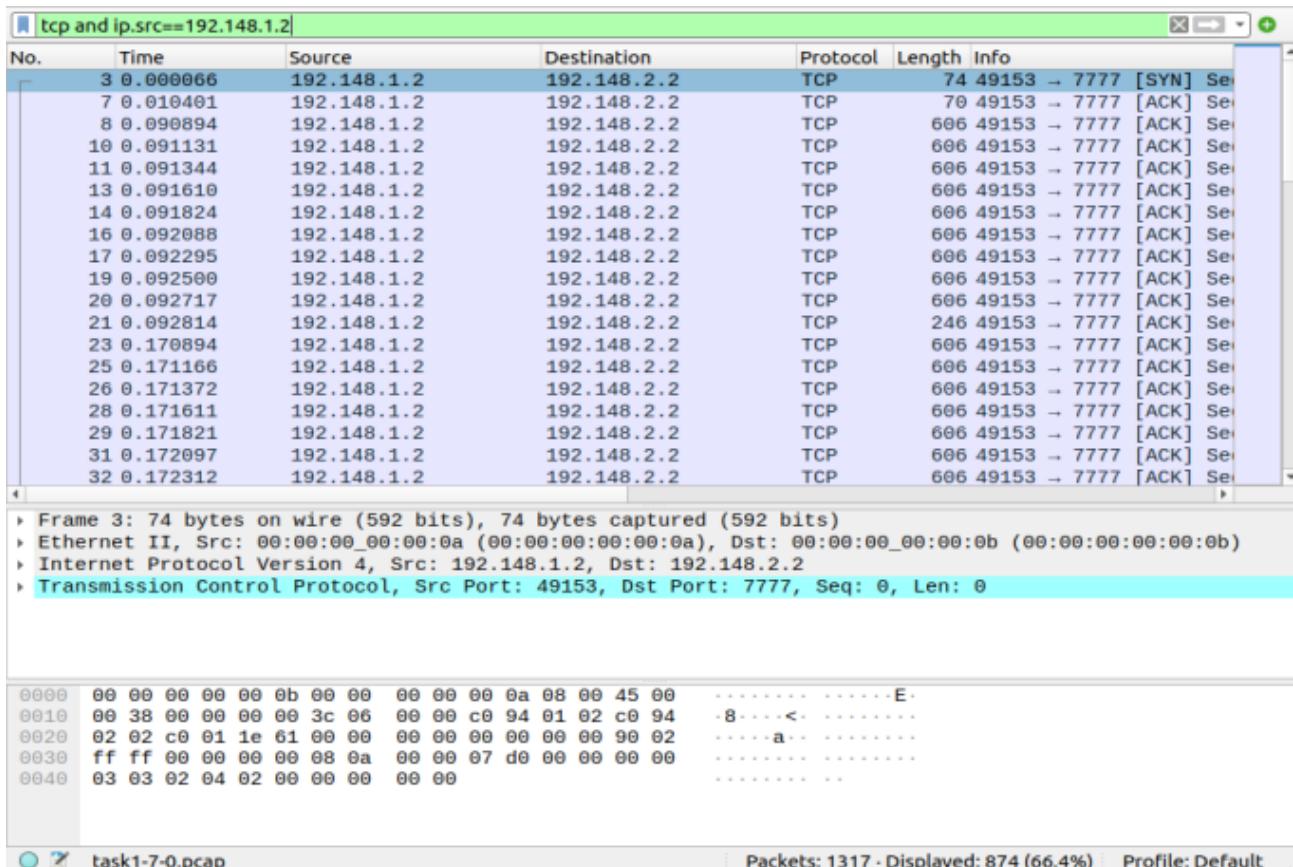


Fig. 2.12 - File task1-7-0.pcap, notare Packets totali e Displayed

Configurazione 2

UDP n8 ↔ n2

Vengono inviati 5 pacchetti dal nodo n8 al nodo n2, evidenziati applicando il filtro `udp and ip.dst==192.148.1.3`, per selezionare solo i pacchetti basati sul protocollo UDP inviati al nodo n2 con indirizzo IP 192.168.1.3.

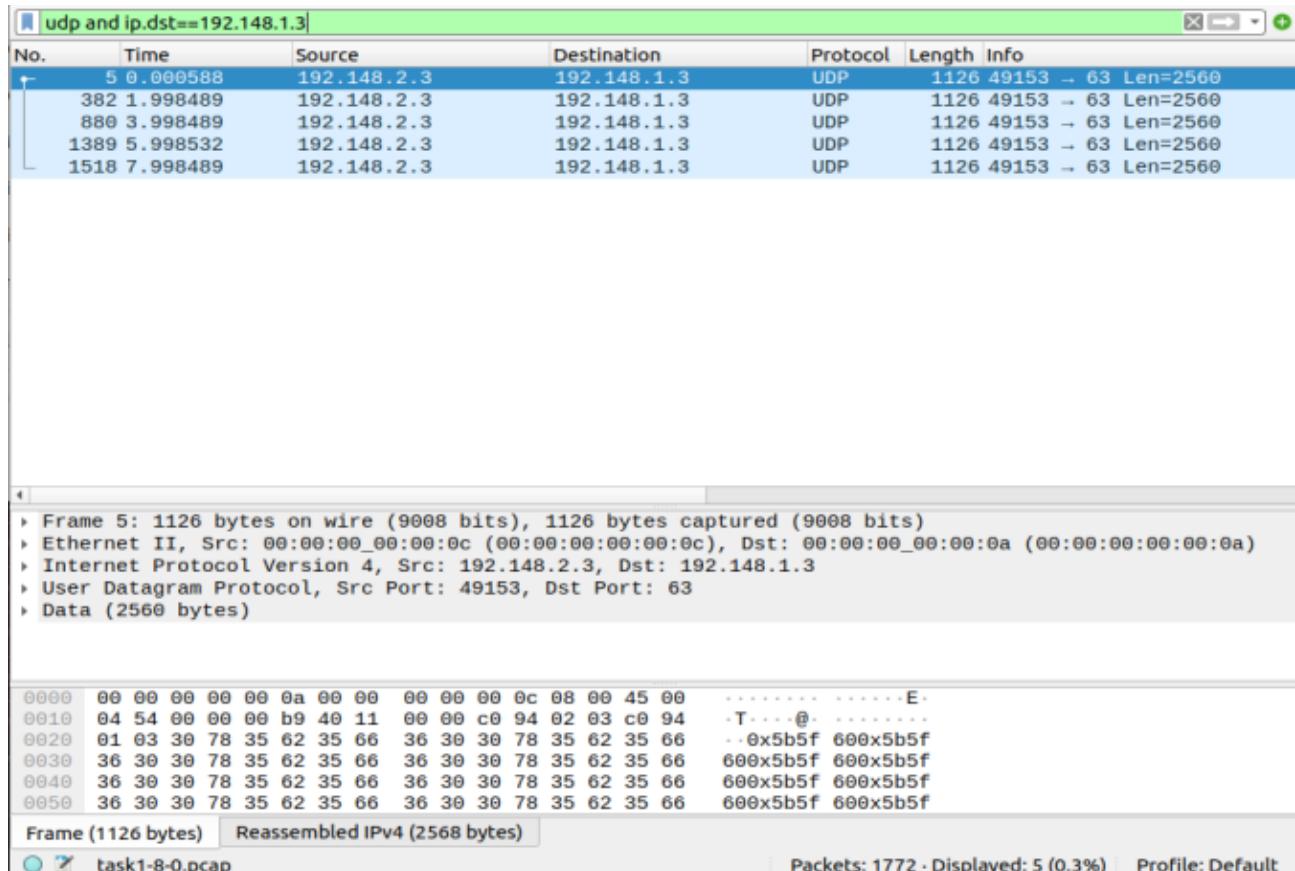


Fig. 2.13 - File task1-8-0.pcap, notare Packets totali e Displayed

I pacchetti attraversano il canale LAN per raggiungere il nodo n6, per poi essere inviati sul nodo n4 o n5. Su n4 vengono inviati 0 pacchetti, su n5 ne vengono inviati 5.

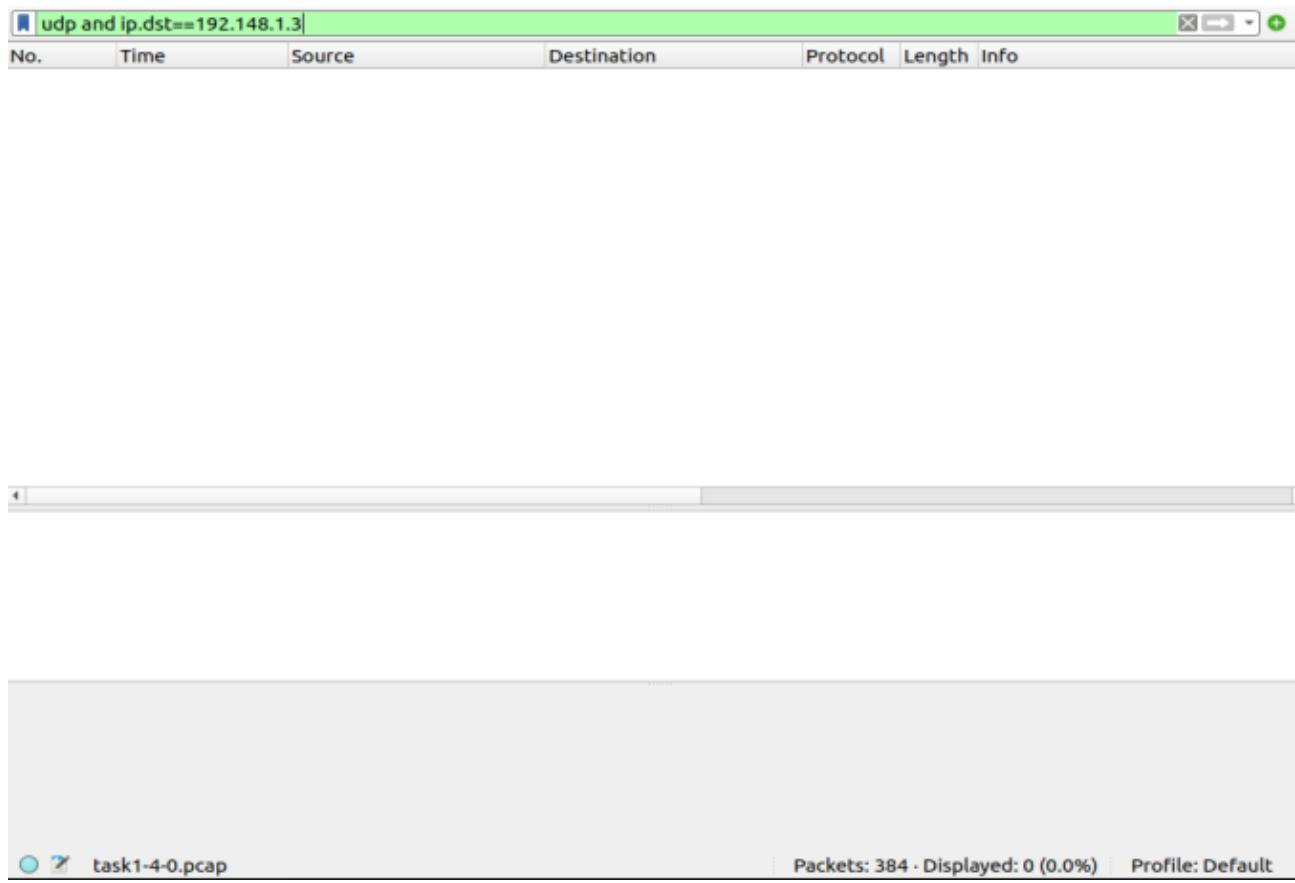


Fig. 2.14 - File task1-4-0.pcap, notare Packets totali e Displayed

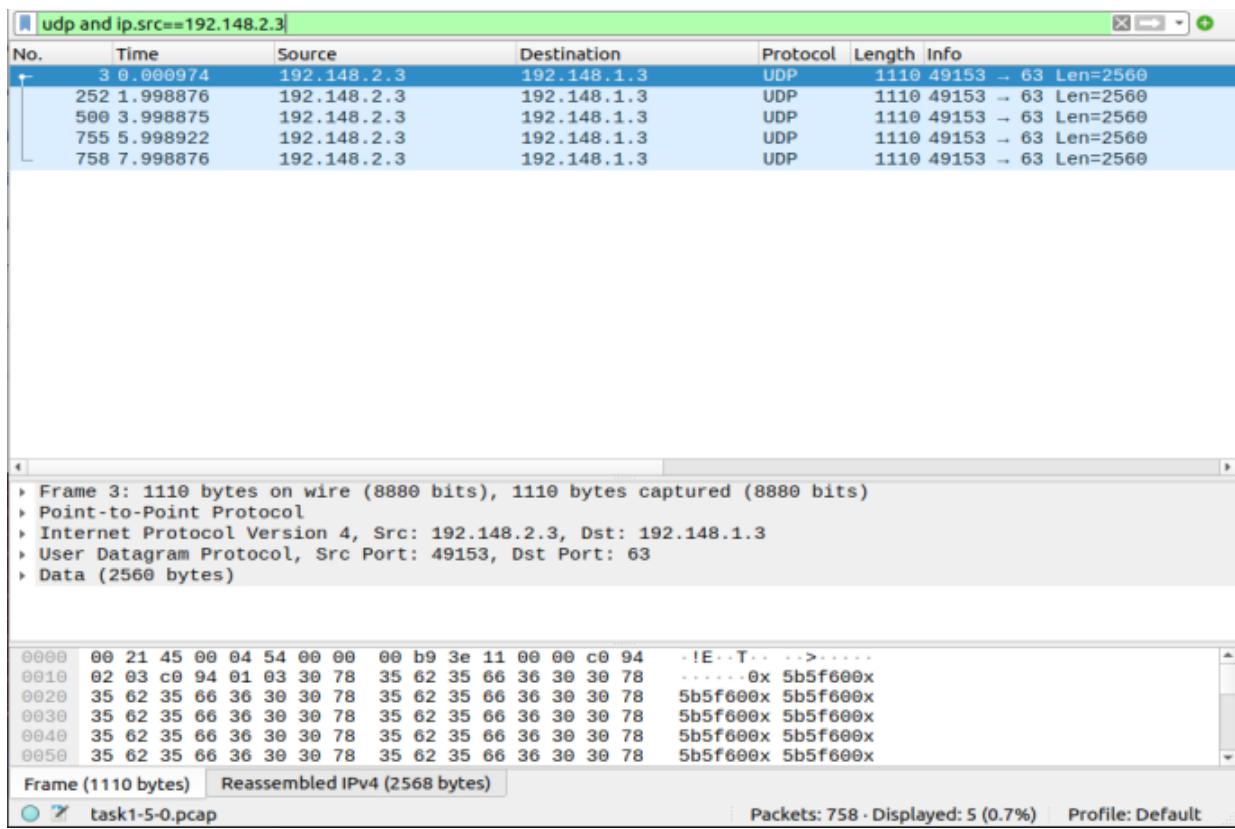


Fig. 2.15 - File task 1-5-0.pcap, notare Packets totali e Displayed

I pacchetti raggiungono poi il nodo n3 e passano ad n2. In questo caso i pacchetti non attraversano il primo bus.

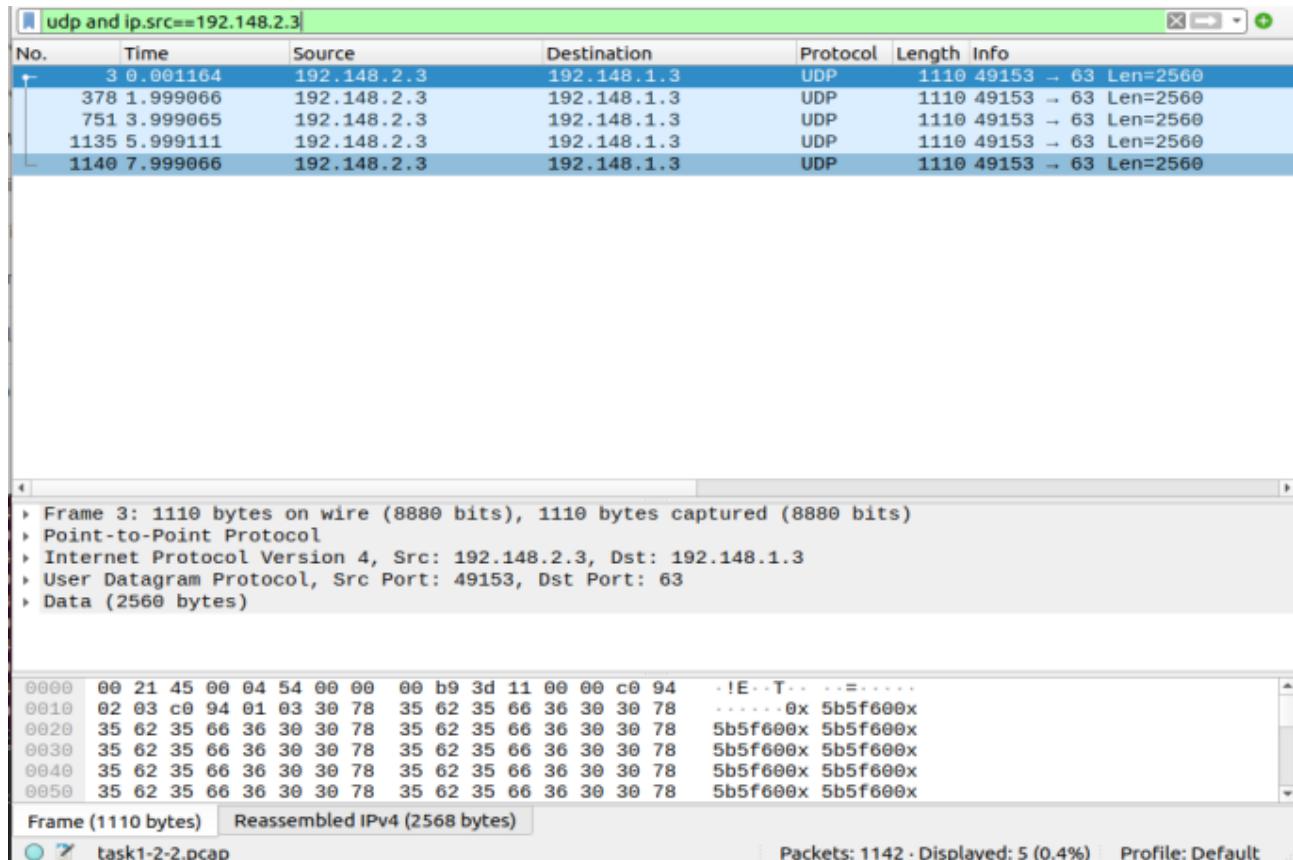


Fig. 2.16 - File task1-2-2.pcap, notare Packets totali e Displayed

TCP n8 ↔ n0

Vengono inviati 748 pacchetti dal nodo n8 al nodo n0, evidenziati applicando il filtro `tcp and ip.src==192.148.2.3`, per selezionare solo i pacchetti basati sul protocollo UDP inviati dal nodo n8 con indirizzo IP 192.168.2.3.

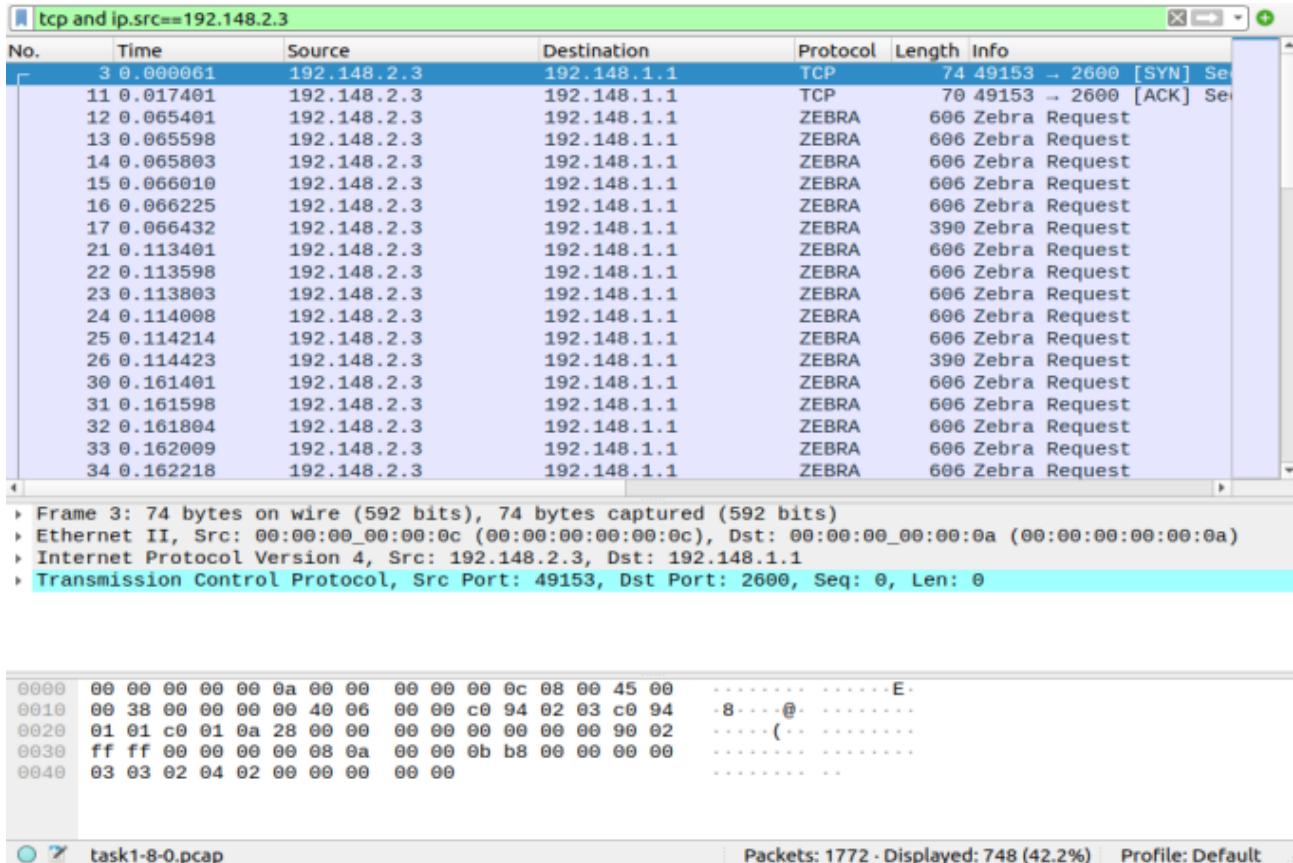


Fig. 2.17 - File task1-8-0.pcap, notare Packets totali e Displayed

I pacchetti attraversano il canale LAN per raggiungere il nodo n6, per poi essere inviati sul nodo n4 o n5. Su n4 vengono inviati 0 pacchetti, mentre su n5 ne vengono inviati 748.

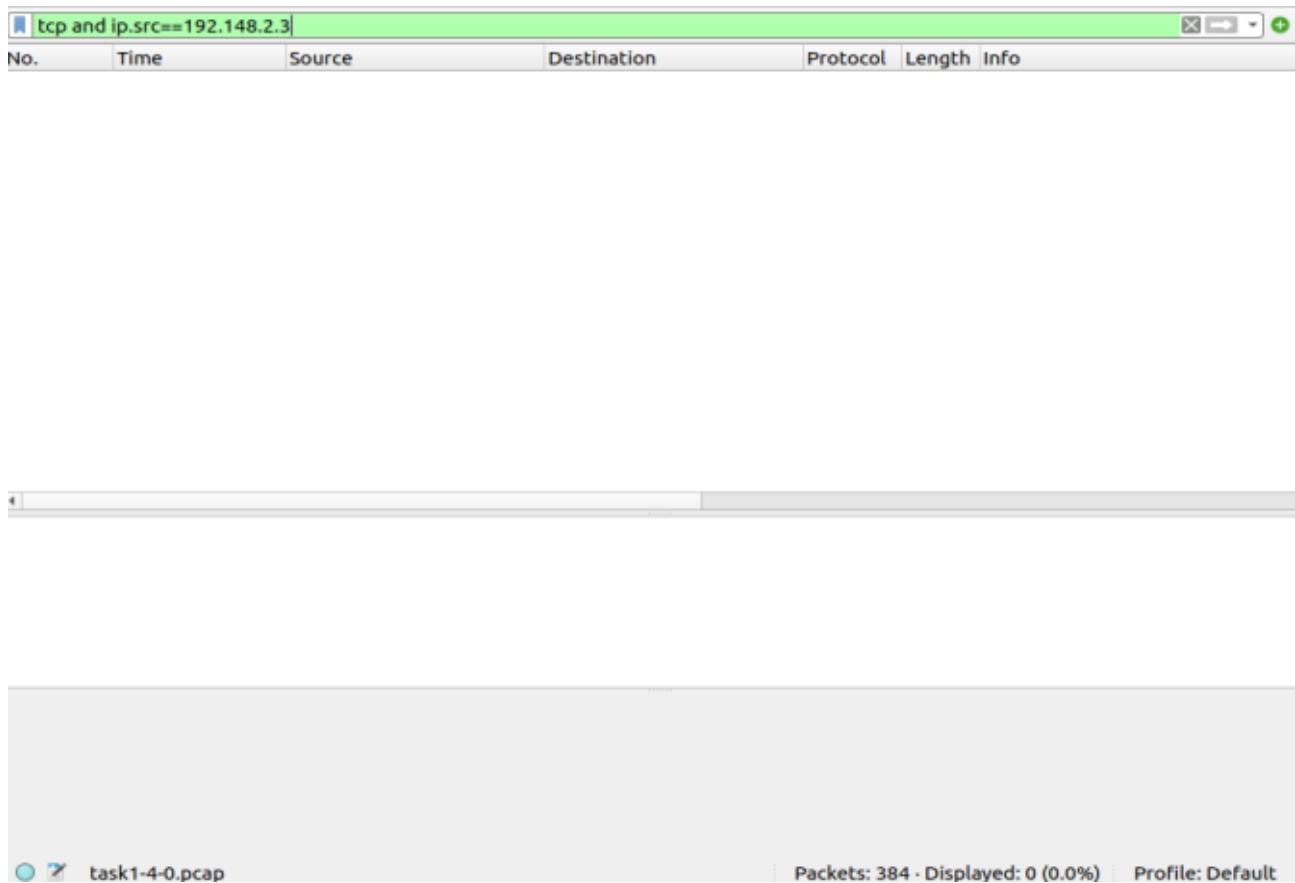


Fig. 2.18 - File task1-4-0.pcap, notare Packets totali e Displayed

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.148.2.3	192.148.1.1	TCP	58	49153 → 2600 [SYN] Seq: 0
4	0.017337	192.148.2.3	192.148.1.1	TCP	54	49153 → 2600 [ACK] Seq: 1
5	0.065562	192.148.2.3	192.148.1.1	ZEBRA	590	Zebra Request
6	0.065766	192.148.2.3	192.148.1.1	ZEBRA	590	Zebra Request
7	0.065973	192.148.2.3	192.148.1.1	ZEBRA	590	Zebra Request
8	0.066188	192.148.2.3	192.148.1.1	ZEBRA	590	Zebra Request
9	0.066396	192.148.2.3	192.148.1.1	ZEBRA	590	Zebra Request
10	0.066511	192.148.2.3	192.148.1.1	ZEBRA	374	Zebra Request
11	0.113562	192.148.2.3	192.148.1.1	ZEBRA	590	Zebra Request
12	0.113766	192.148.2.3	192.148.1.1	ZEBRA	590	Zebra Request
13	0.113971	192.148.2.3	192.148.1.1	ZEBRA	590	Zebra Request
14	0.114177	192.148.2.3	192.148.1.1	ZEBRA	590	Zebra Request
15	0.114387	192.148.2.3	192.148.1.1	ZEBRA	590	Zebra Request
16	0.114501	192.148.2.3	192.148.1.1	ZEBRA	374	Zebra Request
17	0.161562	192.148.2.3	192.148.1.1	ZEBRA	590	Zebra Request
18	0.161767	192.148.2.3	192.148.1.1	ZEBRA	590	Zebra Request
19	0.161972	192.148.2.3	192.148.1.1	ZEBRA	590	Zebra Request
20	0.162181	192.148.2.3	192.148.1.1	ZEBRA	590	Zebra Request
21	0.162392	192.148.2.3	192.148.1.1	ZEBRA	590	Zebra Request

> Frame 1: 58 bytes on wire (464 bits), 58 bytes captured (464 bits)
 > Point-to-Point Protocol
 > Internet Protocol Version 4, Src: 192.148.2.3, Dst: 192.148.1.1
 > Transmission Control Protocol, Src Port: 49153, Dst Port: 2600, Seq: 0, Len: 0

0000	00	21	45	00	00	38	00	00	00	00	3e	06	00	00	c0	94	..!E..8.. ..>....
0010	02	03	c0	94	01	01	c0	01	0a	28	00	00	00	00	00	00(.....
0020	00	00	90	02	ff	ff	00	00	00	00	08	0a	00	00	0b	b8
0030	00	00	00	00	03	03	02	04	02	00

Fig. 2.19 - File task1-5-0.pcap, notare Packets totali e Displayed

I pacchetti raggiungono poi il nodo n3, passano a n2 e attraversano il canale LAN per arrivare a destinazione nel nodo n0.

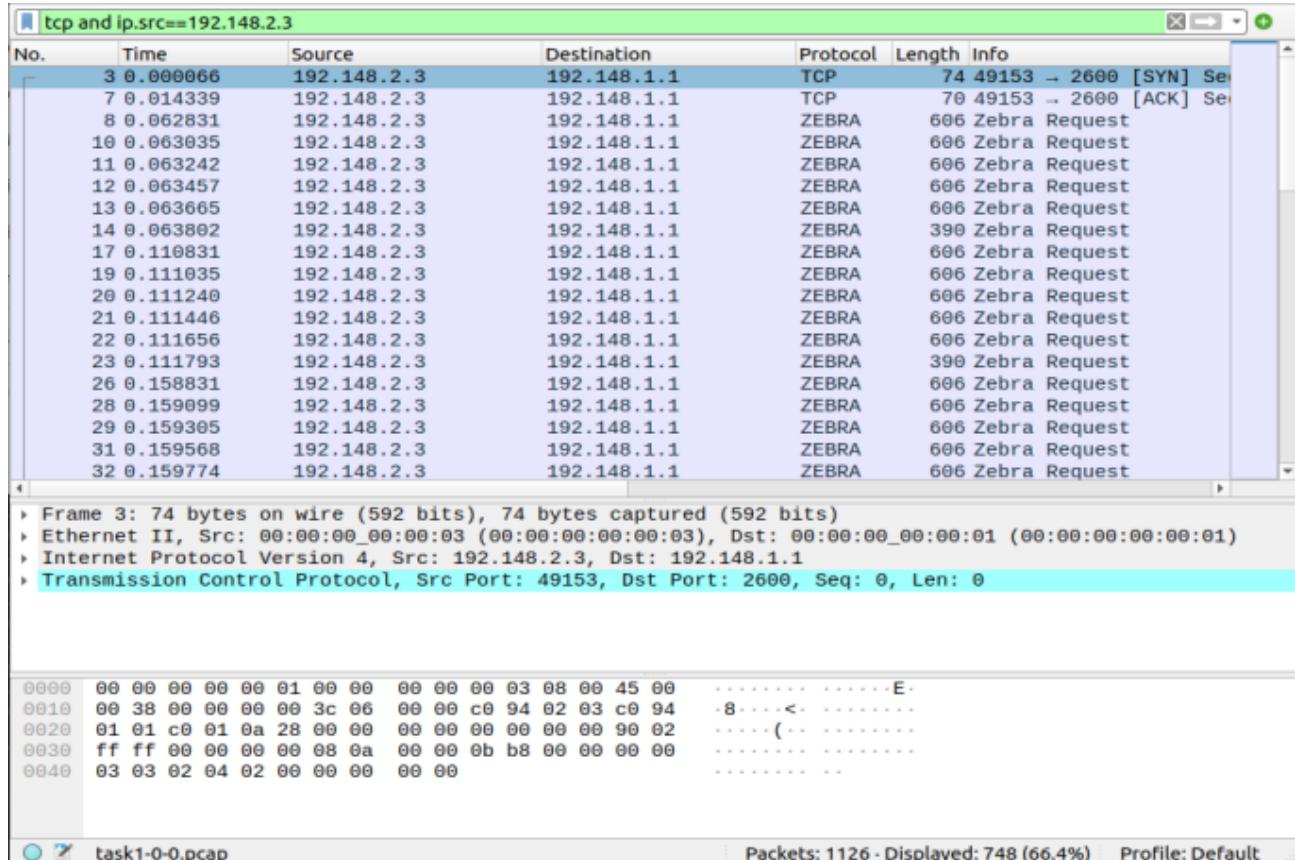


Fig. 2.20 - File task1-0-0.pcap, notare Packets totali e Displayed

UDP n8 ↔ n7

Vengono inviati 208 pacchetti dal nodo n8 al nodo n7, evidenziati applicando il filtro `udp and ip.src==192.148.2.2`, per selezionare solo i pacchetti basati sul protocollo UDP inviati al nodo n7 con indirizzo IP 192.168.2.2.

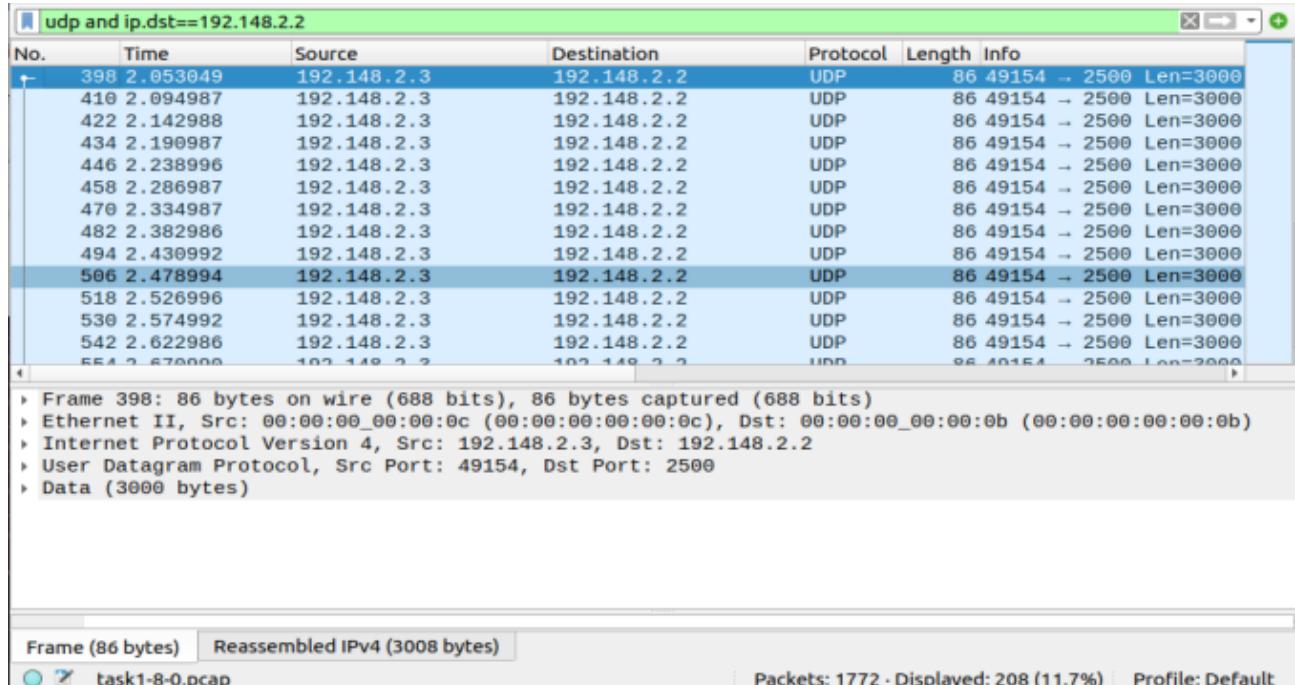


Fig. 2.21 - File task1-8-0.pcap, notare Packets totali e Displayed

I pacchetti attraversano il canale LAN per raggiungere il nodo n7. In questo caso i pacchetti attraversano solo la seconda LAN senza attraversare tutti gli altri collegamenti che formano la topologia.

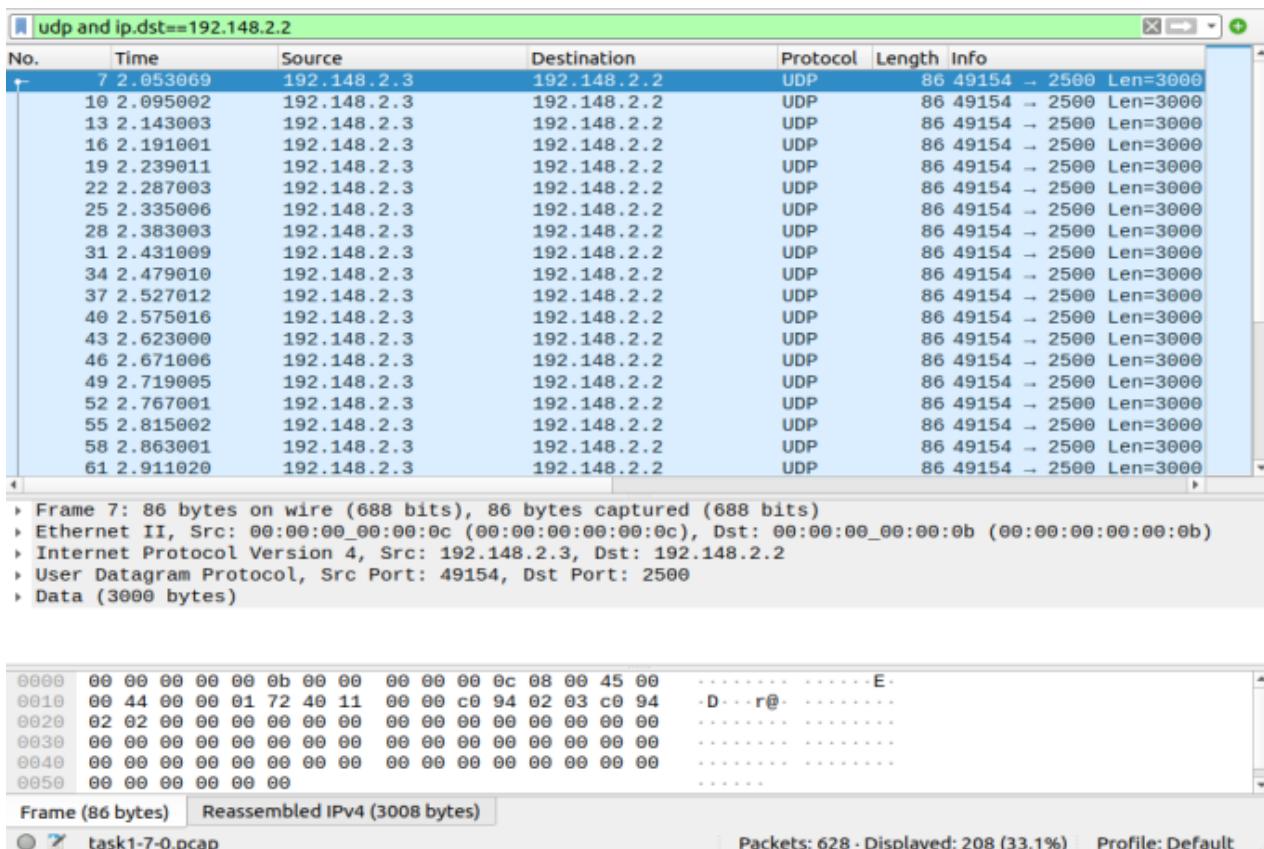


Fig. 2.22 - File task1-7-0.pcap, notare Packets totali e Displayed

A3: Calcolo e grafico di round trip time(RTT) e commento

Configurazione 0
Connessione nodi n8↔n0

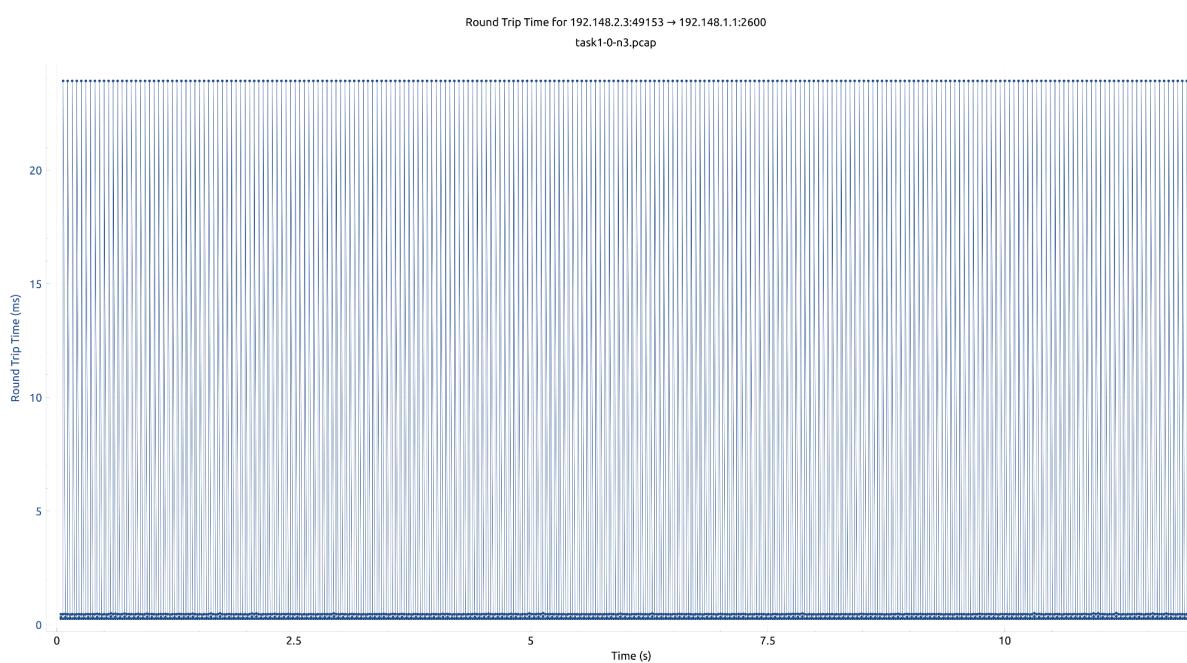


Fig. 3.1 - Grafico RTT

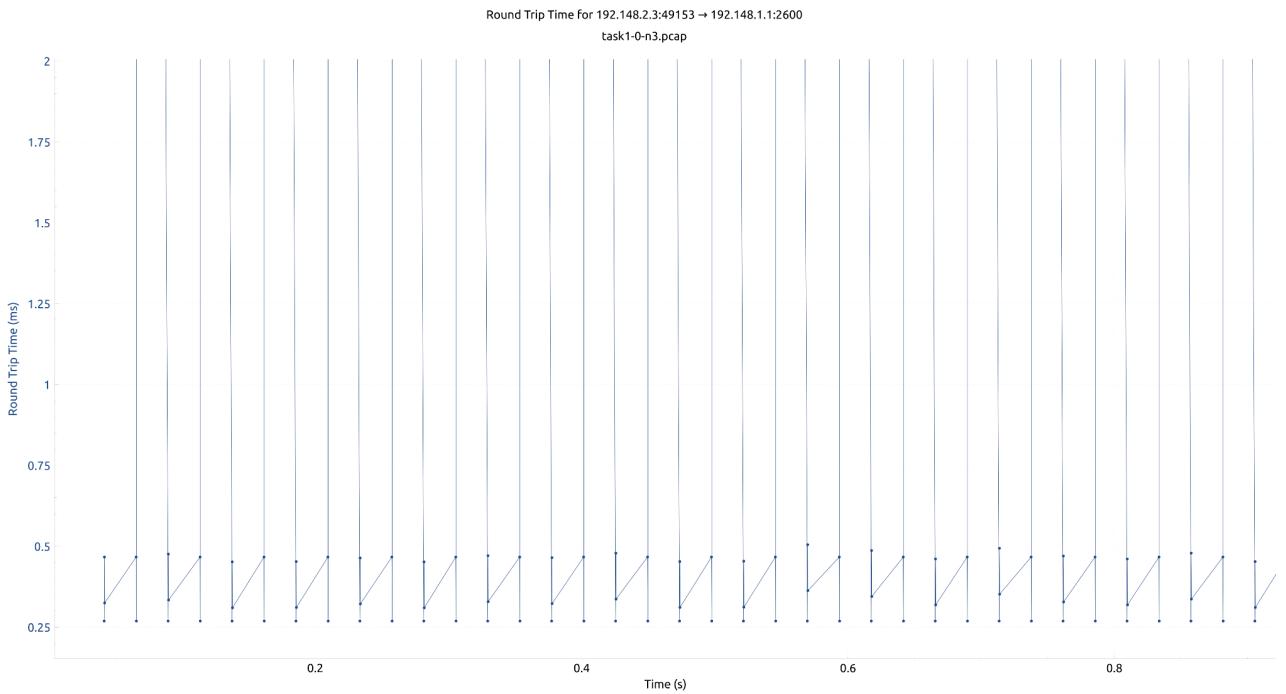


Fig. 3.2 - Grafico RTT in dettaglio sui valori minimi

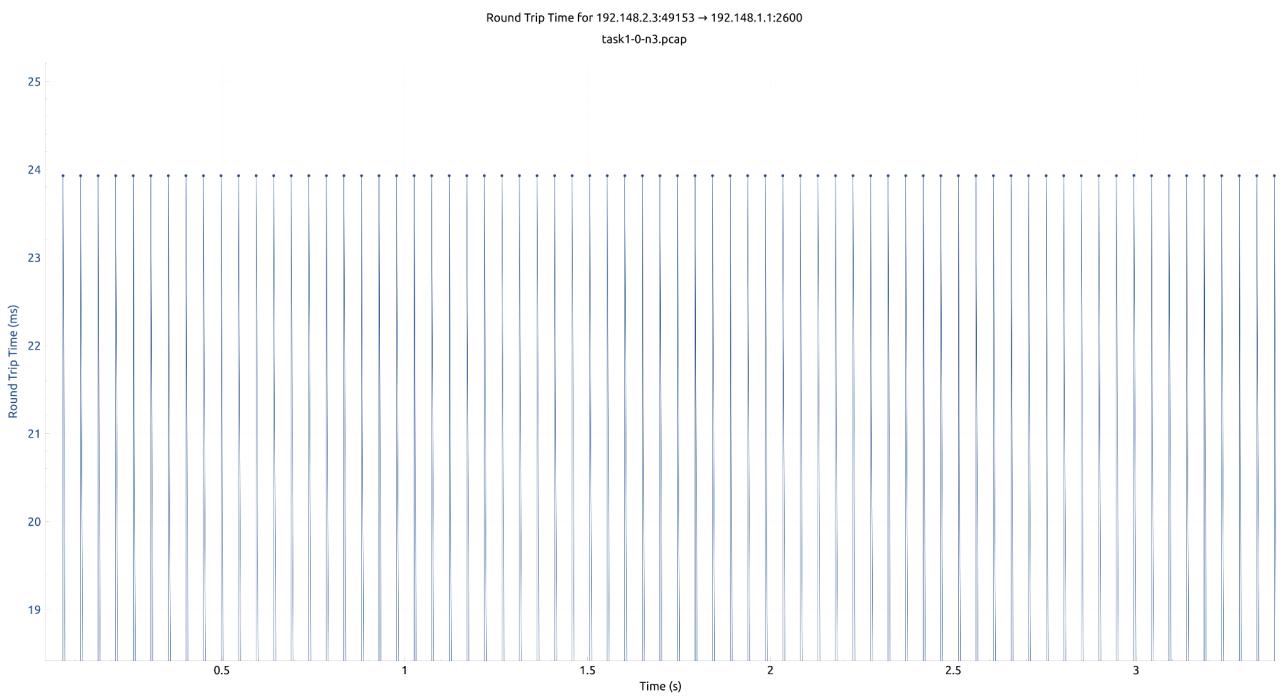


Fig 3.3 - Grafico RTT in dettaglio sui valori massimi

Il RTT è periodico con periodo $T = 0.023 \text{ sec}$ circa, un valore minimo $RTT_{min} = 0.32 \text{ ms}$ circa e uno massimo $RTT_{max} = 25 \text{ ms}$ circa.

Notiamo che quando la trasmissione raggiunge un $RTT = 0.5 \text{ ms}$ circa incontra della congestione debole e la dimensione della finestra viene dimezzata, mentre tende ad arrivare al suo RTT_{max} fino a quando non incontra un livello di congestione forte tale che la finestra di congestione venga totalmente chiusa facendo arrivare il RTT al suo minimo.

Configurazione 1

Connessione nodi n1↔n7

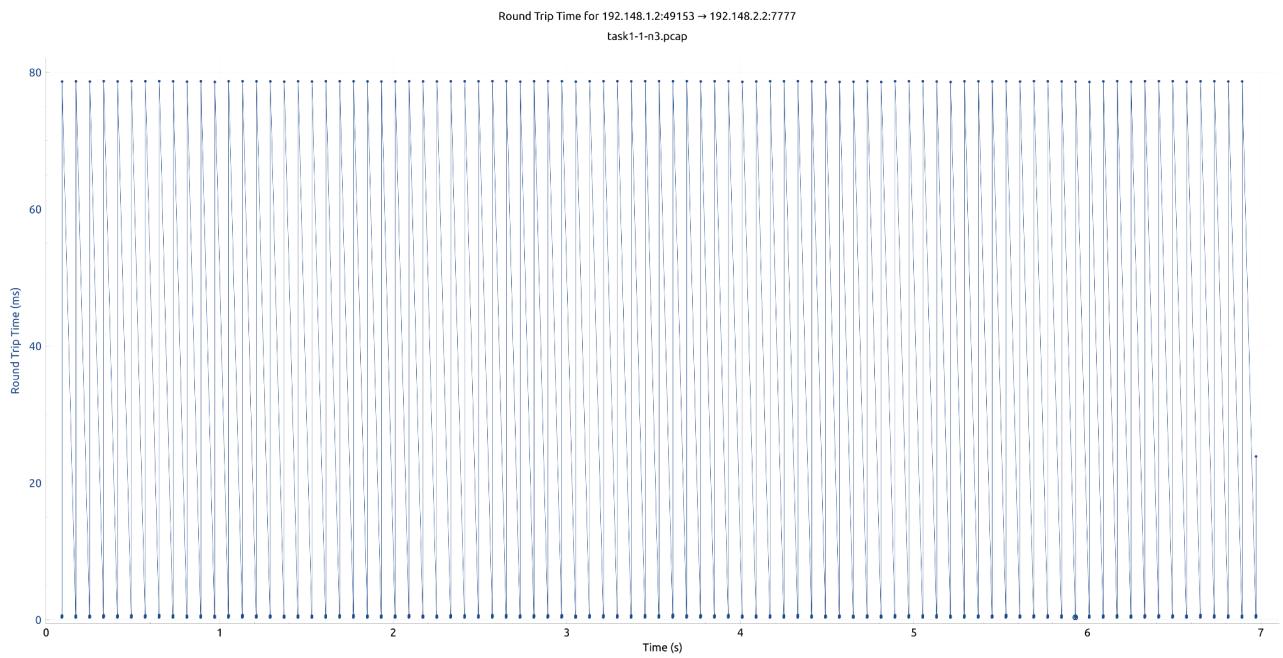


Fig. 3.4 - Grafico RTT

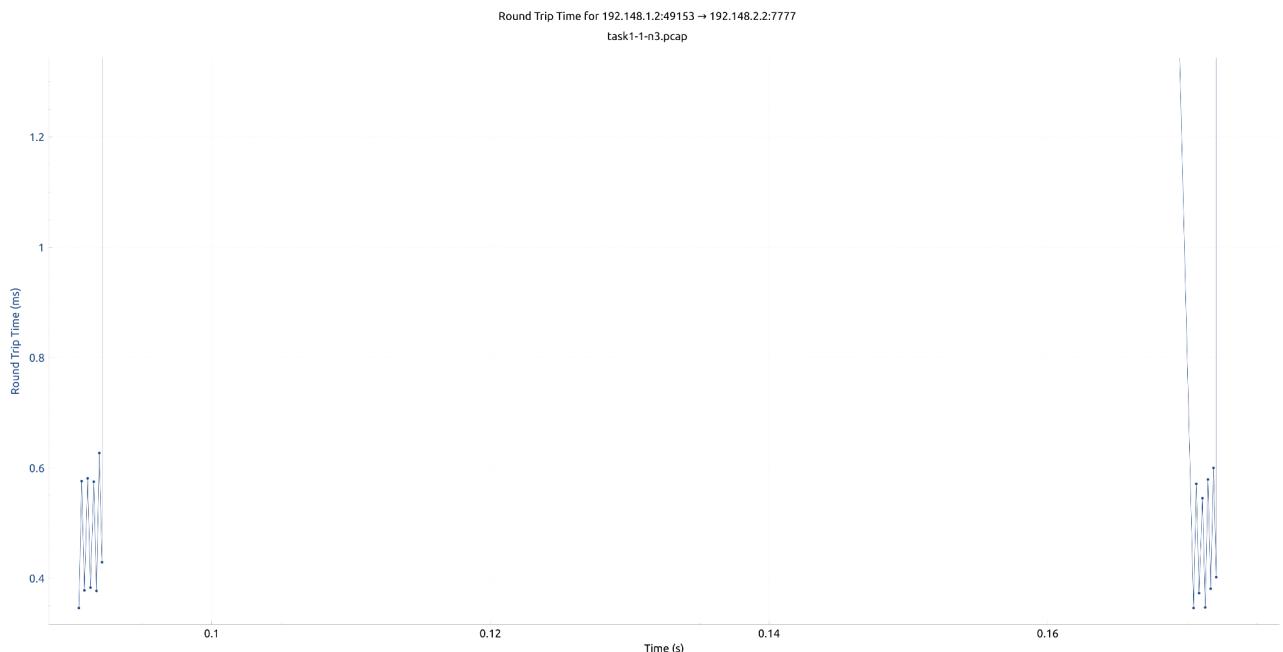


Fig. 3.5 - Grafico RTT in dettaglio sui valori minimi

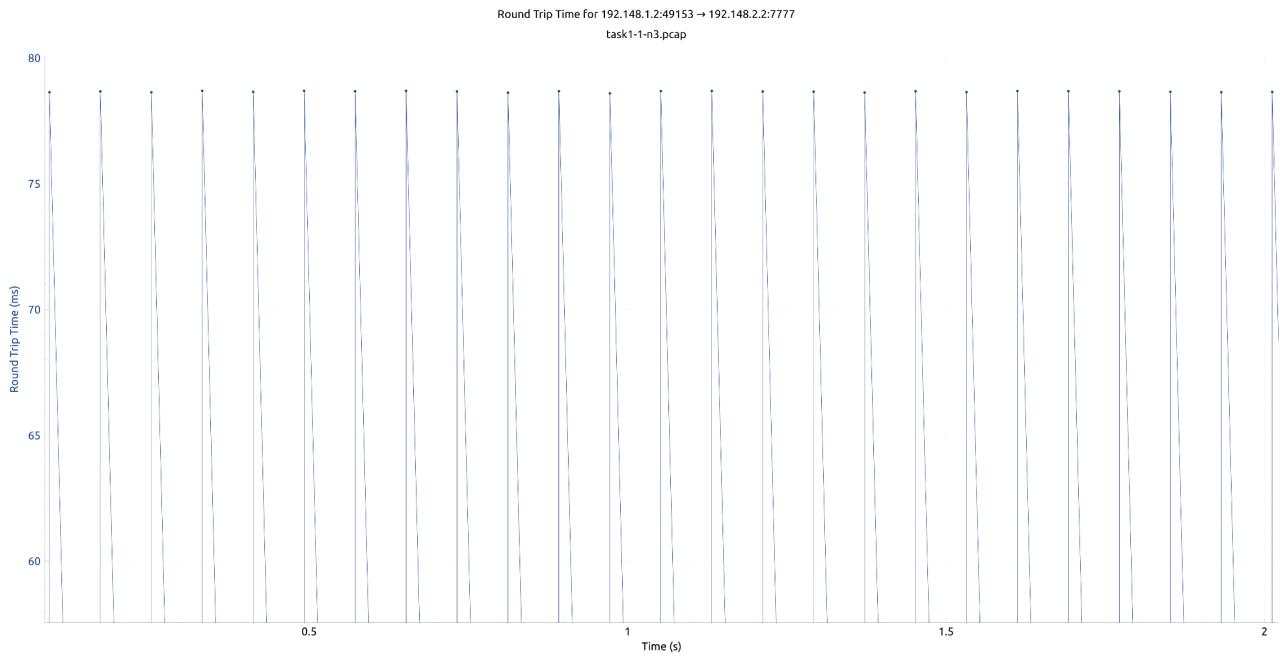


Fig. 3.6 - Grafico RTT in dettaglio sui valori massimi

Il RTT è periodico con periodo $T = 0.08 \text{ sec}$ circa, un valore minimo $RTT_{min} = 5.5 \text{ ms}$ circa e uno massimo $RTT_{max} = 86 \text{ ms}$ circa.

Notiamo che quando la trasmissione raggiunge un $RTT = 5.65 \text{ ms}$ circa incontra della congestione debole e la dimensione della finestra viene dimezzata, mentre tende ad arrivare al suo RTT_{max} fino a quando non incontra un livello di congestione forte tale che la finestra di congestione venga totalmente chiusa facendo arrivare il RTT al suo minimo.

Connessione nodi n8↔n0

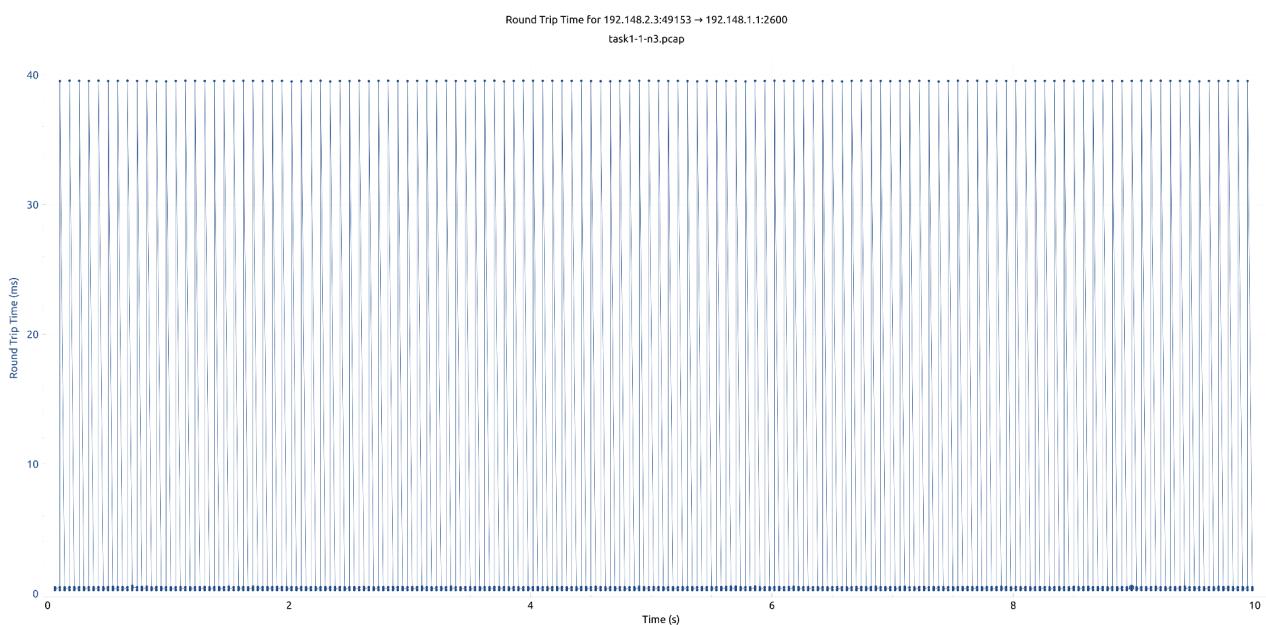


Fig. 3.7 - Grafico RTT

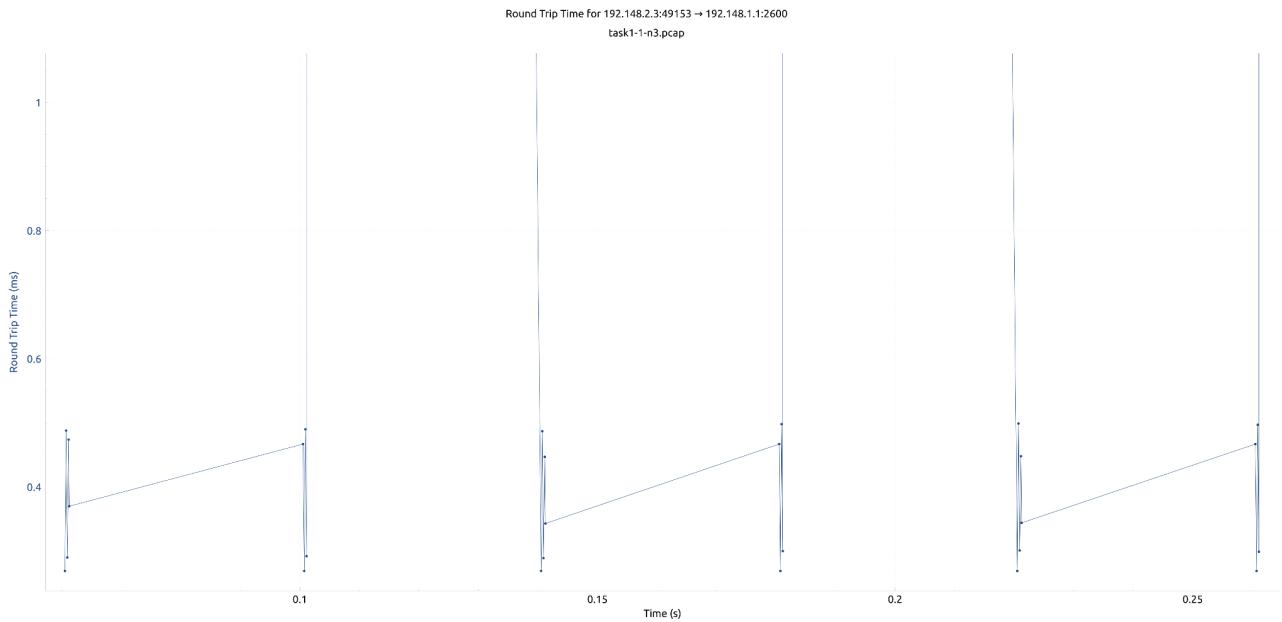


Fig. 3.8 - Grafico RTT in dettaglio sui valori minimi

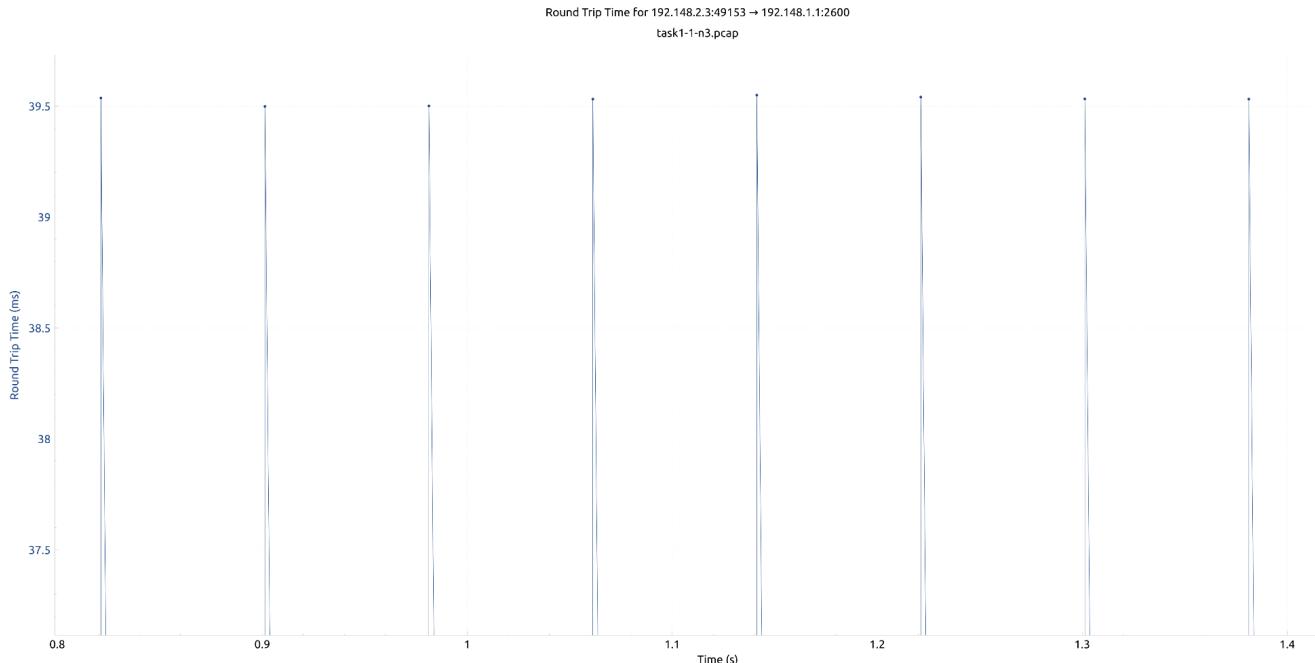


Fig. 3.9 - Grafico RTT con dettaglio sui valori massimi

Il RTT è periodico con periodo $T = 0.04 \text{ sec}$ circa, un valore minimo $RTT_{min} = 0.3 \text{ ms}$ circa e uno massimo $RTT_{max} = 41 \text{ ms}$ circa.

Notiamo che quando la trasmissione raggiunge un $RTT = 0.6 \text{ ms}$ circa incontra della congestione debole e la dimensione della finestra viene dimezzata, mentre tende ad arrivare al suo RTT_{max} fino a quando non incontra un livello di congestione forte tale che la finestra di congestione venga totalmente chiusa facendo arrivare il RTT al suo minimo.

Configurazione 2

Connessione nodi n8↔n0

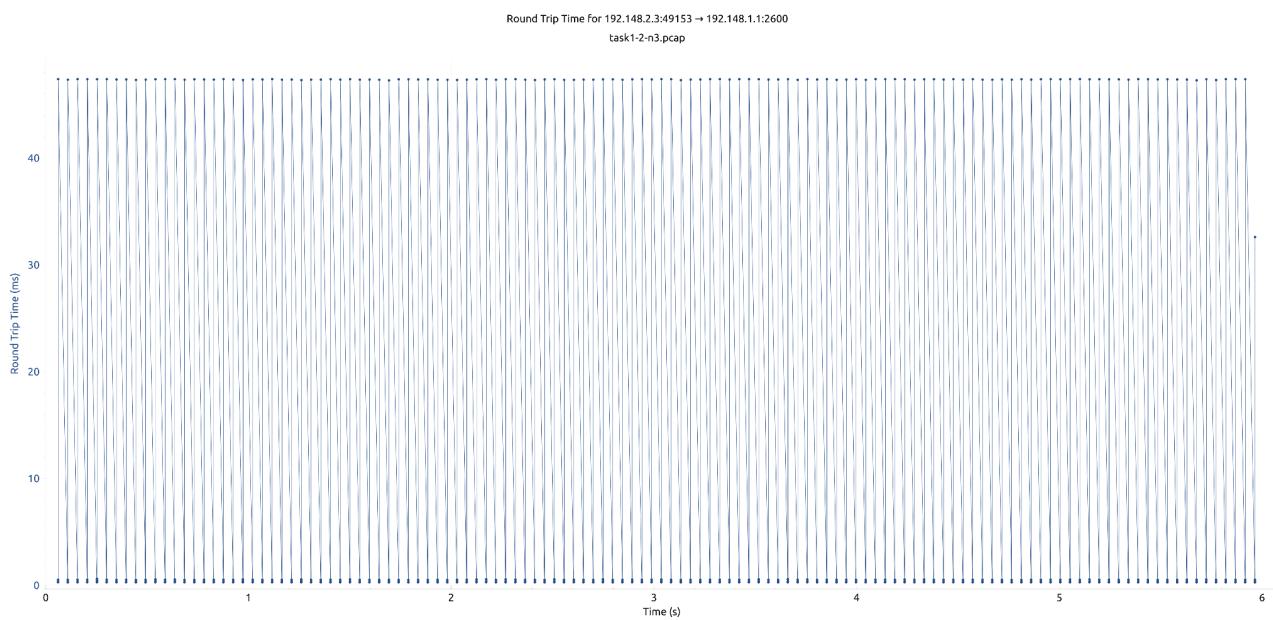


Fig. 3.10 - Grafico RTT

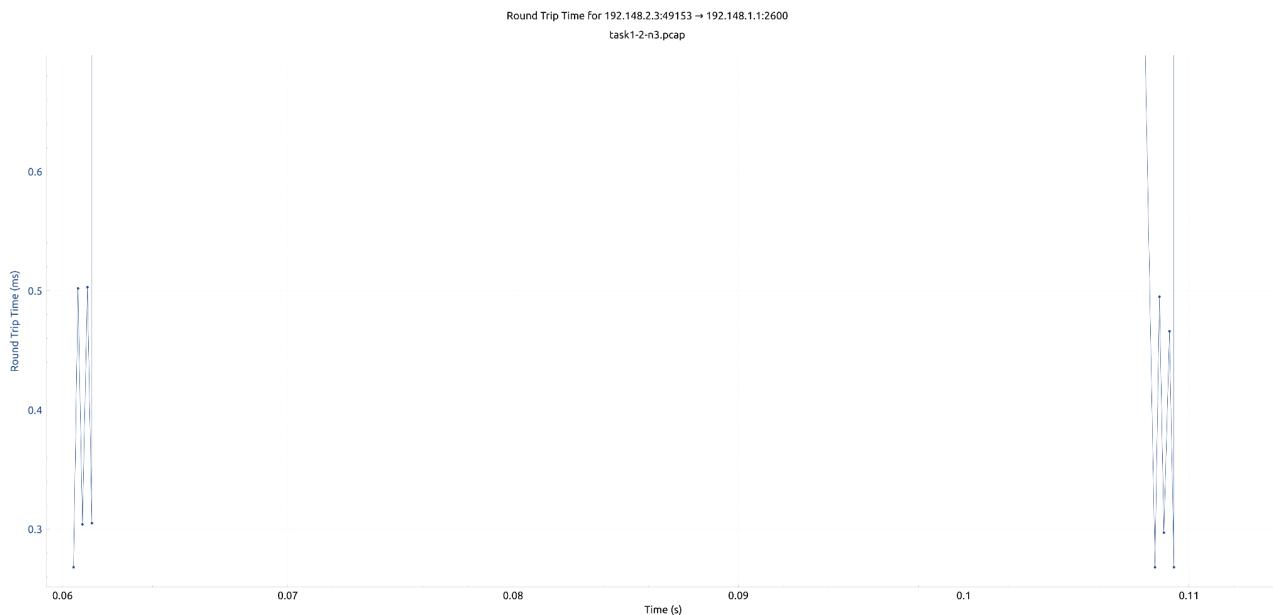


Fig. 3.11 - Grafico RTT con dettaglio sui valori minimi

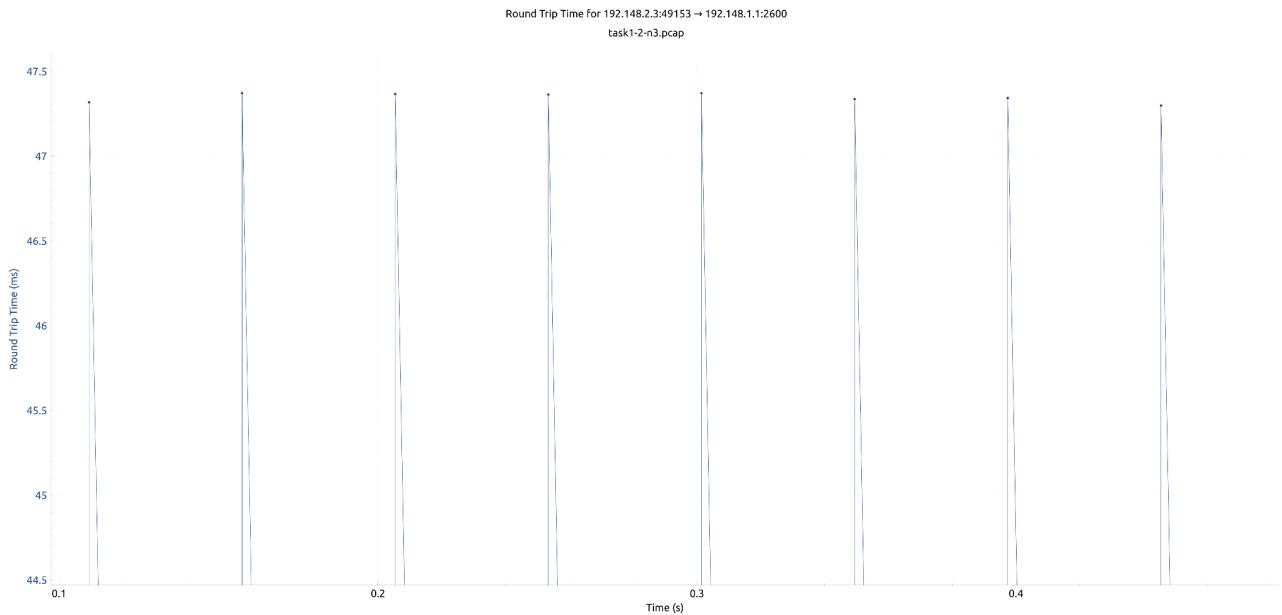


Fig. 3.12 - Grafico RTT con dettaglio sui valori massimi

Il RTT è periodico con periodo $T = 0.048 \text{ sec}$ circa, un valore minimo $RTT_{min} = 5.5 \text{ ms}$ circa e uno massimo $RTT_{max} = 54 \text{ ms}$ circa.

Notiamo che quando la trasmissione raggiunge un $RTT = 5.8 \text{ ms}$ circa incontra della congestione debole e la dimensione della finestra viene dimezzata, mentre tende ad arrivare al suo RTT_{max} fino a quando non incontra un livello di congestione forte tale che la finestra di congestione venga totalmente chiusa facendo arrivare il RTT al suo minimo.

A4: Vi sono dei bottleneck nella rete? Se sì, individuare gli eventuali link e discutere eventuali contromisure e soluzioni.

Il bottleneck è il link o punto della trasmissione in cui il throughput è minore. In questi tratti di collegamento la trasmissione sarà più lenta e quindi il Data Rate sarà il minore possibile, rispetto a quello degli altri link della comunicazione.

Nella topologia presa in esempio i bottleneck sono rappresentati dai collegamenti LAN che hanno un Data Rate di 25 Mbps, nettamente minore rispetto a quello dei collegamenti Point-to-Point. Un throughput basso è dovuto al fatto che i bit trasmessi al secondo sono pochi a causa della bassa capacità di trasmissione del mezzo.

Una possibile contromisura sarebbe la sostituzione del mezzo trasmittivo con uno avente Data Rate maggiore, in modo da avere una più alta trasmissione di bit al secondo e un aumento del throughput.

C01: Calcolare il throughput istantaneo del flusso TCP

Soluzione 1

Analizziamo il throughput istantaneo del flusso TCP in due differenti istanti di tempo, in modo da osservare come varia il trasferimento dei bit.

Consideriamo il pacchetto No. 1 nell'analisi Wireshark inviato a 0.00006 sec. Utilizzando le informazioni fornite dal grafico Statistics/ I/O Graphs notiamo che in quell'istante di tempo si ha un throughput istantaneo di 584594 bit/sec.

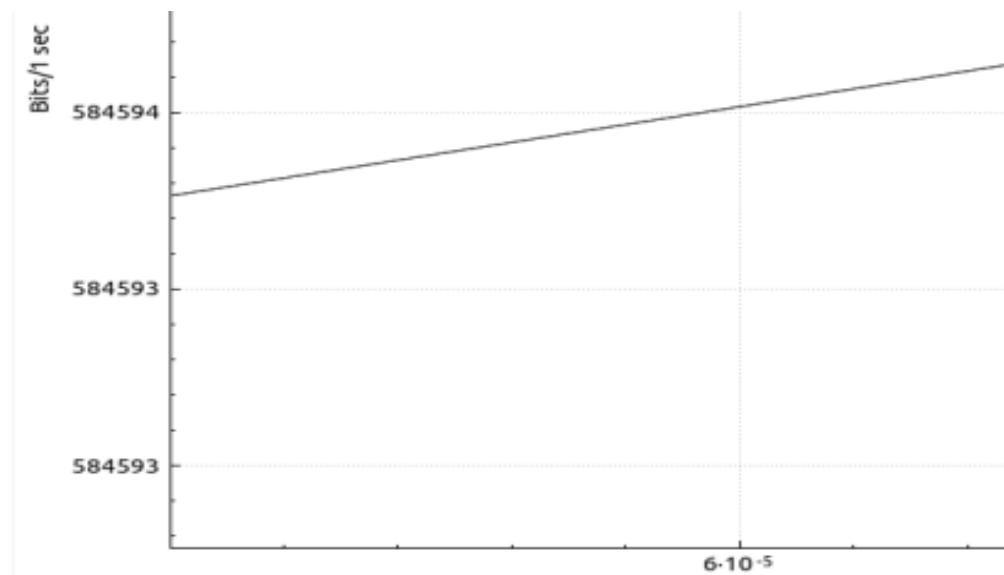


Fig. 01.1 - Grafico throughput pacchetto No. 1

Scegliamo il pacchetto con No. 949 nell'analisi Wireshark inviato a 5.01465 sec. Notiamo che in quell'istante di tempo si ha un throughput istantaneo di 608536 bit/sec.

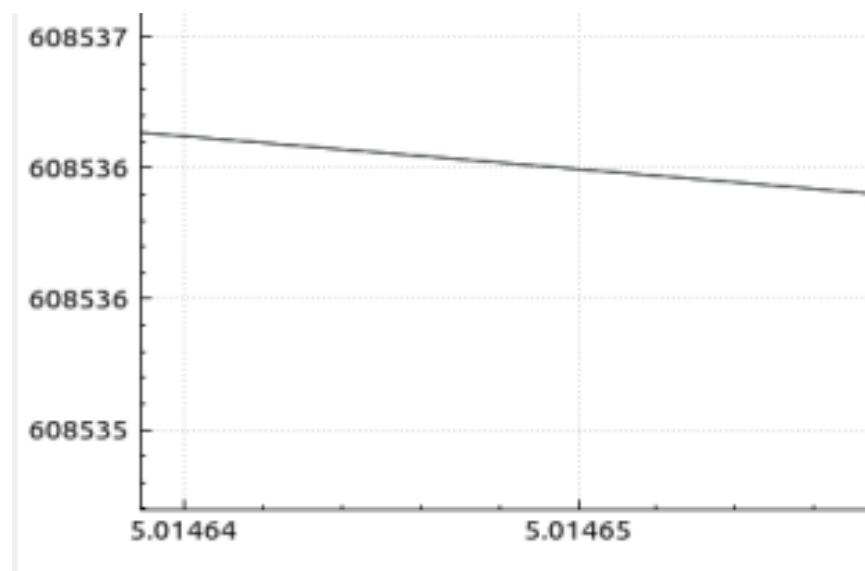


Fig. 01.2 - Grafico throughput pacchetto No. 949

Possiamo quindi notare che a distanza di 5,01459 sec e 948 pacchetti c'è un aumento del throughput di 23942 bit/sec.

Soluzione 2

Il throughput del flusso TCP è dato da

$$TH = \frac{\Delta t_{RTT}}{RTT} = \frac{W}{W+2\cdot d \cdot C} = \frac{W}{W+2\alpha} = \min[1, \frac{W}{2\alpha}] =$$

$$\begin{cases} 1 & \text{se } W \geq 2\alpha \\ \frac{W}{2\alpha} & \text{se } W < 2\alpha \end{cases}$$

Dove

$$\Delta t_{RTT} = \text{tempo di trasmissione utile in un RTT}$$

$$W = \text{dimensione finestra di trasmissione}$$

$$d = \text{ritardo EndToEnd della trasmissione}$$

$$C = \text{bitRate del bottleneck}$$

In questa configurazione, da Wireshark otteniamo $W = 8192 \text{ byte}$. Questo valore è stato ricavato prendendo il flag $WIN = 131072 \text{ byte}$, dimensione ricorrente della finestra della maggior parte dei pacchetti, e dividendolo per il fattore 4 di Window Scale, ricavato dai dettagli dei pacchetti, ovvero 2^4 .

$$W = \frac{WIN}{2^4} = \frac{131071}{16} \text{ byte} = 8292 \text{ byte} = 66336 \text{ bit}$$

Sommando i ritardi di propagazione dei collegamenti attraversati, ricavati dal testo, otteniamo

$$d = 10 + 5 + 5 + 15 + 10 = 45 \mu\text{sec} = 0.000045 \text{ sec}$$

Dal testo $C = 25 \text{ Mbps}$. Otteniamo dunque

$$TH = \frac{66336 \text{ bit}}{66336 \text{ bit} + 2 \cdot 0.000045 \text{ sec} \cdot 25000000 \text{ bit/sec}} = 0.9672$$

Per quanto affermato precedentemente, $TH = 1$.

C02: Calcolare il throughput medio del flusso TCP a tempo t=4.0s.

Il throughput medio del flusso TCP calcolato in un tempo t rappresenta il numero di bit inviati fino all'istante t dall'inizio della connessione.

$$TH(t) = \frac{T}{t-t_0}$$

$$T = \text{numero dei pacchetti trasmessi nell'istante di tempo } t$$

$$t_0 = \text{istante iniziale di trasmissione}$$

Abilitando le informazioni di Log su PacketSink e su OnOffClient, attraverso la simulazione, osserviamo che all'istante $t = 3.98446 \text{ sec} \approx 4 \text{ sec}$ vengono trasmessi da OnOffClient $T = 60000 \text{ byte} = 480000 \text{ bit}$. Dalla traccia abbiamo $t_0 = 3 \text{ sec}$, istante in cui il TCP OnOffClient su n8 inizia ad inviare i dati.

$$TH(4) = \frac{480000 \text{ bit}}{4-3 \text{ sec}} = 480000 \frac{\text{bit}}{\text{sec}}$$

Questo risultato è anche osservabile nel grafico del throughput medio presente nell'analisi Wireshark attraverso la sezione Statistics/ TCP Stream Graphs/ Throughput.

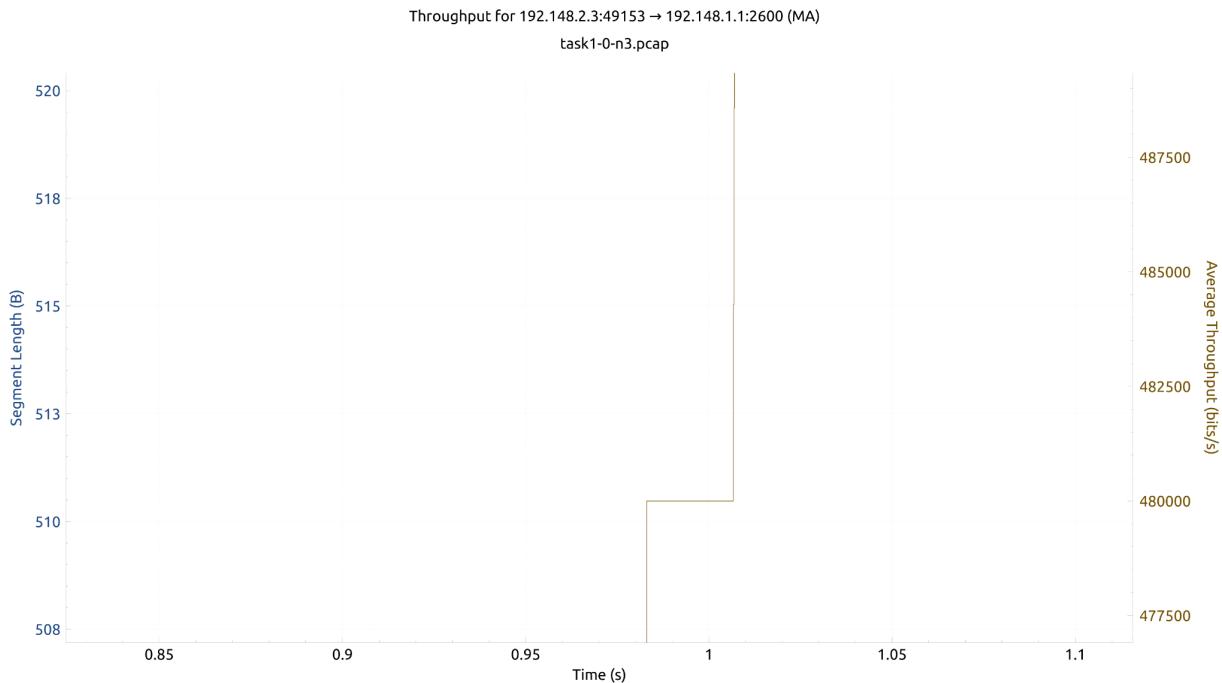


Fig. 02.1 - Grafico throughput

C03: Calcolare il throughput medio del flusso TCP a tempo t=7.0s. Commentare eventuali cambiamenti rispetto a C02.

Abilitando le informazioni di Log su PacketSink e su OnOffClient, attraverso la simulazione osserviamo che all'istante $t = 7 \text{ sec}$ vengono trasmessi da OnOffClient $T = 249000 \text{ byte} = 1992000 \text{ bit}$. Dalla traccia sappiamo che $t_0 = 3 \text{ sec}$, istante in cui il TCP OnOffClient su n8 inizia ad inviare i dati.

$$TH(7) = \frac{1992000 \text{ bit}}{7-3 \text{ sec}} = 498000 \frac{\text{bit}}{\text{sec}}$$

Questo risultato si può anche osservare nel grafico del throughput medio presente nell'analisi Wireshark attraverso la sezione Statistics/TCP Stream Graphs/Throughput.

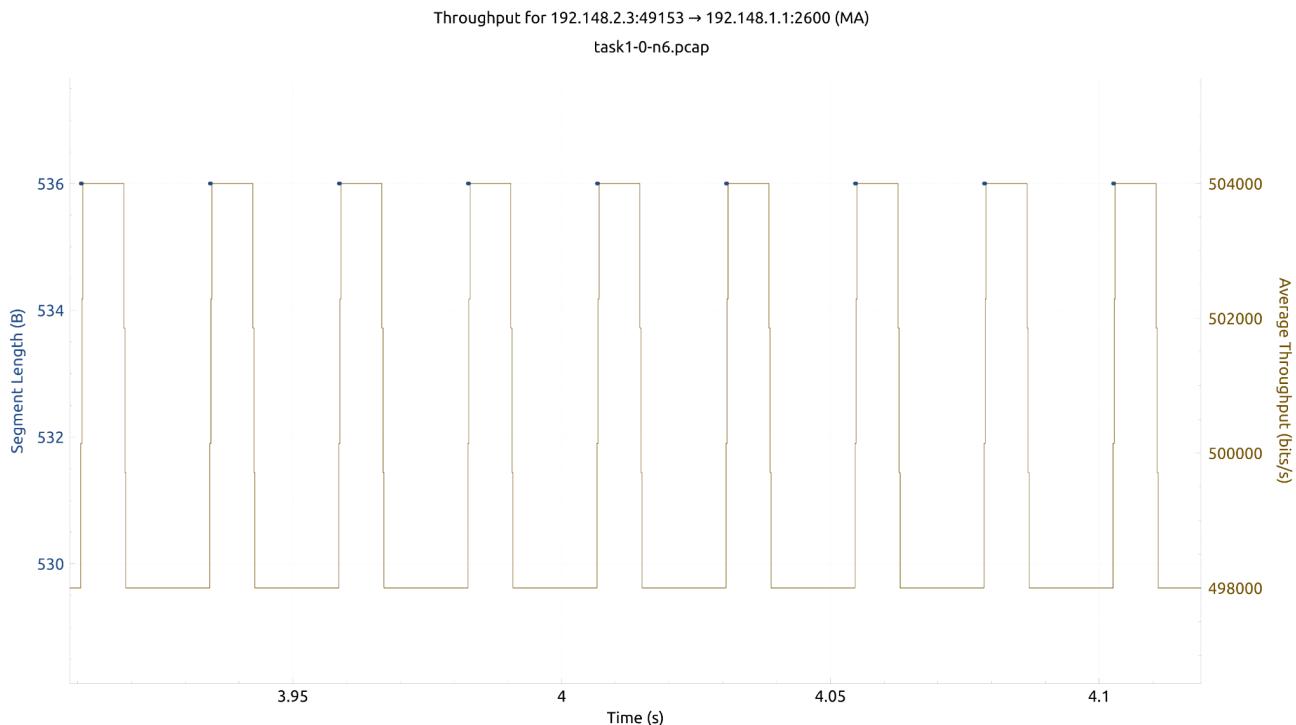


Fig. 03.1 - Grafico throughput - Assumiamo MA = 2 per rendere più dettagliato il grafico.

A differenza di C02, il throughput è aumentato, infatti a distanza di 5.01459 sec c'è una variazione del throughput di 23942 bit/sec, proprio perché è aumentata la quantità di bit trasmessa a tempo 7 sec. Questo implica che da $t_1 = 4 \text{ sec}$ a $t_2 = 7 \text{ sec}$ c'è stato un aumento dei bit al secondo trasmessi dal client, probabilmente per un aumento della velocità di trasmissione.

C04: Calcolare il ritardo di trasferimento complessivo di tutti i pacchetti inviati.

Assumiamo che il ritardo di accodamento e di elaborazione nei nodi attraversati dai pacchetti inviati dal nodo n8 al nodo n0 siano trascurabili.

Il ritardo di trasferimento complessivo può essere ottenuto sommando i ritardi di propagazione, sui collegamenti attraversati, ai ritardi di trasmissione, che cambiano a seconda del collegamento attraversato.

Ricavando i ritardi di propagazione dal testo, si ha

$$d_{prop} = 10 + 5 + 5 + 15 + 10 = 45 \mu\text{s} = 0,000045 \text{ s}$$

Il ritardo di trasferimento su un singolo collegamento si ricava come $d_{tras} = \frac{L}{R}$, dove L sono i bit trasferiti sul collegamento e R è il Data Rate. Avendo 1498 pacchetti inviati, ricavati dall'analisi Wireshark, $L = 1498 \cdot 1500 \text{ byte} = 17976000 \text{ bit}$. Perciò il ritardo di trasferimento totale è dato da

$$d_{tras} = \frac{17976000}{25000000} + 2 \frac{17976000}{80000000} + \frac{17976000}{100000000} + \frac{17976000}{25000000} \text{ sec} = 1.25832 \text{ sec}$$

Quindi il ritardo totale di trasmissione

$$d_{tot} = d_{prop} + d_{tras} = 0.000045 + 1.25832 \text{ sec} = 1.258365 \text{ sec}$$

C11: Calcolare il throughput medio dei flussi TCP.

Nella configurazione 1 ci sono due flussi TCP, il primo manda pacchetti dal nodo n8 a n0, il secondo dal nodo n1 a n7.

Nella sezione Statistics/Conversations di Wireshark osserviamo che il flusso TCP da n8 con indirizzo IP 192.148.2.3 a n0 con IP 192.148.1.1 vengono inviati 689000 bytes = 5512000 bit. La durata del trasferimento è di 10 sec, allora il throughput medio è

$$TH_1 = \frac{5512000 \text{ bit}}{10 \text{ sec}} = 551200 \frac{\text{bit}}{\text{sec}}$$

Il flusso TCP da n1 con indirizzo IP 192.148.1.2 a n7 con IP 192.148.2.2 invia 482000 byte=3856000 bit. La durata del trasferimento è di 7 sec, allora il throughput medio è

$$TH_2 = \frac{3856000 \text{ bit}}{7 \text{ sec}} = 550857 \frac{\text{bit}}{\text{sec}}$$

Entrambi i valori equivalgono a quelli ricavati da Wireshark.

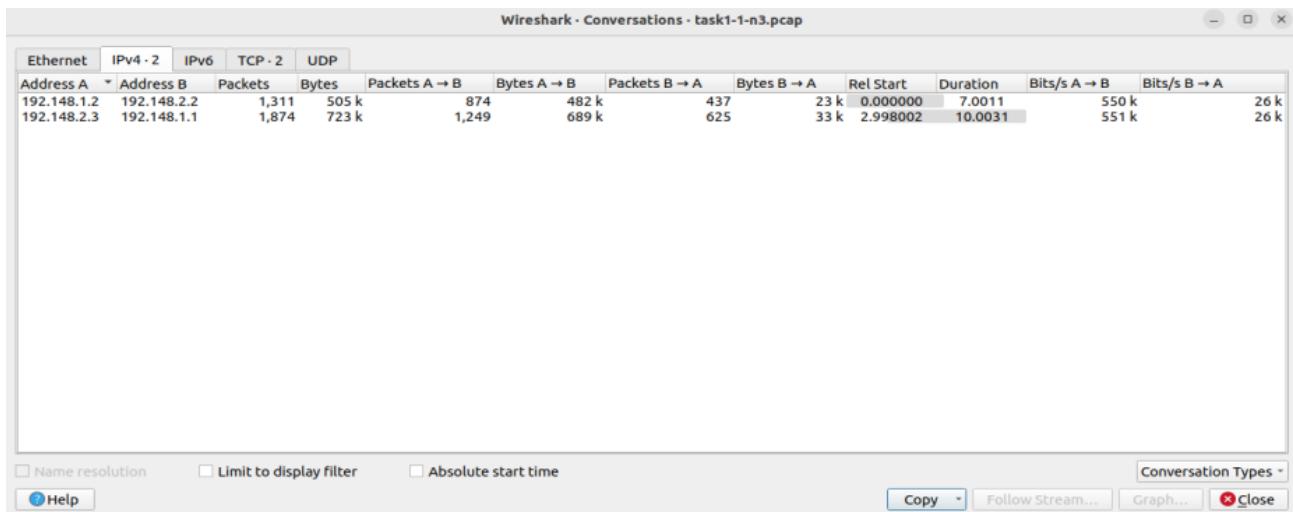


Fig. 11.1 - Tabella Conversations

C12: Calcolare il throughput medio del flusso TCP n8 verso n0 a tempo t=6s.

Abilitando le informazioni di Log su PacketSink e su OnOffClient, attraverso la simulazione vediamo che all'istante $t = 6 \text{ sec}$ vengono trasmessi da OnOffClient $T = 250000 \text{ byte} = 2000000 \text{ bit}$. $t_0 = 2 \text{ sec}$ è l'istante in cui la trasmissione viene aperta.

$$TH(6) = \frac{2000000 \text{ bit}}{6-2 \text{ sec}} = 500000 \frac{\text{bit}}{\text{sec}}$$

Questo risultato si può anche osservare nel grafico del throughput medio presente nell'analisi Wireshark attraverso la sezione Statistics/TCP Stream Graphs/Throughput.

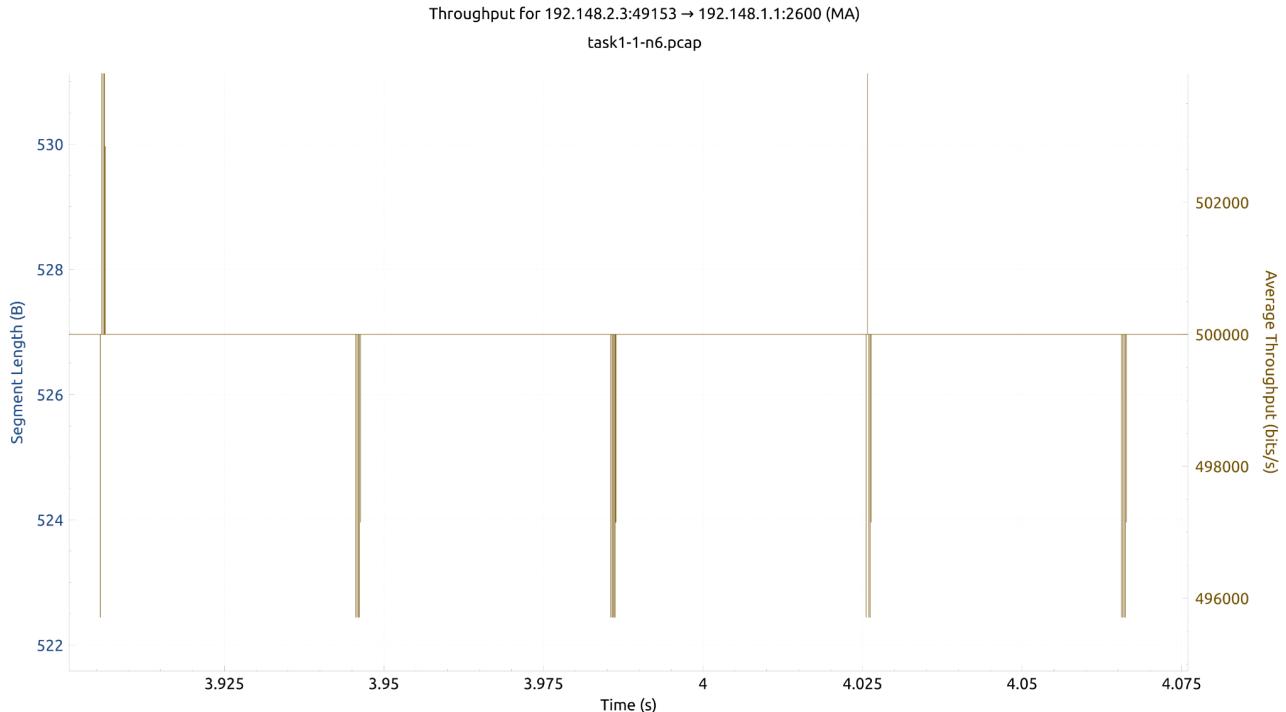


Fig. 12.1 - Grafico throughput

C13: Calcolare il throughput medio del flusso TCP n8 verso n0 a tempo t=8s. Commentare eventuali cambiamenti rispetto a C12.

Abilitando le informazioni di Log su PacketSink e su OnOffClient, attraverso la simulazione vediamo che all'istante $t = 8 \text{ sec}$ vengono trasmessi da OnOffClient $T = 375000 \text{ byte} = 3000000 \text{ bit}$. $t_0 = 2 \text{ sec}$ è l'istante in cui la trasmissione viene aperta.

$$TH(8) = \frac{3000000 \text{ bit}}{8 - 2 \text{ sec}} = 500000 \frac{\text{bit}}{\text{sec}}$$

Questo risultato si può anche osservare nel grafico del throughput medio presente nell'analisi Wireshark attraverso la sezione Statistics/TCP Stream Graphs/Throughput.

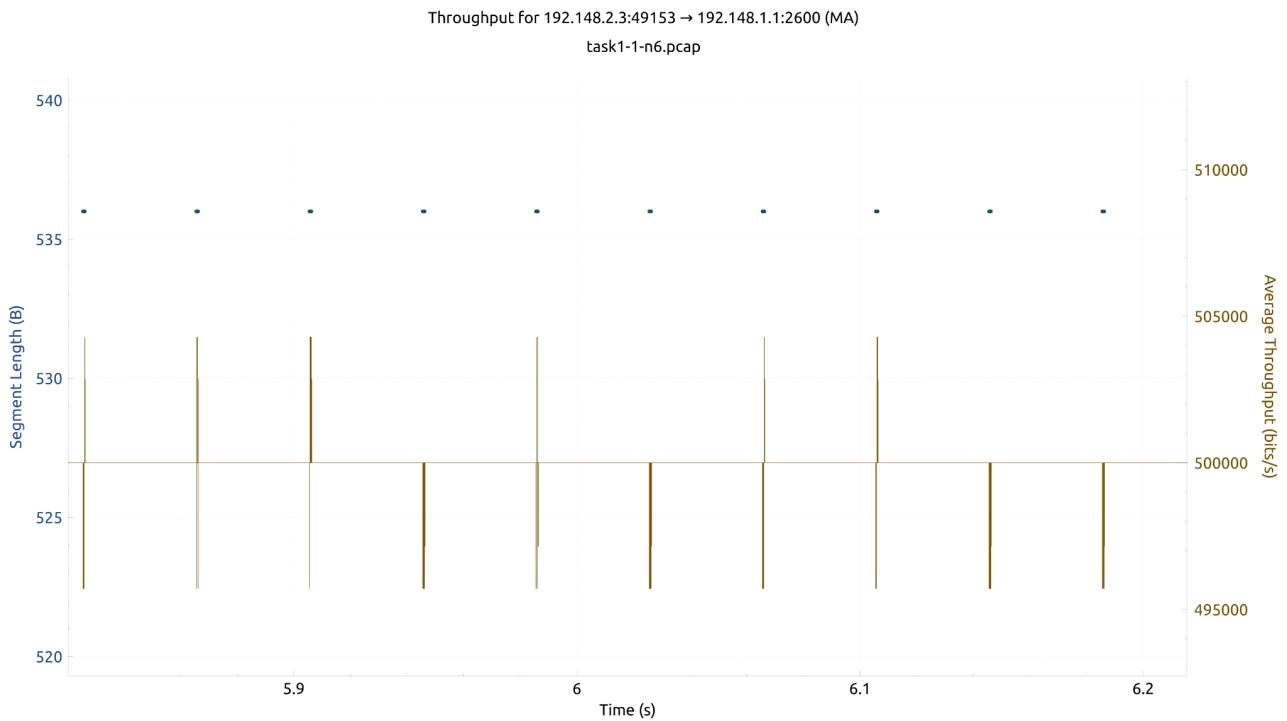


Fig. 13.1 - Grafico throughput

Rispetto al risultato ottenuto in C12 il throughput medio rimane invariato nel tempo, infatti anche se i bit trasferiti nei due istanti sono differenti, con una variazione di 1000000 bit, il bit-rate rimane costante, probabilmente dovuto ad una velocità di trasmissione costante nel tempo.

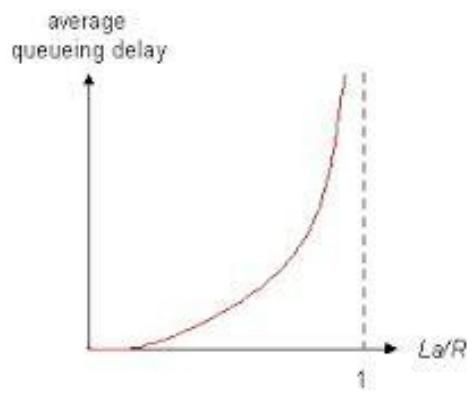
C14: [Extra] Ritardo di accodamento vs congestione: Disegnare un grafico che mostri il ritardo di accodamento in funzione del livello di congestione in rete.

Il ritardo di accodamento è il tempo in cui un pacchetto resta in un nodo, generalmente in un buffer, in attesa di essere trasmesso.

È un ritardo che dipende dal traffico di rete, quindi dalla congestione dovuta a quanti pacchetti sono in trasmissione. Questo ritardo non è prevedibile ed è calcolabile approssimativamente con una curva che misura il tempo medio di accodamento in un router in funzione dell'intensità di traffico definita come $\frac{La}{R}$ dove

$L = \text{lunghezza del pacchetto (bit)},$
 $R = \text{frequenza di trasmissione (bps)},$
 $a = \text{tasso medio di arrivo dei pacchetti}.$

Da questo grafico si osserva che il ritardo di accodamento cresce esponenzialmente in funzione della congestione di rete, se $\frac{La}{R} \sim 0$ è molto limitato, se $\frac{La}{R} \rightarrow 1$ cresce in modo non lineare e se $\frac{La}{R} > 1$ è infinito, poiché arriva più "carico" di quanto in realtà possa essere effettivamente elaborato.



C21: Calcolare il throughput medio del flusso TCP a tempo t=5s.

Abilitando le informazioni di Log su PacketSink e su OnOffClient, attraverso la simulazione vediamo che all'istante $t = 5 \text{ sec}$ vengono trasmessi da OnOffClient $T = 126000 \text{ byte} = 1008000 \text{ bit}$. $t_0 = 3 \text{ sec}$ è l'istante in cui la trasmissione viene aperta.

$$TH(5) = \frac{1008000 \text{ bit}}{5-3 \text{ sec}} = 504000 \frac{\text{bit}}{\text{sec}}$$

Questo risultato si può anche osservare nel grafico del throughput medio presente nell'analisi Wireshark attraverso la sezione Statistics/TCP Stream Graphs/Throughput.

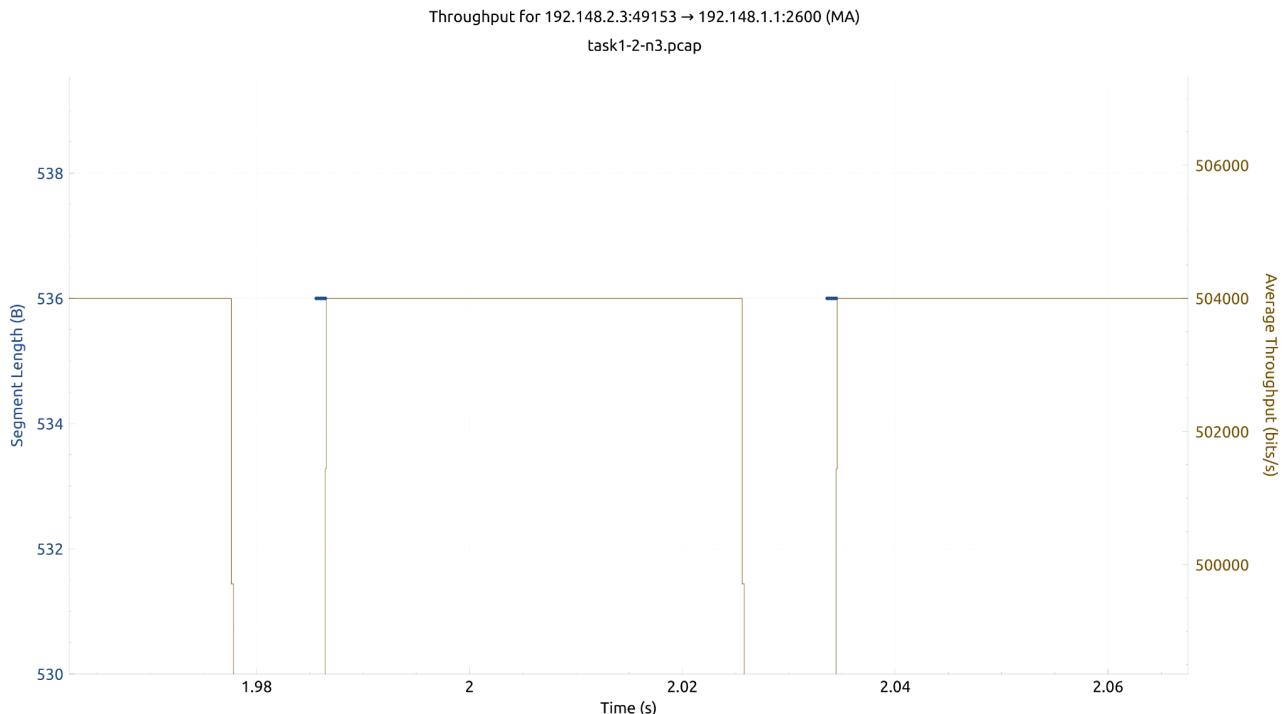


Fig. 21.1 - Grafico throughput

C22: Calcolare il throughput medio del flusso TCP a tempo t=7s. Commentare eventuali cambiamenti rispetto a C21.

Abilitando le informazioni di Log su PacketSink e su OnOffClient, attraverso la simulazione vediamo che all'istante $t = 7 \text{ sec}$ vengono trasmessi da OnOffClient $T = 246000 \text{ byte} = 1968000 \text{ bit}$. $t_0 = 3 \text{ sec}$ è l'istante in cui la trasmissione viene aperta.

$$TH(5) = \frac{1968000 \text{ bit}}{7-3 \text{ sec}} = 492000 \frac{\text{bit}}{\text{sec}}$$

Questo risultato si può anche osservare nel grafico del throughput medio presente nell'analisi Wireshark attraverso la sezione Statistics/TCP Stream Graphs/Throughput.

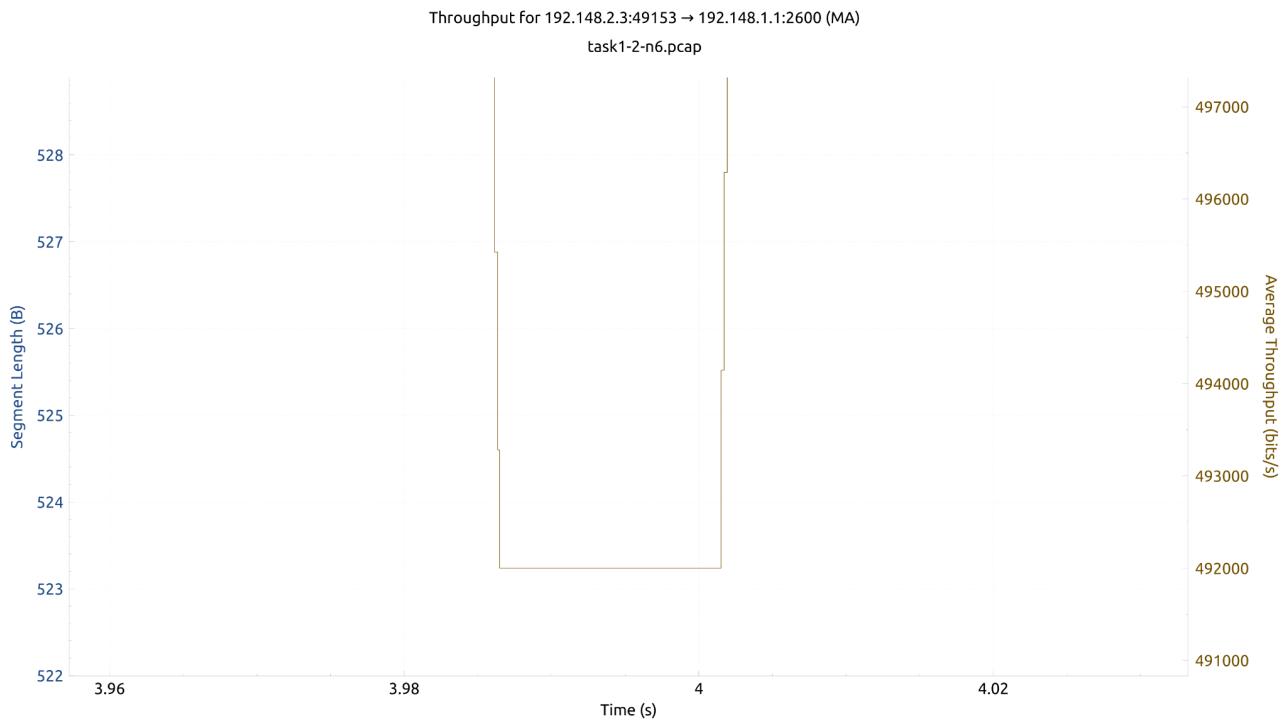


Fig. 22.1 - Grafico throughput - Assumiamo MA=4 per rendere più dettagliato il grafico

Si può notare che a differenza di C02, il throughput è diminuito, proprio perché è diminuita la quantità di bit trasmessa all'istante di tempo $t = 7 \text{ sec}$. Questo implica che da $t = 5 \text{ sec}$ sec a $t = 7 \text{ sec}$ c'è stata una diminuzione dei bit al secondo trasmessi dal client, probabilmente dovuto ad un rallentamento della velocità di trasmissione.

C23: [Extra] Ritardo di accodamento vs congestione: Disegnare un grafico che mostri il ritardo di accodamento in funzione del livello di congestione in rete

//