REPORT HOMEWORK

Configurazione 0

A1) Individuare le varie topologie note che compongono la rete.

In questa rete ci sono delle sottoreti componenti, due reti locali ethernet ciascuna delle quali utilizza un bus lineare condiviso (topologia a bus lineare).
La rete nel suo complesso può anche essere considerata un albero con radice n3 (topologia ad albero).

I collegamenti tra endpoints come quello tra n4 e n5 sono del tipo punto a punto, analogamente sono collegamenti punto a punto anche l0, l1 e l3.
I parametri che distinguono le varie configurazioni non alterano la topologia complessiva della rete e delle sue sottoreti.

A2) Ricostruzione del percorso dei pacchetti attraverso la rete di tutti i flussi simulati usando Wireshark evidenziando i filtri utilizzati per isolare i singoli flussi dello strato di trasporto tra le tracce

Nota: Ogni configurazione a secondo dello stato del canale e della topologia può seguire un percorso diverso, è importante quindi evidenziare eventuali differenze al variare della configurazione e i filtri utilizzati

Nella seguente configurazione i pacchetti TCP partono dal nodo n4, dov'è situata la OnOff Application e vengono inviati al nodo n5 tramite il collegamento Point to Point l2.
Poi vengono inviati a n6 attraverso il collegamento l3.

La connessione LAN tra n6, n7 e n8 non è coinvolta nella comunicazione, infatti la cattura effettuata su ciascuno dei nodi di questo specifico collegamento (non quindi sulle connessioni Point to Point di n6) risulta vuota.

I pacchetti vengono poi spediti da n6 a n3 attraverso il collegamento Point to Point l1 e ancora a n1 attraverso il collegamento Point to Point l0.
Qui, i pacchetti vengono elaborati, come è possibile vedere dalle catture ASCII effettuate sul nodo n1, prima di essere spediti con un Header differente, contenente il protocollo Ethernet in luogo di quello Point to Point, a n2 mediante il CSMA Link che li interconnette.
In n2, i pacchetti vengono raccolti dal Sink e terminano il loro "viaggio".

Filtro usato: "tcp.srcport!=2400"

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

Per calcolare RTT di un pacchetto generico, prendiamo la cattura del nodo client e calcoliamo la differenza tra tempo di arrivo dell'ultimo ACK e tempo di partenza del primo pacchetto (definizione di RTT), che è in linea con le aspettative e il grafico.

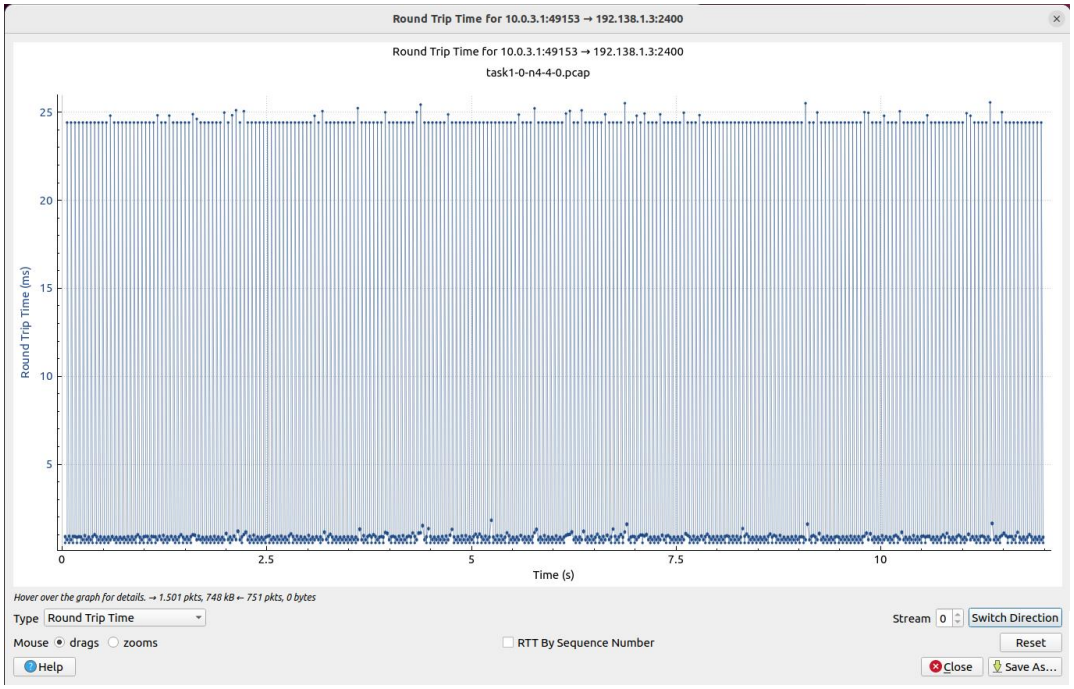
Calcolo per il primo pacchetto: RTT = 3,035258 - 3,034279 = 0,000979 s.
Come si può vedere dal grafico **[Fig.1]**, alcune volte l'RTT assume valori più alti di quello calcolato, questo perché l'invio del messaggio di ACK riferito all'ultimo frammento di un pacchetto viene inviato solo dopo che il client ha cominciato a trasmettere il pacchetto successivo, quindi con un ritardo sensibilmente più alto.

Gruppo di lavoro: #34
Componenti: Alessio Maiola (GL), Alessandro Maone, Andrea De Carlo, Davide Caronna

Questo è visibile mediante il calcolo per il secondo pacchetto: RTT = 3,082814 - 3,058279 = 0,024535 s.

File di cattura usati: File di pcap su n4.

Appendice: nel grafico Wireshark **[Fig.1]** calcola l'RTT sui singoli pacchetti a livello di rete. Noi invece ci siamo attenuti alla definizione di RTT per l'intero segmento.



[Fig.1]

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

In questa configurazione il bottleneck è il CSMA link della rete locale che contiene il nodo n2 che è il ricevente della comunicazione TCP tra n2 e n4 perché il suo Data Rate è quello più basso nel collegamento considerato.

Una possibile soluzione a ciò è aumentare la banda trasmissiva del CSMA link in oggetto oppure collegare il nodo n2 ad un altro nodo fuori dalla LAN o direttamente al destinatario utilizzando un link con data rate maggiore.

C01) Calcolare il throughput istantaneo del flusso TCP.

Per il throughput istantaneo, bisogna identificare gli istanti di tempo rilevanti in cui calcolare il throughput: il client alterna periodi di attività a periodi di inattività, dove il throughput istantaneo è ovviamente 0 bps. Il client invia pacchetti della dimensione di 1500 B, ma nell'attraversare la rete il pacchetto viene frammentato in due pacchetti da 590 B, uno da 482 B. Inoltre, nel collegamento LAN il Point to Point Protocol viene sostituito da quello Ethernet e le dimensioni diventano rispettivamente pari a 606 e 498 B.

Per il calcolo del throughput consideriamo la quantità di dati inviati dal client, cioè 590*2+482=1662 B, trasmessi, nel caso del primo pacchetto, in 3.035183-3.034279 s, con throughput istantaneo di 14.708 Mbps in t=3.035s.

Il throughput è influenzato anche dalla congestione della rete e varia. Ad esempio, nell'invio del pacchetto 417 vale 1662 / (12.995274−12.994297) Bps = 13.609 Mbps in t=12.995s.

File pcap utilizzato: file di cattura su n4 per i tempi di partenza e su n2 per i tempi di arrivo.

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

A tempo t=4s la quantità di dati trasmessa dal flusso TCP si può evincere dai Sequence Numbers dei pacchetti inviati: a t=3.99 s, il server invia al client la comunicazione di ricezione dell'ultimo pacchetto considerato, con ACK_no=61501.
Ciò implica che finora sono stati inviati 61500/1500 = 41 pacchetti.

Noi, tuttavia, sappiamo che ad attraversare la rete sono tre pacchetti frammentati delle dimensioni sopracitate; quindi, il throughput medio sarà (1662 bytes * 41) / (4-3) s = 68142 Bps = 545 kbps. Ciò è visibile anche dall'analisi degli I/O Graphs tracciati sulla cattura dei pacchetti inviati da n4 tramite Wireshark.

Nel grafico che si riferisce a n2, invece, il throughput sembra più elevato, questo perché n1 modifica i pacchetti, come anticipato nella precedente risposta, aumentandone la dimensione. All'interno della LAN, difatti, il throughput medio è pari a (606*2 + 498) * 41 bytes / 1 s = 70110 Bps = 561 kbps.

File pcap utilizzato: file di cattura su n4 per i tempi di partenza e su n2 per i tempi di arrivo.

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

A tempo t=7s l'ultimo ACK_no inviato dal server è pari a 249000, perciò sono stati inviati 166 pacchetti. Il throughput, calcolato come nella precedente risposta, risulta pari a 551 kbps.

Il seguente risultato si discosta davvero di poco rispetto al precedente. Questo ci sembra un risultato evidente, poiché il livello di congestione della rete è minimo, se non nullo: il client invia un pacchetto ogni 0.024s, mentre il tempo di trasferimento per il singolo pacchetto si attesta intorno agli 0.0009 s. Ciò implica che la rete è vuota per la maggior parte del tempo e anche quando trasmette lo fa a un bitrate inferiore rispetto alla capacità del collo di bottiglia della rete, che è 25 Mbps, come si può evidenziare dalla risposta alla prima domanda.

File pcap utilizzato: file di cattura su n4 per i tempi di partenza e su n2 per i tempi di arrivo.

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

Sfruttando i SEQ_no e gli ACK_no si vede che durante tutta la comunicazione la quantità totale di pacchetti inviati è pari a 499. Ogni pacchetto mediamente impiega 0.0009 s per essere trasferito completamente dal momento in cui viene inviato. Perciò, una possibile stima del ritardo complessivo di ogni pacchetto inviato è pari a 499 pacchetti * 0.0009 s/pacchetto = 0.4491 s. La stima per il ritardo medio del pacchetto deriva dall'analisi Wireshark dei pacchetti, calcolando la differenza tra il tempo d'arrivo dell'ultimo dei tre pacchetti frammentati al nodo n2 e la partenza del primo dei tre pacchetti dal nodo n4. Questo è il valore della differenza tra i tempi di invio e ricezione segnalata dai log e si può ricavare utilizzando il throughput istantaneo durante l'invio di un pacchetto in C0 e la grandezza in bit del pacchetto mediante la formula $delay\_pacchetto = \frac{dim\_pacchetti\_inviati}{TH\_istantaneo}$.
File pcap utilizzato: file di cattura su n4 per i tempi di partenza e su n2 per i tempi di arrivo.

Gruppo di lavoro: #34
Componenti: Alessio Maiola (GL), Alessandro Maone, Andrea De Carlo, Davide Caronna

Configurazione 1

A1) Individuare le varie topologie note che compongono la rete.

In questa rete ci sono delle sottoreti componenti, due reti locali ethernet ciascuna delle quali utilizza un bus lineare condiviso (topologia a bus lineare).
La rete nel suo complesso può anche essere considerata un albero con radice n3(topologia ad albero).

I collegamenti tra endpoints come quello tra n4 e n5 sono del tipo punto a punto, analogamente sono collegamenti punto a punto anche l0, l1 e l3.
I parametri che distinguono le varie configurazioni non alterano la topologia complessiva della rete e delle sue sottoreti.

A2) Ricostruzione del percorso dei pacchetti attraverso la rete di tutti i flussi simulati usando Wireshark evidenziando i filtri utilizzati per isolare i singoli flussi dello strato di trasporto tra le tracce
Nota: Ogni configurazione a secondo dello stato del canale e della topologia può seguire un percorso diverso, è importante quindi evidenziare eventuali differenze al variare della configurazione e i filtri utilizzati

In questa configurazione, il primo flusso TCP segue lo stesso percorso già descritto nella precedente configurazione, ad eccezione della parte finale, ad esso però si sovrappone un ulteriore flusso, che parte dal nodo n8 e attraversa tutto il CSMA link attraversando i nodi n7 e n6. In quest'ultimo nodo i due flussi attraversano la stessa sezione della rete, attraversando i nodi n3, tramite il link l1, e n1, tramite il link l0. Infine, il flusso partito da n4 termina la propria percorrenza attraversando il CSMA link e venendo mandato al Sink in n0, mentre il flusso partito da n8 termina nel Sink in n2. Per isolare i flussi tramite Wireshark occorre utilizzare i filtri:

"(tcp.srcport!=7777 && tcp.port==7777)" permette di isolare il flusso di pacchetti da n4 a n0

"(tcp.srcport!=2400 && tcp.port==2400)" permette di isolare il flusso di pacchetti da n8 a n2

"(tcp.srcport!=7777 && tcp.srcport!=2400)" permette di visualizzare solo i due flussi.

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

In questa configurazione sono presenti due flussi che verranno trattati separatamente, ma il comportamento è analogo a quello descritto per la configurazione 0, come è visibile anche dal grafico.

Flusso n8->n2:
Calcolo per il primo pacchetto: RTT = 2,092004 - 2,089468 = 0,002536 s.
Anche qui, talvolta l'invio dell'ACK avviene solo dopo che il client ha interrotto la trasmissione e cominciato quella del pacchetto successivo.
Calcolo per il secondo pacchetto: RTT = 2,234074 - 2,161468 = 0,072606 s.

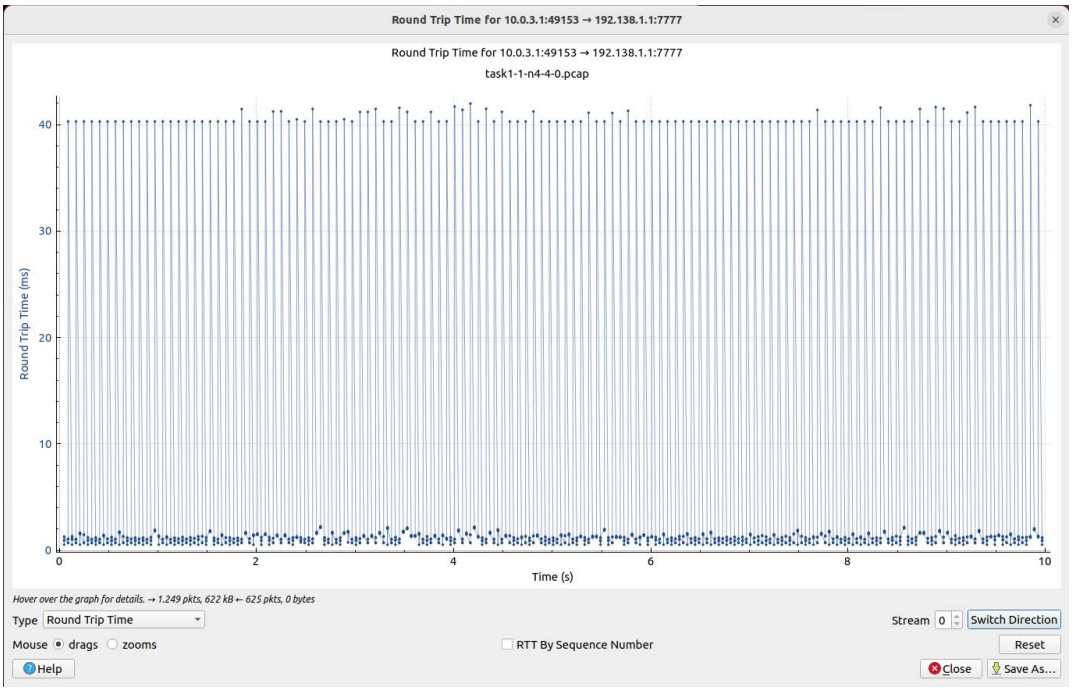
Flusso n4->n0:
Calcolo per il primo pacchetto: RTT = 5,049753 - 5,048279 = 0,001474 s.
A causa dello stesso fenomeno sopracitato l'RTT può assumere valori più alti.

Gruppo di lavoro: #34
Componenti: Alessio Maiola (GL), Alessandro Maone, Andrea De Carlo, Davide Caronna

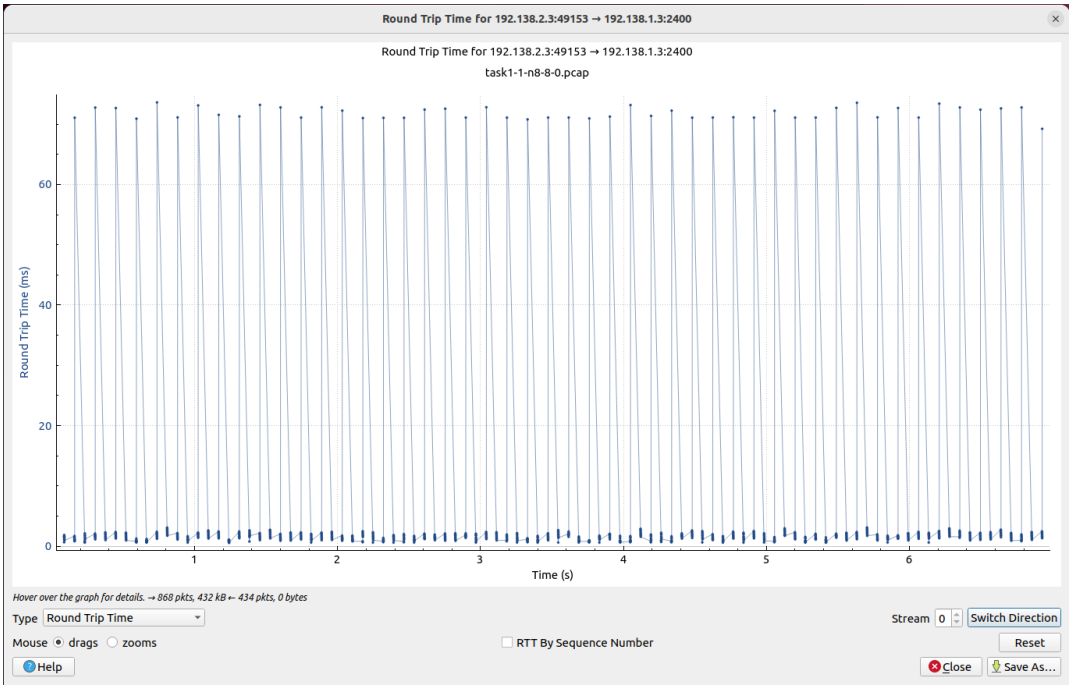
Calcolo per il secondo pacchetto: RTT= 5,128814 - 5,088279 = 0,040535 s.

I grafici elaborati da Wireshark **[Fig.2] [Fig.3]** risultano leggermente imprecisi in quanto sono valide le considerazioni riportate in appendice (Vedi A3, configurazione 0).

File di cattura usati: catture pcap su n4 e n8.



[Fig.2]



[Fig.3]

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

In questa configurazione il bottleneck è il CSMA link della rete locale che contiene il nodo n2 e il nodo n0, che sono i riceventi della comunicazione TCP tra n2 e n8 e tra n0 e n4 perché il suo Data

Rate è quello più basso nei collegamenti considerati. Inoltre, tale link è anche condiviso dalle due comunicazioni.

Una possibile soluzione a ciò è aumentare la banda trasmissiva del CSMA link in oggetto oppure collegare il nodo n2 o il nodo n0 ad un altro nodo fuori dalla LAN o direttamente al destinatario utilizzando un link con sufficiente data rate.

C11) Calcolare il throughput medio dei flussi TCP.

Dalla cattura di un qualsiasi nodo attraversati da entrambi i flussi, isolati con i filtri, è possibile vedere che il flusso da n4 a n0 trasmette complessivamente 622500 B (ultimo ACK_no ricevuto), quindi 249 pacchetti di 2500 B, ciascuno frammentato in 4 pacchetti di 590 B e uno di 410 B.

Il throughput medio di questo flusso risulta pari a (590 * 4 + 410) * 249 / (15-5) = 68973 Bps = 552 kbps. Il flusso da n8 a n2 invia complessivamente 432000 B (anche qui ultimo ACK_no ricevuto dal suo server), cioè 96 pacchetti di 4500 B, frammentati stavolta in 8 pacchetti di 590 B e uno di 266 B. Il throughput medio di questo flusso, quindi, risulta pari a (590 * 8 + 266) * 96 / (9-2) = 68379 Bps = 547 kbps.

Il throughput medio complessivo dei due flussi è quindi pari a:
[(590*8+266)*96+(590*4+410)*249]/(15-2) = 89876 Bps = 719 kbps

File pcap utilizzato: file di cattura su n4 per i tempi di partenza e su n0 per i tempi di arrivo e file di cattura su n8 per i tempi di partenza e su n2 per i tempi di arrivo.

C12) Calcolare il throughput medio del flusso TCP n8 verso n2 a tempo t=6s.

A tempo t = 6s la quantità di bytes trasmessa sul flusso TCP si ricava dai Sequence Numbers e, in particolare, si ricava dall'ACK_no=247501. Quindi finora sono stati inviati e ricevuti 247500/4500=55 pacchetti.

Tali pacchetti, attraversando la rete, vengono però frammentati in 8 pacchetti da 590B e uno da 266B, comportando un TH medio pari a ((590*8 + 266) bytes * 55) / (6-2) s = 68557 Bps = 548 kbps. Tale risultato è confermato dall'analisi dell'I/O Graph di Wireshark relativi alla cattura dei pacchetti inviati da n8.

File pcap utilizzato: file di cattura su n8 per i tempi di partenza e su n2 per i tempi di arrivo.

C13) Calcolare il throughput medio del flusso TCP n8 verso n2 a tempo t=8s. Commentare eventuali cambiamenti rispetto a C12.

A tempo t = 8s il server invia ACK_no=373501 relativo all'ultimo pacchetto ricevuto; quindi, in totale i pacchetti inviati da n8 finora sono 373500/4500=83.

Quindi, con la stessa procedura utilizzata in C12, il TH medio risulta pari a ((590*8 + 266) bytes * 83) / (8-2) s = 68973 Bps = 552 kbps.

Come già evidenziato in precedenza in C03, tale risultato, che si discosta di poco dal TH calcolato in C12, ci sembra coerente con il fatto che il livello di congestione della rete è minimo, se non proprio nullo.

File pcap utilizzato: file di cattura su n8 per i tempi di partenza e su n2 per i tempi di arrivo.

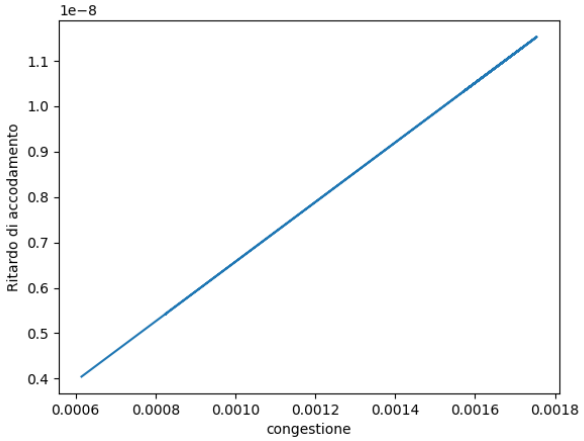
C14) [Extra] *Ritardo di accodamento vs congestione*: Disegnare un grafico che mostri il ritardo di accodamento in funzione del livello di congestione in rete

Scelgo come nodo n3 in riferimento al quale effettuare la mia analisi.

Il nostro calcolo prende in esame i flussi provenienti da n8 a n2 e da n4 a n0. Entrambi i flussi utilizzano il collegamento che parte da n6 e arriva a n3.
Calcolo l'intensità di traffico $I = La/R$ con L dimensione dei pacchetti, a tasso di arrivo dei pacchetti e R bit-rate del link.
Mostro attraverso un grafico il ritardo di accodamento in funzione della congestione calcolato come $I(L/R)(1-I)$.
Utilizzo uno script Python per calcolare tale grafico **[Fig.4]**. Alleghiamo tale script(graphconf1.py).
I dati sono stati esportati da Wireshark. Alleghiamo il file contenente i dati(conf1.csv).

File pcap utilizzato: “task1-1-n3-3-1.pcap”.
Filtri usati: "ip.src==192.138.2.3 or ip.src==10.0.3.1".

Si nota per ottenere il grafico che mostriamo nella risposta è necessario che il file contenente i dati sia nella stessa cartella dello script python.



[Fig.4]

Configurazione 2

A1) Individuare le varie topologie note che compongono la rete.

In questa rete ci sono delle sottoreti componenti, due reti locali ethernet ciascuna delle quali utilizza un bus lineare condiviso (topologia a bus lineare).
La rete nel suo complesso può anche essere considerata un albero con radice n3(topologia ad albero).

I collegamenti tra endpoints come quello tra n4 e n5 sono del tipo punto a punto, analogamente sono collegamenti punto a punto anche l0, l1 e l3.
I parametri che distinguono le varie configurazioni non alterano la topologia complessiva della rete e delle sue sottoreti.

A2) Ricostruzione del percorso dei pacchetti attraverso la rete di tutti i flussi simulati usando Wireshark evidenziando i filtri utilizzati per isolare i singoli flussi dello strato di trasporto tra le tracce

Nota: Ogni configurazione a secondo dello stato del canale e della topologia può seguire un percorso diverso, è importante quindi evidenziare eventuali differenze al variare della configurazione e i filtri utilizzati

Nella seguente configurazione i pacchetti UDP indirizzati all'Echo Server e quelli in risposta dal Server al Client seguono un percorso affine al caso precedente. Il pacchetto, dopo essere partito da n8 ed aver attraversato n7, n6, n3 e n1, viene inviato attraverso il CSMA link all'Echo Server in n2, che invia la risposta, la quale seguirà il percorso opposto (n2, n1, n3, n6, n7 e n8 saranno attraversati con i rispettivi link che li collegano). Il percorso effettuato dai pacchetti TCP che il nodo n4 con OnOff Application invia al Sink del nodo n2 è esattamente lo stesso descritto in A2 per la configurazione 0.
I pacchetti UDP, che l'OnOff Application in n7 invia, attraversano il nodo n6 tramite il link CSMA, poi il nodo n3, attraverso il link l1, infine n1 attraverso l0. Infine, il CSMA link inoltra i pacchetti al nodo n0.

I filtri utilizzati per isolare i flussi sono:
"udp.srcport!=2500 and udp.port==2500" UDP OnOffApp

"tcp.srcport!=2600 and tcp.port==2600" TCP OnOffApp

"udp.srcport!=63 and udp.port==63" Echo Client -> Echo Server

"udp.srcport==63" Echo Server -> Echo Client

"udp.port==63" Echo Server <-> Echo Client

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

In questa configurazione sono presenti tre flussi che verranno, tuttavia l'RTT è definito solo per i flussi TCP; quindi, verrà calcolato solo per il flusso da n4 a n2, con la stessa metodologia usata per le precedenti configurazioni.

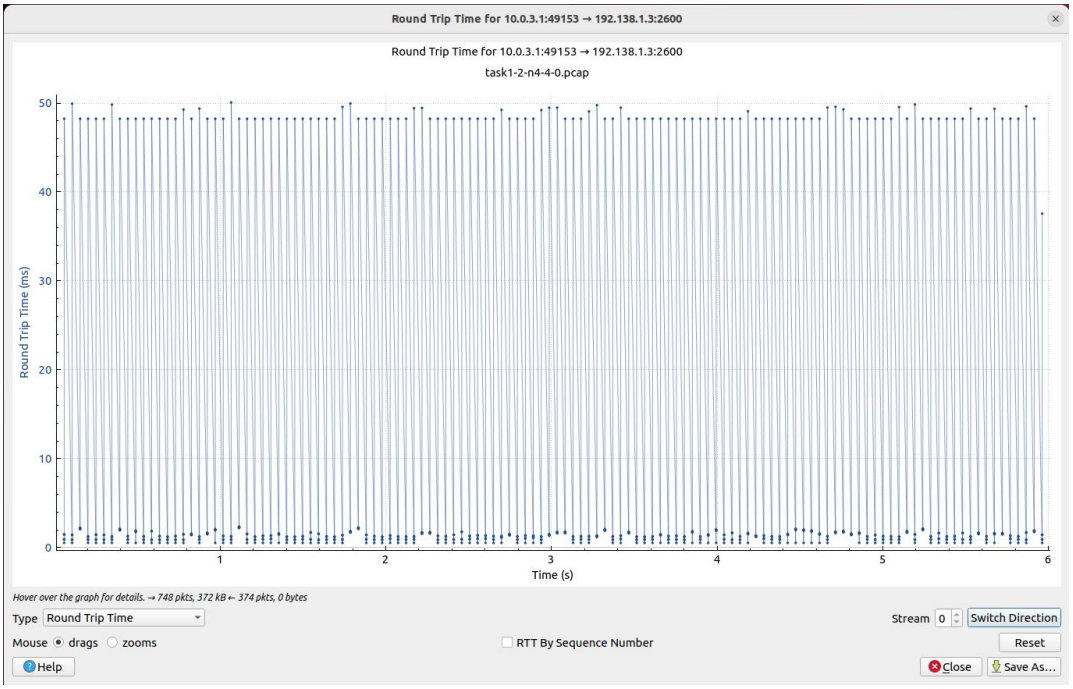
In questa configurazione, come si evince dalla cattura sul nodo n4, la segnalazione dell'ACK per l'arrivo completo di un pacchetto arriva sempre dopo che è terminato l'invio del primo frammento

del pacchetto successivo. Perciò il RTT ha sempre un valore più elevato di quelle mostrati nella parte bassa del grafico.

Calcolo RTT per il primo pacchetto: 3,106814 − 3,058279 = 0,048535 s.

Poiché il grafico **[Fig.5]** si riferisce ai singoli frammenti dei pacchetti, per alcuni di essi l'RTT assume valori inferiori. Per mostrarlo calcoliamo l'RTT anche per il primo frammento del primo pacchetto: 3,058814 − 3,058279 = 0,000535 s.

File di cattura usati: catture pcap su n4.



[Fig.5]

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

Il bottleneck è il CSMA link della LAN contenente n2 e n0. Infatti, tutte le comunicazioni vengono trasmesse o ricevute passando anche attraverso questa lan. Quindi una contromisura può essere aumentare la banda del CSMA link in oggetto oppure collegare i server ai clients corrispondenti o comunque collegarli al di fuori della LAN.

È importante sottolineare che, una volta risolto tale bottleneck, andrebbe considerato anche aumentare la banda del CSMA link della LAN contenente il nodo n8 oppure collegare al di fuori della LAN tale nodo, che potrebbe a sua volta rallentare alcuni flussi della configurazione considerata.

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

A tempo t=5s la quantità di dati trasmessi dal flusso TCP è ricavabile dai Sequence Numbers dei pacchetti inviati ed è pari a 123000B (ultimo SEQ_no + dim. payload), quindi dopo 5s sono stati inviati 123000/3000=41 pacchetti. Sapendo però che la frammentazione durante l'attraversamento della rete produce 5 pacchetti da 590B e un pacchetto da 374B, avremo un TH medio di (3324 bytes * 41) / (5-3) s = 66142 Bps = 545 kbps.

Tale valore si discosta di poco da quello riscontrabile dall'analisi dell'I/O Graph prodotto da Wireshark dalla cattura dei pacchetti inviati da n4.

File pcap utilizzato: file di cattura su n4 per i tempi di partenza e su n2 per i tempi di arrivo.

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

Utilizzando il medesimo procedimento e le medesime considerazioni di C21, a t=7s rileviamo ACK_no=248681 e, aggiungendo la dimensione dell'ultimo pacchetto inviato dal client (il cui ACK salta e viene unito a quello del pacchetto successivo), otteniamo (248680+320) / 3000 = 83 pacchetti inviati da n4 finora.

Il relativo TH medio, considerando i pacchetti frammentati, è quindi pari a (3324 bytes * 83) / (7-3) s = 68973 Bps = 552kbps.

Le considerazioni relative al seguente risultato coincidono con quelle relative alle precedenti configurazioni 0 ed 1. Infatti, la rete, non essendo significativamente congestionata, non rischia mai di raggiungere o superare la capacità di bottleneck.

File pcap utilizzato: file di cattura su n4 per i tempi di partenza e su n2 per i tempi di arrivo.

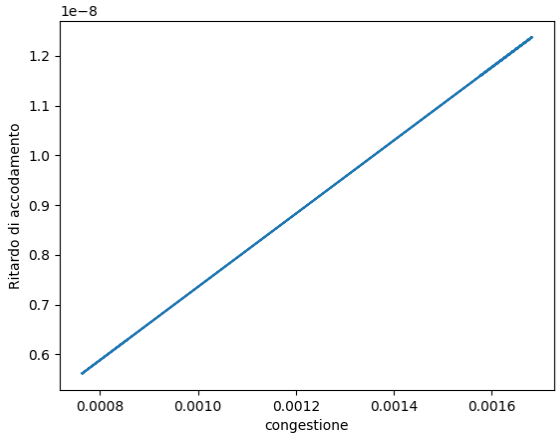
> C23) [Extra] *Ritardo di accodamento vs congestione*: Disegnare un grafico che mostri il ritardo di accodamento in funzione del livello di congestione in rete

Scelgo come nodo n3 in riferimento al quale effettuare la mia analisi.

Come nella configurazione precedente tutti i flussi utilizzano il collegamento che parte da n6 e arriva a n3. Calcolo l'intensità di traffico $I = La/R$ con L dimensione dei pacchetti, a tasso di arrivo dei pacchetti e R bit-rate del link.
Mostro attraverso un grafico il ritardo di accodamento in funzione della congestione calcolato come $I(L/R)(1-I)$.
Utilizzo uno script Python per calcolare tale grafico. Alleghiamo tale script(graphconf2.py). I dati sono stati esportati da Wireshark. Alleghiamo il file contenente i dati(conf2.csv).

File pcap utilizzato: task1-2-n3-3-1.pcap.
Filtri usati: "ip.src==192.138.1.3 or ip.src==10.0.3.1 or ip.src==192.138.2.2".

Si nota per ottenere il grafico **[Fig.6]** che mostriamo nella risposta è necessario che il file contenente i dati sia nella stessa cartella dello script python.



[Fig.6]