POLITECNICO DI TORINO

A.A. 2019/2020

Report 2:

Stima della velocità di trasmissione a livello fisico tramite RTT minimo

Student:

Simone Galota - 233727 – Gruppo 16

Teacher:

Prof. Marco Mellia

Course:

Laboratorio di Internet

Descrizione Setup:

Gli esperimenti sono stati svolti cercando di creare il più possibile lo scenario del laboratorio, quindi senza utilizzare VMs. Per quanto riguarda la LAN fisica è stato utilizzato un home router il cui Default Gateway si trova all'indirizzo 192.168.1.1 e a cui sono connessi i seguenti dispositivi:

Device	Sistema Operativo	Indirizzo IP – connessione - velocità
PC (dal quale partono i ping)	Ubuntu (Live da usb)	192.168.1.17 – cablata – 100 Mb/s o 10 Mb/s
Mac	MacOS	192.168.1.221 – cablata – 100 Mb/s
iPhone	iOS	192.168.1.76 – WiFi – 100 Mb/s

Sono stati effettuati tre esperimenti: PC con Ubuntu che pinga al default gateway, a MacOS e a iOS. Le interfacce Ethernet saranno configurate a 100 Mb/s (o a 10Mb/s quando possibile) per far sì che, nel calcolo del RTT, T_η fosse effettivamente trascurabile rispetto al tempo di trasmissione T_{TX} (minore è la velocità più grande sarà il tempo di tx). Dove T_η è la somma di tempi di propagazione, tempi di elaborazione dei pacchetti, tempi di risposta ed eventuali tempi di accodamento. Quello che vogliamo misurare sperimentalmente è il RTT minimo al variare della dimensione della PDU a livello 7 (e di conseguenza a L1), che ci consentirà di fare una stima della velocità del canale grazie all'ausilio di un modello appropriato. Tutti i test sono stati svolti con e senza frammentazione a livello IP, in particolare:

- Per dimensioni di PDU < MSS: pacchetti che vanno da 100 a 1400 bytes con passo 100.
- Per dimensioni di PDU = MSS = 1472 bytes. (1 pacchetto)
- Per dimensioni di PDU > MSS: pacchetti che vanno da 2952 a 30000 bytes, con passo 1480. In modo tale da avere a livello 2 dimensione massima (MTU = 1500 bytes) per ogni segmento. (1472 + 8 +20=1500, 2952 + 8 = 1480(+20) + 1480(+20) = 1500 + 1500; e così via...)

In appendice si trovano gli script che permettono di automatizzare gli esperimenti.

Test 1: PC cablato che fa ping a defualt gateway

Questo test è stato svolto con il canale configurato a 10 Mb/s. Se fosse stato svolto a 100 Mb/s l'effetto di T_η avrebbe influito troppo, in questo scenario, sulle misurazioni del RTT. La configurazione della scheda di rete si cambia, su Ubuntu, tramite il comando: *"sudo ethtool -s eth0 speed 10 duplex full autoneg on"*. In questo scenario la trasmissione dei pacchetti avviene senza store&forward, perché una volta che l'home-router ha ricevuto il pacchetto di Echo_request, anziché inoltrarlo ad un altro host della rete, manda direttamente una Echo_reply a chi l'ha pingato. In figura 1 si trova il grafico del RTT minimo (viene preso il minimo così da minimizzare l'errore dovuto a T_η). Sull'asse delle ascisse abbiamo la dimensione del pacchetto a livello 1, così ottenuta:

D(L1-PHY) = S (L7) + 8 (ICMP) + 20 (IP) + 38 (ETH);

Dove **S** è la dimensione al livello 7 passata tramite il comando ping, e gli altri valori sono le dimensioni degli header che si aggiungono man mano che il pacchetto attraversa la pila protocollare OSI fino ad arrivare al livello 1 (PHY).

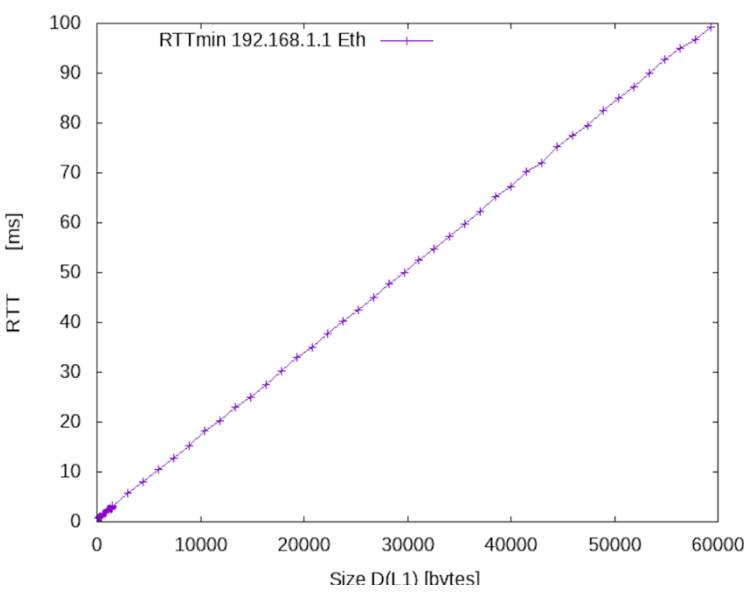

Figura 1

Dal grafico possiamo notare che:

- La retta interpolante i punti non intercetta l'asse delle ascisse precisamente in 0. Qualora la dimensione del pacchetto fosse prossima a 0, nel calcolo del RTT la componente T_η assume rilevanza secondo l'equazione: $RTT = 2T_{TX} + T_\eta$. *[1]*
  Se il T_{TX} → 0, allora RTT = T_η;
- L'andamento è piuttosto lineare e regolare (non sarebbe stato così se il test fosse stato svolto a 100 Mb/s). Maggiore è la dimensione D, più trascurabile è T_η, più è accurato il valore RTT_minimo rappresentato sull'asse delle ordinate. Quindi il RTT_minimo è direttamente proporzionale alla dimensione dei dati che devo trasmettere;

- Non ci sono anomalie e/o ritardi particolari a dimostrazione del fatto che il protocollo Ethernet (802.3) sia più affidabile e semplice rispetto ad altri protocolli di livello 2 (ad es. WiFi – 802.11).

Dal grafico della stima della velocità (fig. 2) si evince come all'inizio il valore sia sottostimato. La stima sarà sempre più corretta man mano che la dimensione D dei pacchetti a livello 1 aumenta. Di conseguenza, se aumenta D, la misura del RTT_min è più precisa perché è molto piccolo l'errore (rispetto a $2T_{TX}$) dovuto a T_η, quindi trascurabile. Secondo l'equazione [1], ha maggior peso il $2T_{TX}$, per cui $RTT \cong 2T_{TX}$. Da ***[1]*** si può anche ricavare il modello per stimare V_C in questo scenario particolarmente semplice:

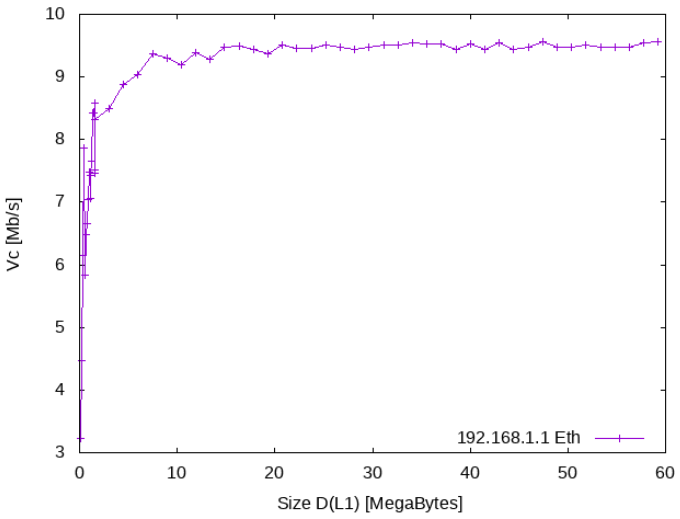

Figura 2

- Essendo $T_{TX} = \frac{D}{V_C} \implies RTT = 2\frac{D}{V_C} \implies$
- $V_C = 2\frac{D}{\mathrm{RTT}_{min}}$;

dove $2T_{TX}$ è il tempo che impiega ogni pacchetto per essere trasmesso da Pc a home-router e ritorno.

Test 2-3: PC cablato che fa ping ad altro host cablato e altro host WiFi

Non potendo configurare il canale DG-MacOS a 10Mb/s, i test sono stati effettuati a 100 Mb/s, così da avere la stessa velocità V_C su entrambi i canali. In questo caso avremo comunque delle buone approssimazioni. Questo scenario è leggermente più complesso a causa della presenza contemporanea di store&forward e frammentazione. In particolare, nel caso dell'host WiFi, assumiamo che l'over-head dovuto a livello 2 sia comunque di 38 byte (in realtà è maggiore, quindi in questo test avremo una sottostima della velocità del WiFi).

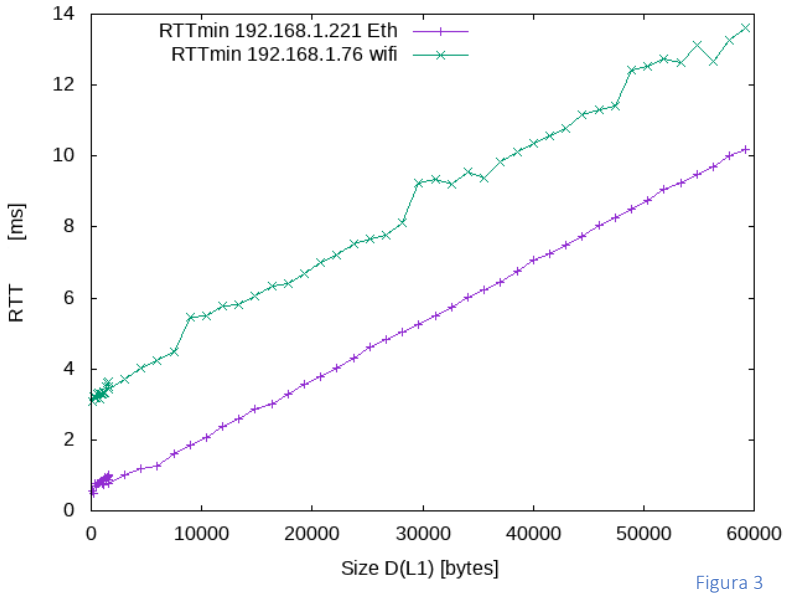

Figura 3

Dal grafico del RTT_minimo, presente in figura 3, notiamo che:

- Nonstante la velocità sia decuplicata rispetto al test 1, la retta intercetta comunque l'asse Y lontano da 0, per gli stessi motivi precedentemente citati.
- Qui, a fronte di una trasmissione al livello applicazione, ne abbiamo due a livello fisico (H1 -> SW e SW -> H2). Quindi in totale:

$$RTT \cong 4T_{TX} = 4\frac{D}{V_C} \quad \textbf{[2]}$$

- Il RTT in WiFi è maggiore di quello in ethernet, nonostante sia un WiFi a 5GHz, ciò è dovuto sia alla sottostima precedentemente citata, sia alla complessità del protocollo 802.11 (ac). In generale, il canale fisico di trasmissione dei bit è composto da onde elettromagnetiche. A causa di questa sua natura è più rumoroso del cavo Eth e soggetto a continue interferenze da parte di altri dispositivi. Quindi si tratta di un canale con una più alta probabilità di errore e/o perdita di pacchetti, con conseguente necessità di ritrasmissioni (e quindi tempi più lunghi).
  Ed è per questi motivi che si ha una minor regolarità e linearità nella retta interpolante i dati del test via WiFi (linea verde);

- Rispetto al test 1, dopo che la dimensione dei pacchetti supera i 1472 bytes, abbiamo una variazione (anche se in questo caso è quasi impercettibile) della direzione della retta. Il segmento che congiunge i punti corrispondenti a dimensioni di D maggiori di 1472 ha una pendenza minore del segmento cha ha come ascisse l'intervallo [0,1472]. Perché?

Perché, nonostante il modello per il calcolo della dimensione dei dati a livello fisico rimanga invariato, cambia il modello per la stima della velocità del canale. Rispetto a quello precedente, c'è da considerare il fatto che ogni pacchetto viene trasmesso su due canali prima di arrivare al ricevitore. In particolare, nel caso in cui ci sia frammentazione (D > 1472), prima che la prima trama inviata arrivi al ricevitore, deve essere ricevuta totalmente e poi inoltrata dallo switch al RX. Per cui, bisogna tener conto di questo ritardo aggiungendo un offset pari a $2T_{MTU}$ al modello precedente (considerando richiesta e risposta).

$$\mathrm{RTT} = 2T_{TX} + 2T_{MTU} = 2\frac{D}{V_C} + 2\frac{MTU}{V_C} = \frac{2}{V_C}(D + MTU) \Rightarrow$$

$$\Rightarrow V_C = \frac{2}{RTT}(D + MTU)$$

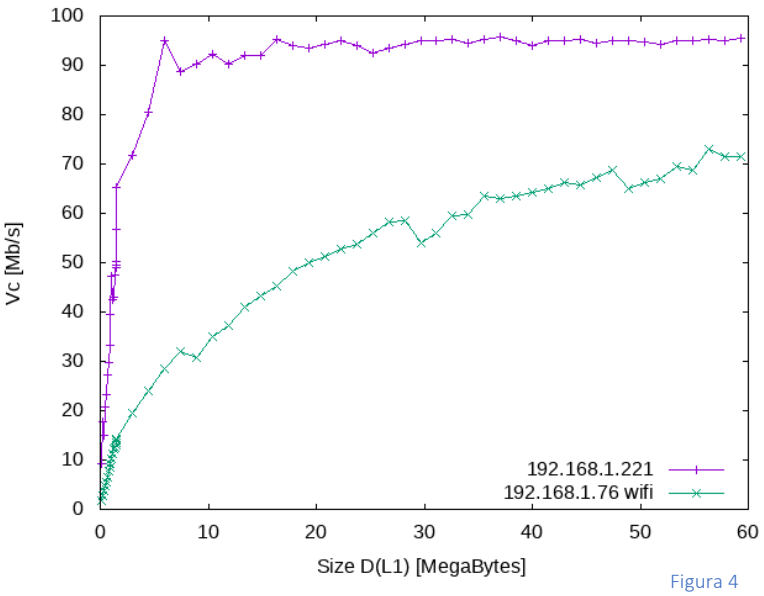

Figura 4

Nel caso in cui non ci sia frammentazione, dalla [2] si ha: $V_C = 4\frac{D}{RTT}$, poiché sì avrà semplicemente la presenza di un intero pacchetto che dovrà transitare da Tx a SW e da SW a Rx, e viceversa. Inoltre, dal grafico presente in figura 4, si capisce come, nel caso di Eth, a RTT minori corrispondano ovviamente velocità maggiori rispetto a WiFi. Infine, notiamo come la trasmissione WiFi raggiunga la velocità massima molto più lentamente di ethernet. Per giustificare questo gap, ai motivi già citati (sottostima modello, interfenze, etc…) si aggiunge il fatto che la qualità del segnale per un dispositivo collegato in modalità wireless dipende anche dalla distanza dall'access-point e del numero di devices presenti sulla rete.

Appendice:

```
#################### SCRIPT AUTOMATIZZAZIONE ESPERIMENTI ##################
#numero di ripetizioni di ogni dimensione
N=20

#target
TARGETS=" 192.168.1.1 192.168.1.221 192.168.1.76" #si S&F

#intervallo tra ripetizioni
I=0.05


#scelta lunghezza pacchetti al livello 7
myLenShort=`seq 100 100 1400`
myLenLong=`seq 2952 1480 60000`


#ripeti ping per ogni lunghezza e per ogni host, e stampa nel file il RTT minimo e la lunghezza
for host in $TARGETS;do

    #cancella i file vecchi
    rm  len${host}.dat rttMin${host}.dat all${host}.dat speed${host}.dat vel.dat

    for len in $myLenShort 1472 $myLenLong; do
        echo $len >> len${host}.dat
        rtt=`sudo ping $host -s $len -c $N -i $I | grep min | cut -d '=' -f 2 | cut -d '/' -f 1`

        echo $rtt >> rttMin${host}.dat


        if [ $len -le 1472 ]; then   #no frammentazione
            echo "(2*2*($len+8+20+38)*8/$rtt)" | bc -l >> vel.dat


        else #si frammentazione
            echo "(2*($len+8+20+38+1538)*8/$rtt)" | bc -l >> vel.dat

        fi
    done

    #incolla le lunghezze e i rispettivi RTT minimi
    paste len${host}.dat rttMin${host}.dat >> all${host}.dat
    paste len${host}.dat vel.dat >> speed${host}.dat

done
```

```
############ script GRAFICO TEST 1 ##############

set term png
set out "RTT_DG.png"
set key left


set  xlabel "Size D(L1) [bytes]"
set ylabel "RTT   [ms]"

plot 'all192.168.1.1.dat' using ($1+8+20+38):2 title "RTTmin 192.168.1.1 Eth" with linespoint

set term png
set out "Velocità_DG.png"
set key right bottom
set xlabel "Size D(L1) [MegaBytes]"
set ylabel "Vc [Mb/s]
plot 'all192.168.1.1.dat' using ($1+8+20+38)/1000:(16*($1+8+20+38))/(1000*$2) title "192.168.1.1 Eth" with linespoint




############ script GRAFICO TEST 2 - 3 ##############

set term png
set out "RTT_S&F.png"
set key left


set  xlabel "Size D(L1) [bytes]"
set ylabel "RTT   [ms]"

plot 'all192.168.1.221.dat' using ($1+8+20+38):2 title "RTTmin 192.168.1.221 Eth" with linespoint,\
     'all192.168.1.76.dat' using ($1+8+20+38):2 title "RTTmin 192.168.1.76 wifi" with linespoint

set term png
set out "Velocità_S&F.png"
set key right bottom
set xlabel "Size D(L1) [MegaBytes]"
set ylabel "Vc [Mb/s]
plot 'speed192.168.1.221.dat' using ($1+8+20+38)/1000:($2/1000) title "192.168.1.221 Eth" with linespoint,\
     'speed192.168.1.76.dat' using ($1+8+20+38)/1000:($2/1000) title "192.168.1.76 wifi" with linespoint
```