Massimo Mantovani 5186259
Matteo Attolini 4677905
Dionis Nikolla 5196050

Relazione ping pong

Osservazioni generali:

Per misurare il tempo eravamo indecisi se usare CLOCK_TYPE (alias di CLOCK_MONOTONIC) oppure CLOCK_REALTIME nella funzione clock_gettime. Abbiamo optato per CLOCK_TYPE perché fissato un punto nel tempo esso aumenta in modo lineare (monotonico), perciò per calcolare il tempo trascorso tra due eventi osservati su una macchina senza un riavvio CLOCK_MONOTONIC è l'opzione migliore.

Client TCP:

Abbiamo avuto problemi alla riga 164 perché inizialmente nel progetto scaricato da aulaweb era presente strlen nella funzione read, poi ci è stato detto di modificarlo in sizeof.
Infatti, la funzione read prende come 2° e 3° parametro rispettivamente il buffer in cui scrivere e il numero di byte da leggere, se il 3° parametro è 'strlen(buffer)' ed il buffer è ancora vuoto, il numero di byte da leggere sarà 0 e come scritto nel manuale la funzione read potrebbe ritornare errore.
Abbiamo osservato, consultando materiale sul web, che l'operatore sizeof per una variabile semplice ritorna il numero di byte occupati dal suo tipo, mentre per un array ritorna il numero di byte occupati dal suo tipo moltiplicato per il numero di elementi dell'array.

Client UDP:

Abbiamo avuto qualche problema nella funzione "do_ping" nello scegliere se usare la funzione "write" o la "nonblocking_write_all". Abbiamo optato per quest'ultima perché dà più garanzie di scrivere tutto il messaggio ed evita che la funzione si blocchi.

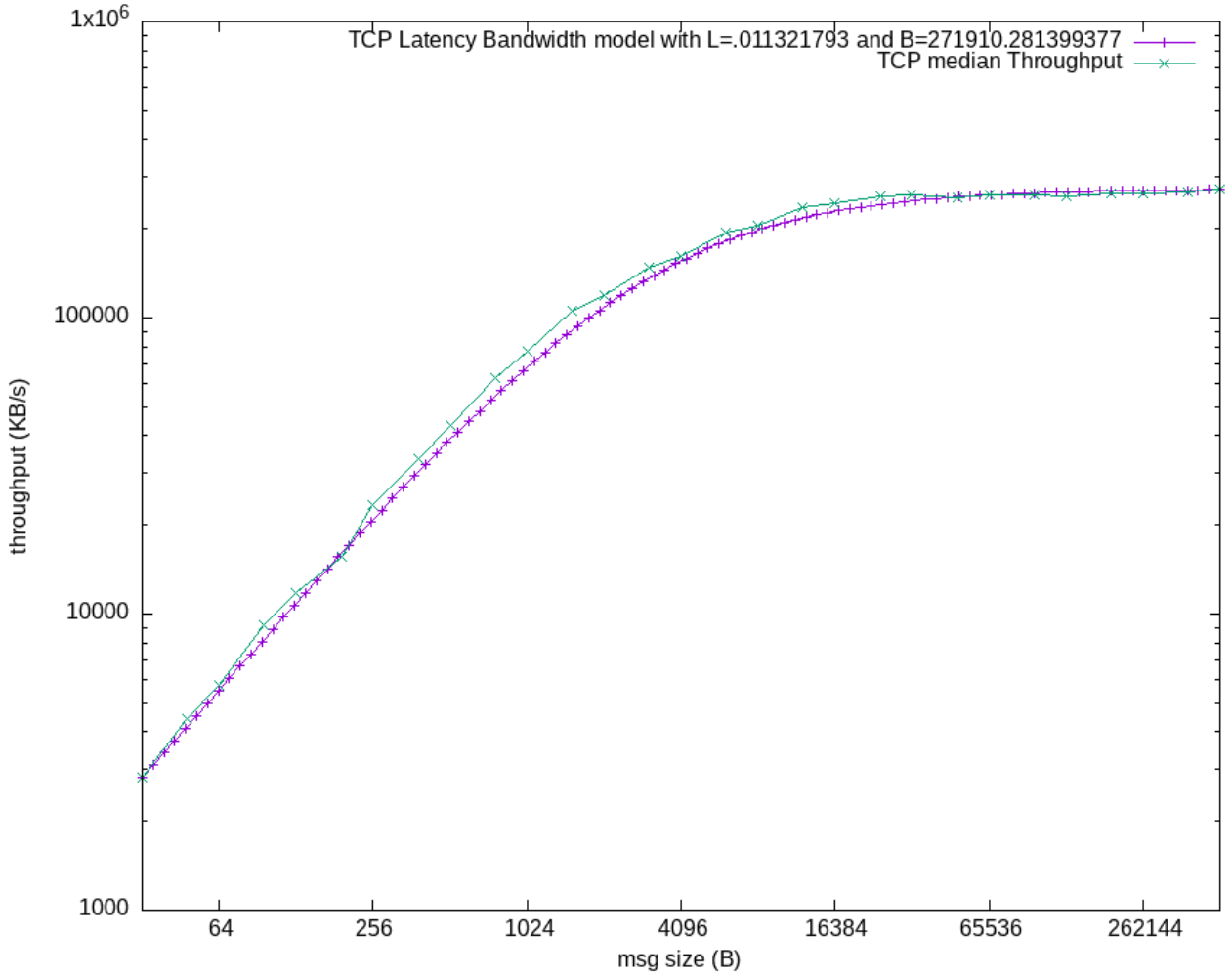
Server PONG:

Non abbiamo avuto particolari problemi a completare il sorgente. La parte che abbiamo trovato più difficile di questo file è stata quella sui segnali per gestire la terminazione del processo figlio (riga 336) che abbiamo risolto leggendo il manuale.

Script e misurazioni:

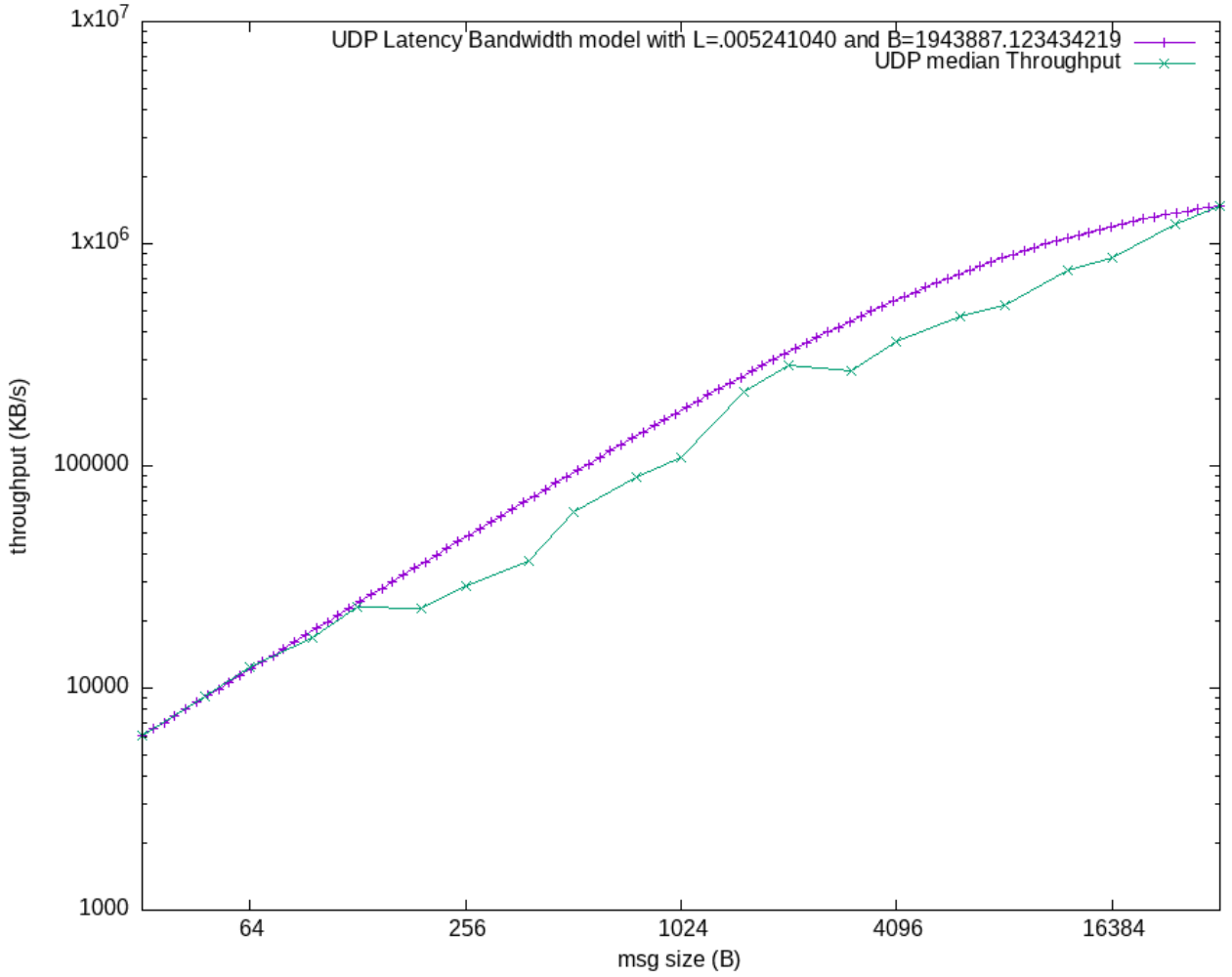
In una vecchia versione dei sorgenti (di cui conserviamo copia, in caso fosse richiesta la visione) siamo riusciti a pingare con successo il server del Dibris (un solo file "broken") tuttavia successivamente né con la versione appena menzionata, né con quella più recente siamo riusciti ad ottenere dei dati corretti con il Dibris.
I dati a cui facciamo riferimento sono stati ottenuti in localhost tramite il comando:
"./mkfile.bash 32 32768 524288 501 localhost 1491"
I grafici sono stati ottenuti tramite lo script "graphics.bash"

Misurazioni con TCP:



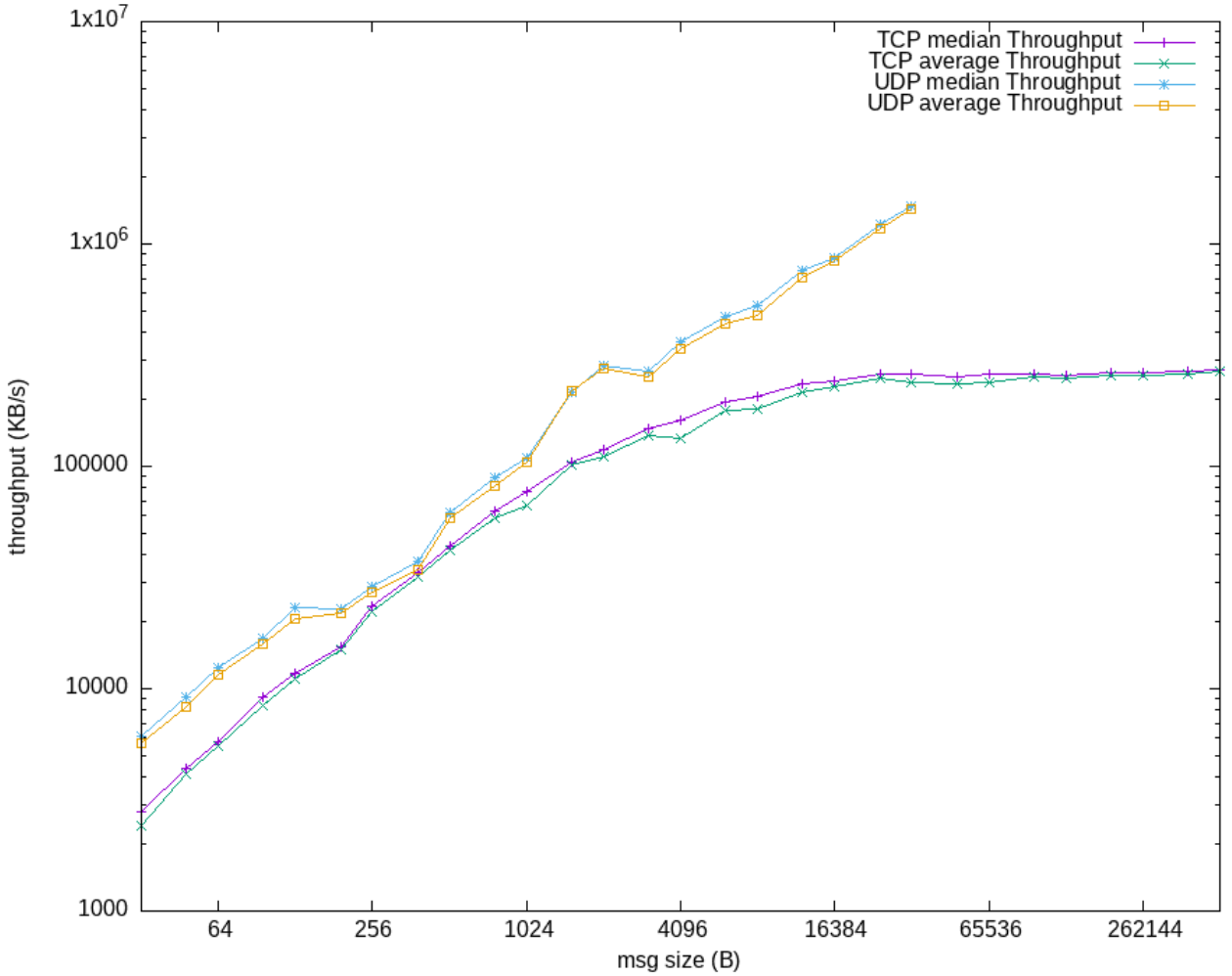
Considerazioni:
Notiamo che il throughput ottenuto con il modello banda-latenza e il throughput mediano ottenuto mediante misurazioni con l'applicazione pingpong sono abbastanza allineate.
Inoltre a partire da dimensione dei messaggi di 32KB la velocità di throughput si stabilizza intorno ai 270323 KB/s

Misurazioni con UDP:



Considerazioni:
Notiamo che, al contrario di TCP, il throughput mediano e quello ottenuto con il modello banda-latenza sono differenti. Questo è dovuto al fatto che il throughput ottenuto con il modello banda-latenza è più “regolare” perché segue una regola (funzione matematica) mentre il throughput mediano è più realistico, cioè rappresenta la “vera” situazione della rete in quel momento ed è influenzato dalle fluttuazioni della rete.

Confronto tra TCP e UDP:



Notiamo che UDP è nel complesso più veloce di TCP perché non deve eseguire il three-way-handshake e non deve aspettare l'acknoledgement di conferma. Inoltre considerando uno scambio reale tra due macchine non eseguirebbe i controlli di sicurezza e affidabilità (congestion control, flow control, timeout per reinvio messaggi persi).
Inotre notiamo che UDP per messaggi di dimensione 32KB (ricordiamo che il limite massimo della lunghezza del datagramma (header + payload) per UDP è 65535 B, dovuto alla rappresentazione in 16 bit nell'header) raggiunge i 1.48285e+06 KB/s di throughput mediano.