L’RTT si dimostra aumentare costantemente all’aumentare del numero di byte inviati. Al variare di pochi byte non ottengo una variazione così netta, ma se l’aumento è di una cifra sufficiente, l’aumento dell’RTT è altamente notabile. Fra valori molto simili di byte si ottengono risultati opposti, in quando la variazione è dettata da frazioni di millisecondo che, all’interno della macchina, sono altamente influenzati dal sistema operativo. I valori rimangono comunque abbastanza simili in quanto la variazione di numero di byte non è estremamente significativa.

Se viene inviato un probe message in più l’errore viene rilevato, lato utente come probe rifiutato, mentre lato server come bye messagge errato in quanto si apsettava proprio un bye message.

Accettiamo tutte le dimensioni di byte da inviare nel payload, salvo esplosioni per eccedenza della memoria.

Abbiamo adottato un messaggio specifico se il messaggio inviato dal client risulta errato.

Per 1K abbimo inteso 1000 e non 1024.

Abbiamo preso come riferimento di dimensione la dimensione dell’ultiml messaggio inviato.

Non sono stati previsti timer per la mancanza di messaggi sulla linea: se uno delle due parti dovesse smettere per qualche strano motivo di inviare pacchetti, l’altra parte non riuscrebbe ad uscire dal suo stato.

Verfico, per uniformità di protocollo, che sia presente un /n, se non arriva il messaggio non termina mai. Gestico le allocazione di 1024 in 1024, piano piano, con realloc per avere una gestione uniforme di tutti i messaggi.

Dai plots è possibile notare come l’RTT tende sempre ad aumentare all’aumentare dei byte inviati, non di molto probabilmente perché l’aumento non è sufficiente per far percepire una reale differenza, in quanto rimare nell’ordine di un millesecondo o meno. Non sono riportati nei grafici casi di invio di grosse moli di byte, ma in quei casi si nota una percebile differenza di tempo nell’ordine dei millisecondi. Il THROUGPUT tende a raddoppiare ogni volta che viene raddoppiato il numero di bytes inviati, infatti all’aumentare massiccio di byte inviati l’RTT rimane quasi invariato, dunque si ha un ordine di proporzione pari all’aumento dei byte inviati.

Quello che è possibile notare da grafici con delay è che il throuput tende a mantenere una crescita come nel caso precendente, ma rimanendo in un ordine di grandezza tanto più basso quanto più alto il delay. L’rtt varia di ordine di grandezza ma sostiene una crescita esattamente come nel caso precednete.

Abbiamo adottato una lettura da file per la profilazione delle informazioni al client relative all’hello message. Il formato è:

….

Il server delay lavora in millisecondi.

Quando eseguo il calcolo dell’RTT prendo come riferimento la dimensione dell’ultimo probe message inviato, che può avere dimensione diversa rispetto a quelli precedenti.

Le informazioni vengono controllate una volta lette da file: quindi se è presente un errore nei valori digitati dall’utente il client non fa partire il servizio, e di conseguenza non contatta il server.

La ricezione dei messaggi è un po complessa: è stata gestita al fine di poter leggere più volte dal buffer (anche non pieno) e di assemblare il messaggio piano piano, eventualemente anche riallocando lo spazio. Finchè non incontra un \n non si arresta: quindi un messaggio preve, che potrebbe essere consegnato in più momenti dal protocollo TCP, non crea problemi.