Relazione PingPong

Componenti:

- Samuele Barrago
- Daniele Sacco
- Lorenzo Migliore

Il **Pong Server** è un'applicazione che risponde ai messaggi inviati dai client **Ping**, sia tramite protocollo **TCP** che **UDP**.
Il server è progettato per accettare **connessioni multiple** da parte dei client Ping e rispondere alle richieste in modo efficiente.

Errori riscontrati e soluzioni adottate

Durante lo sviluppo del Pong Server, abbiamo riscontrato e risolto diversi problemi. Di seguito riportiamo i principali con le relative soluzioni:

- **Invio dei pacchetti**
    - **Problema**: L'invio dei messaggi attraverso il socket in modalità bloccante non riusciva a inviare tutti i dati.
    - **Soluzione**: Abbiamo sostituito la funzione send con una funzione personalizzata chiamata blocking_write_all.
- **Scrittura del numero del messaggio nel buffer**
    - **Problema**: La scrittura del numero del messaggio (msg_no) all'inizio del buffer non funzionava correttamente.
    - **Soluzione**: Abbiamo utilizzato la funzione sprintf per formattare correttamente la stringa.
- **Ricezione pacchetti dal socket**

    - **Problema**: La ricezione delle risposte in modalità non bloccante con timeout presentava problemi.

    - **Soluzione**: Abbiamo implementato un ciclo for per ricevere i dati in più passaggi, anziché in un'unica chiamata.

- **Segmentation fault alla ricezione di tutti i pacchetti**

    - **Problema**: Dopo aver ricevuto tutte le risposte, il programma sollevava un segmentation fault.

    - **Soluzione**: L'errore era legato alla gestione degli argomenti nella funzione timespec_delta2milliseconds.

Debug

Durante lo sviluppo del server Pong, abbiamo adottato un approccio graduale al **debug**.
Inizialmente abbiamo testato il server con il tool gc_*ping, successivamente con i nostri programmi ping.

Abbiamo inoltre utilizzato la modalità **debug del Makefile** per individuare e correggere gli errori. Questo approccio ci ha permesso di migliorare il server in modo efficace e incrementale.

Utilizzo

Il Pong Server deve essere eseguito specificando la **porta** su cui deve rimanere in ascolto.
Esempio di esecuzione:

```
./pong_server 1491
```

Poi si apre un altro terminale e si esegue ad esempio:

```
./tcp_ping 127.0.0.1 1491 100 5
```

e lo stesso vale per testare il client udp

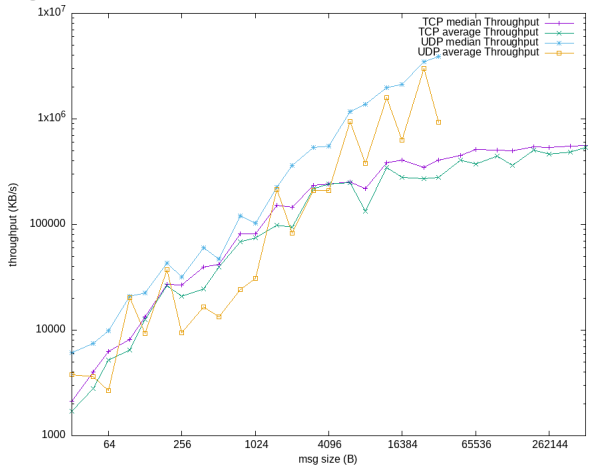
```
./udp_ping 127.0.0.1 1491 100 5
```

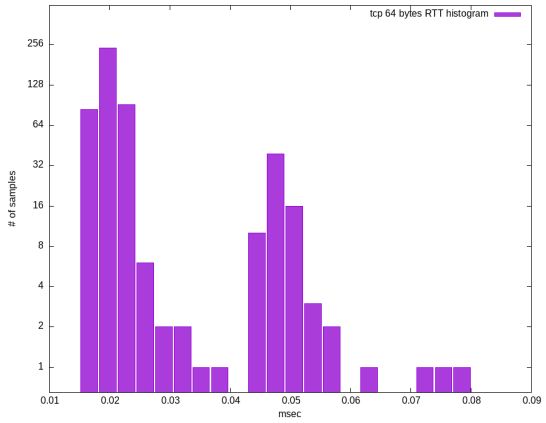
Il server rimarrà in ascolto sulla porta indicata, accetterà le connessioni dei client Ping e risponderà alle loro richieste.

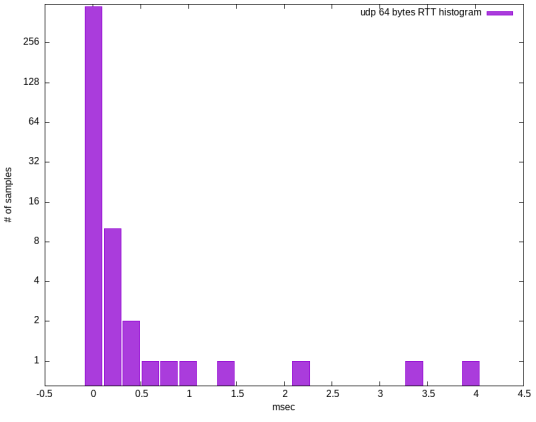
Grafici: delay in funzione dei byte ricevuti

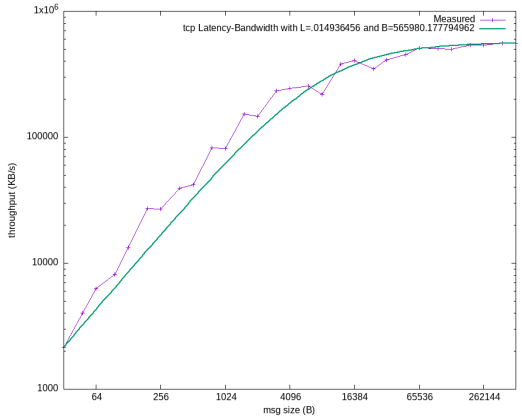
I grafici sono stati generati usando **Gnuplot**, tramite gli script bash che calcolano il delay dei pacchetti ricevuti.

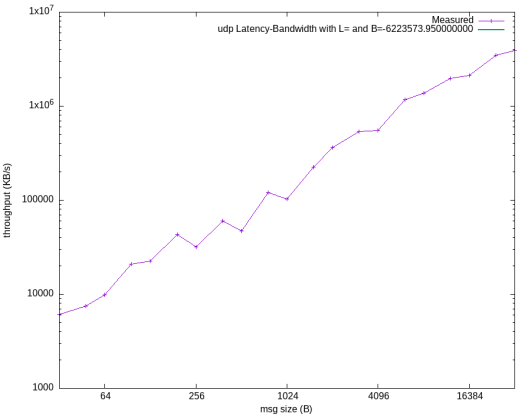
- I **grafici** mostrano i risultati su localhost:



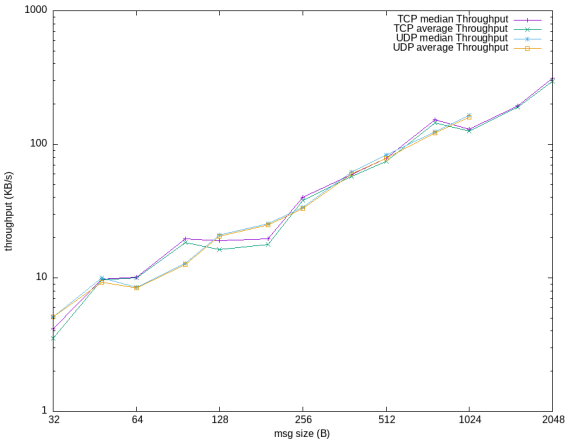


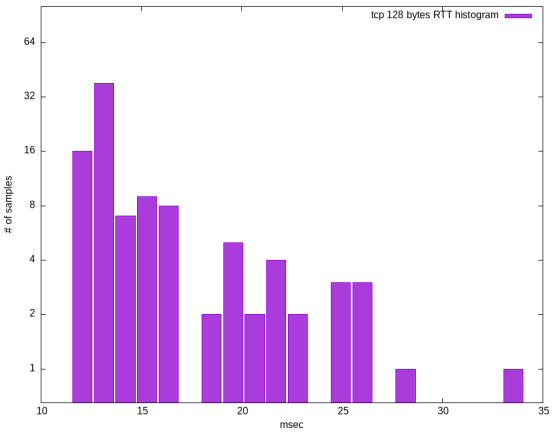


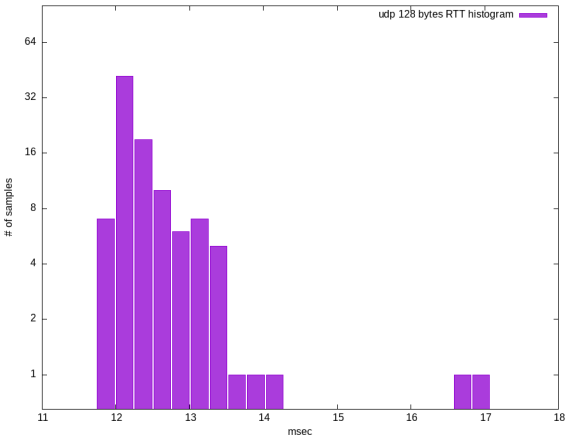


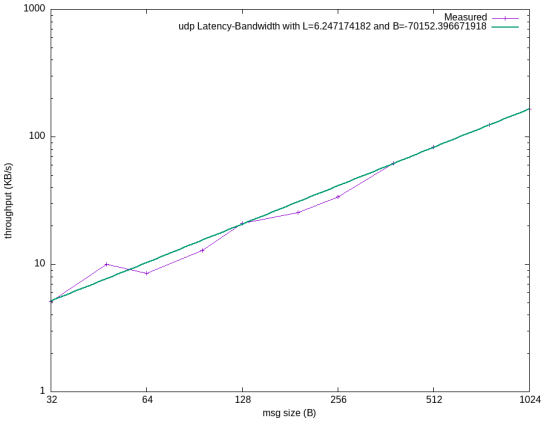


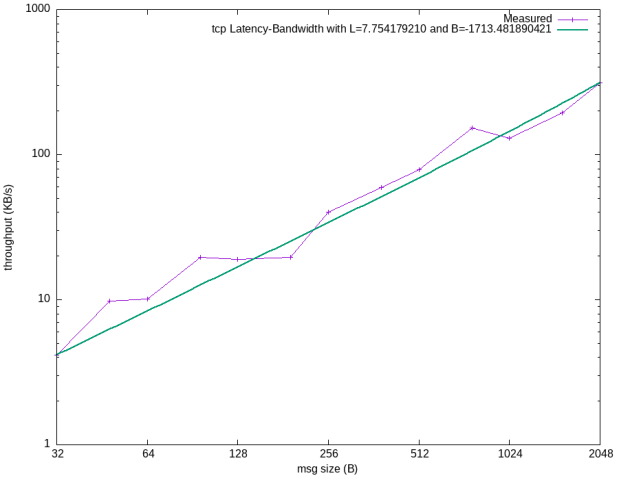
- I **grafici successivi** sono stati ottenuti pingando il server **seti.dibris.unige.it**.











Un problema che **non siamo riusciti a risolvere** riguarda l'invio di pacchetti **UDP superiori a 1024 byte**, che venivano rifiutati dal server.