```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <mpi.h>

int main(int argc, char *argv[]) {
  if (argc != 2) {
    printf("Usage : gather message_size\n");
    return 1;
  }

  int rank;
  int num_procs;
  int size = atoi(argv[1]);
  char input_buffer[size];

  MPI_Init(&argc, &argv);
  MPI_Comm_size(MPI_COMM_WORLD, &num_procs);
  MPI_Comm_rank(MPI_COMM_WORLD, &rank);
  int i;
  char recv_buffer[size*num_procs];
  srand(time(NULL));
  for (i = 0; i < size; i++)
    input_buffer[i] = rand() % 256;
  double total_time = 0.0;
  double start_time = 0.0;
  for (i = 0; i < 100; i++) {
    MPI_Barrier(MPI_COMM_WORLD);
    start_time = MPI_Wtime();

    MPI_Send(input_buffer,size,MPI_CHAR,0,99,MPI_COMM_WORLD);
    if (rank == 0) {
        int j;
        for (j=0; j < num_procs; j++) {
          MPI_Recv(recv_buffer +
j*size,size,MPI_CHAR,j,99,MPI_COMM_WORLD,MPI_STATUS_IGNORE);
        }
    }
    MPI_Barrier(MPI_COMM_WORLD);
    total_time += (MPI_Wtime() - start_time);
  }

  if (rank == 0) {
    printf("Average time for gather : %f \n", total_time/100);
  }

  MPI_Finalize();
}
```