DEMO MPI

I. Danh sách thành viên

- Nguyễn Nhất Thưởng - 20522000
- Kiều Xuân Diệu Hương -20521381
- Lê Quang Hòa - 20521331
- Nguyễn Phan Hiếu Thuận - 20521994
- Nguyễn Xuân Thịnh - 20521967

II. Demo

1. Cài các package cần thiết cho quá trình compiler trên 3 máy ảo
   - Chạy câu lệnh sudo apt-get -y install make gcc g++ openmpi-bin openmpi-common libopenmpi-devtrên 3 máy ảo

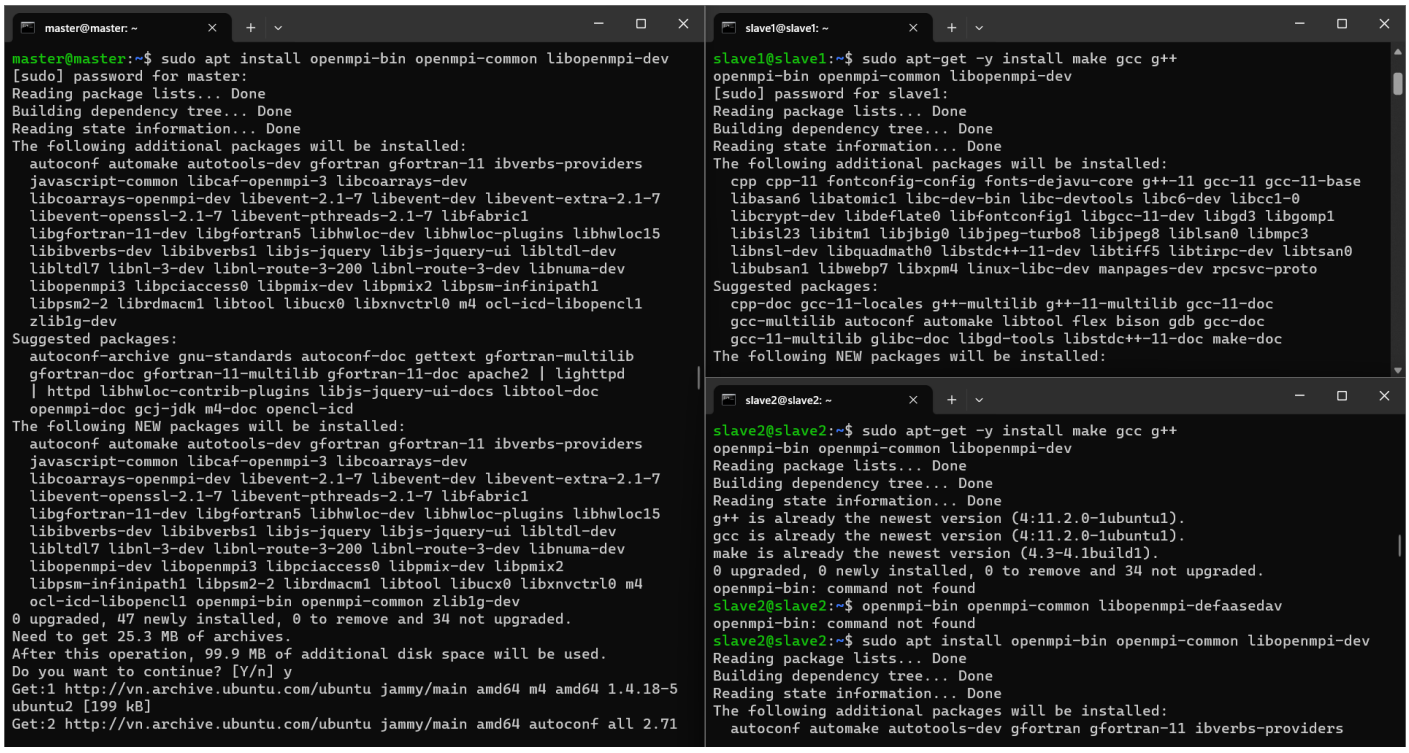

Figure 1: Cài đặt phần mềm cần thiết

2. Cấu hình ssh để máy master có thể kết nối đến các máy slave

Trên máy master

- Dùng lệnh ssh-keygen -t rsa để tạo public key (để mặc định và nhấn enter)
- Dùng lệnh cat .ssh/id_rsa.pub để lấy public key trên máy server

Trên slave:

- Dùng lệnh echo [public_key] > .ssh/authorized_keys

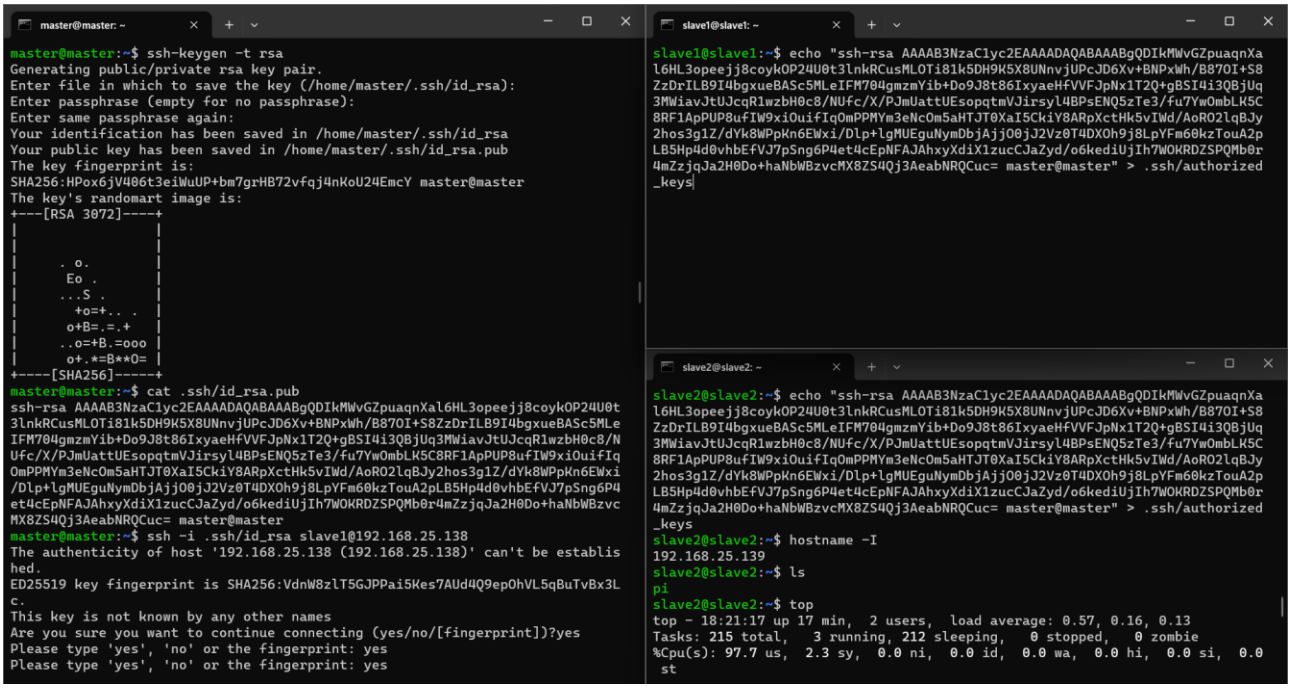

Figure 2: Màn hình termial của máy master sau khi tạo và in ra public key

3. Kiểm tra kết nối ssh từ máy master đến máy slave

   -dùng lệnh ssh -i .ssh/id_rsa [username]@[ip_address] (ex slave1@slave1) để kiểm tra kết nối

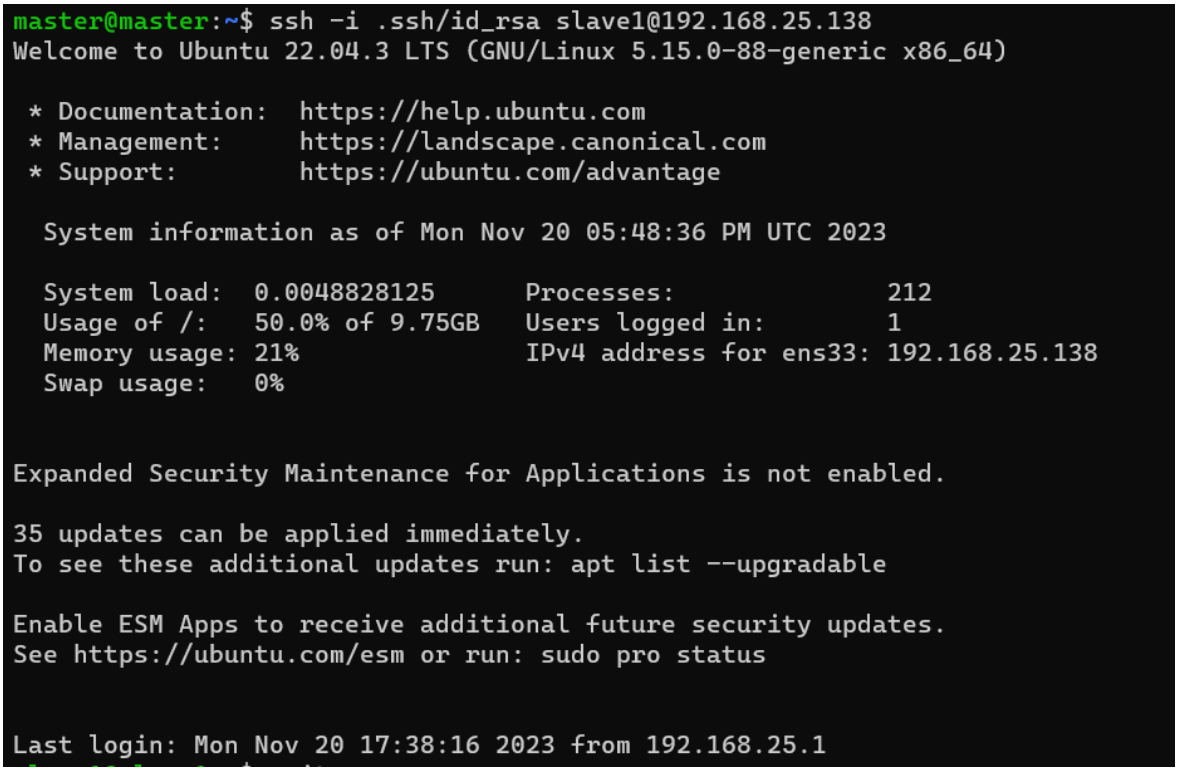

Figure 3: kết nối ssh đến máy slave thành công

4. Cấu hình file /etc/hosts

- Dùng lệnh sudo nano /etc/hosts để mở file /etc/hosts

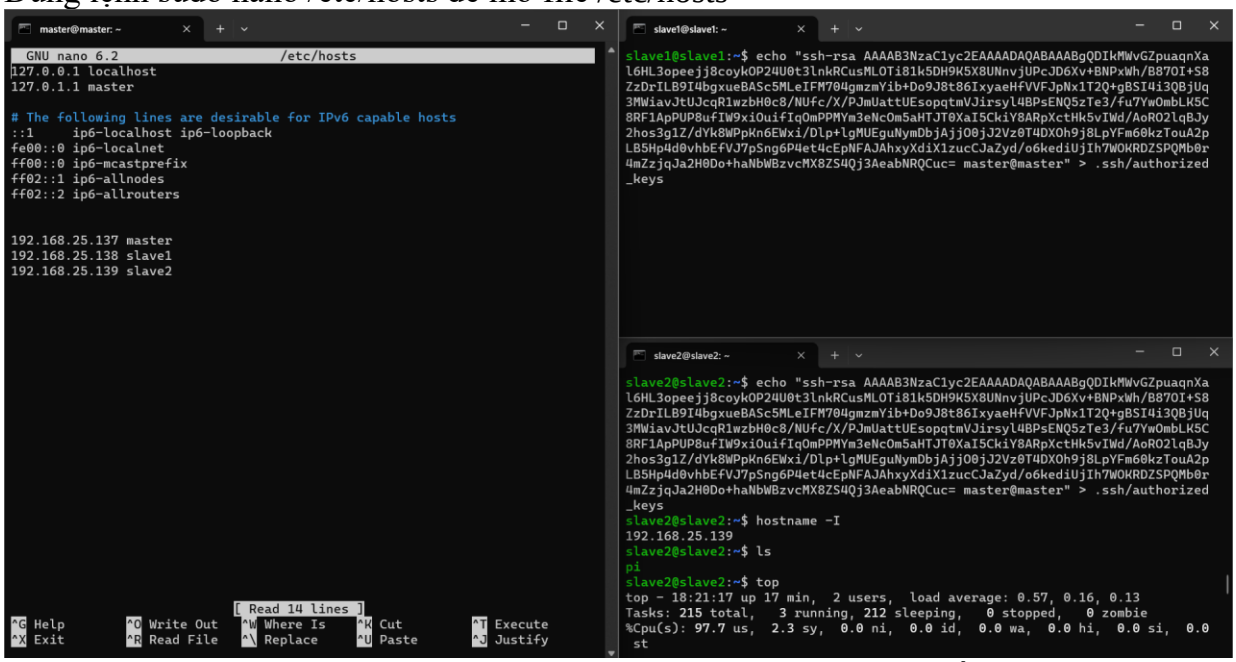

Figure 4: Nội dung trong file /etc/hosts sau khi cấu hình

5. Tạo file hosts.mpi
   - Dùng lệnh nano hosts.mpiđể tạo file hosts.mpi
   - Nội dung file sẽ bao gồm thông tin user name và ip address của các node.

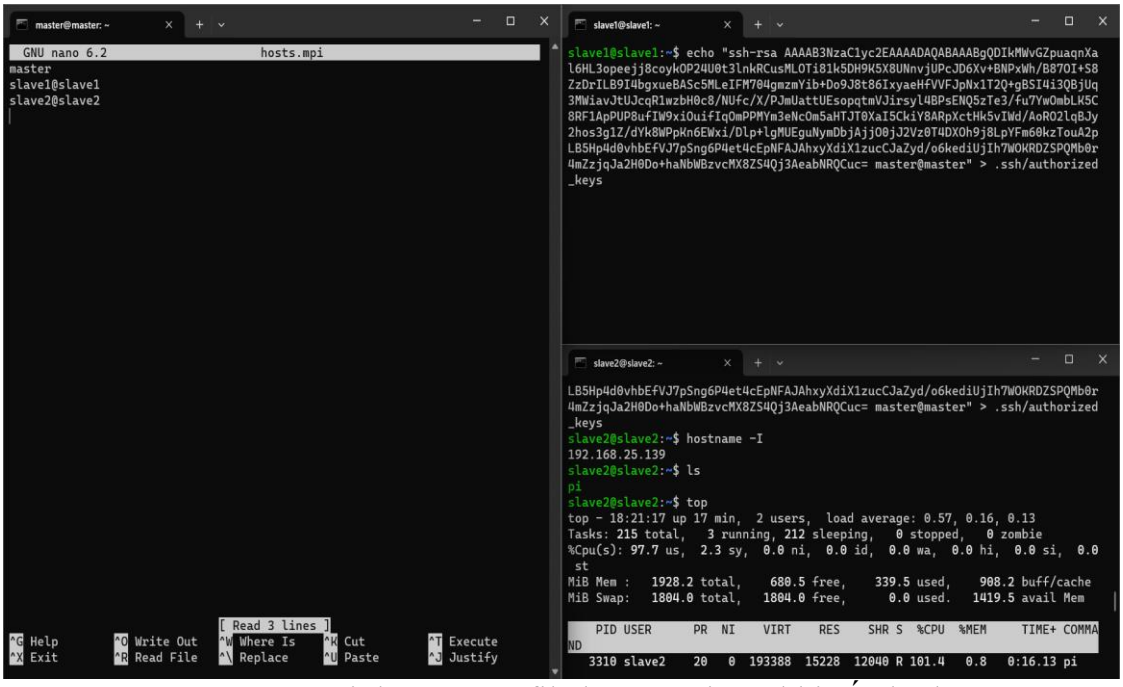

Figure 5: Nội dung trong file hosts.mpi sau khi cấu hình

6. Tạo file tên pi.c
   - Dùng lệnh nano pi.cđể tạo file pi.c

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

int main(int argc, char *argv[]) {
    int myid, numprocs;
    double PI25DT = 3.14159265358979323846264643;
    double t1, t2;

    long long npts = 1e11;
    long long i, mynpts;

    long double f, sum, mysum;
    long double xmin, xmax, x;

    // Initialization routine => starts the MPI environment
    // Defines the communicator MPI_COMM_WORLD
    MPI_Init(&argc, &argv);
    // Determines the number of processes in a communicator
    MPI_Comm_size(MPI_COMM_WORLD, &numprocs);
    // Determines the rank (id) of the calling process in the communicator
    MPI_Comm_rank(MPI_COMM_WORLD, &myid);

    // Data decomposition. Each process gets a part of the work
    mynpts = npts / numprocs;

    if (myid == 0) {
        // Returns the time in seconds since an arbitrary time in the past
        t1 = MPI_Wtime();
    }

    mysum = 0.0;
    xmin = 0.0;
    xmax = 1.0;

    // Seed the pseudo random number generator
    srand(time(0));

    for (i = 0; i < mynpts; i++) {
```

```
    for (i = 0; i < mynpts; i++) {
        x = (long double)rand() / (long double)RAND_MAX * (xmax - xmin) + xmin;
        mysum += 4.0 / (1.0 + x * x);
    }

    // Take all the processes' values of mysum and add them up into sum on process 0.
    MPI_Reduce(&mysum, &sum, 1, MPI_LONG_DOUBLE, MPI_SUM, 0, MPI_COMM_WORLD);

    if (myid == 0) {
        // Returns the time in seconds since an arbitrary time in the past
        t2 = MPI_Wtime();

        f = sum / npts;

        printf("Pi calculated with %lld points.\n", npts);
        printf("Pi calculated: %.16Lf\n", f);
        printf("Correct value of Pi: %.16f\n", PI25DT);
        printf("Error is: %.16f\n", fabsl(f - PI25DT));
        printf("Elapsed time [s] for the relevant part of the program: %f\n", t2 - t1);
    }

    // Stop the MPI environment
    MPI_Finalize();
    return 0;
}
```

7. Compiler code và gửi code từ máy master đến các máy slave
   - Dùng lệnh mpicc pi.c -o pi để complier code trên máy master
   - Dùng lệnh ls ~để xem các file và thư mục có trong thư mục home trên các máy slave

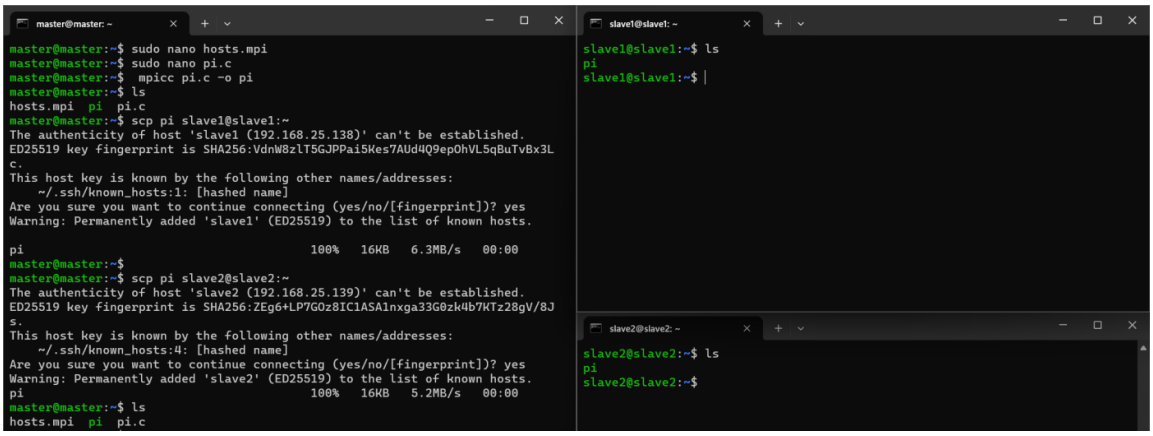

Figure 6: các file và thư mục có trong thư mục home trên các máy slave

   - Dùng lệnh scp pi slave1@slave1:~ && scp pi slave2@slave2:~ để gửi file pi đã được compiler đến thư mục home của các máy slave.

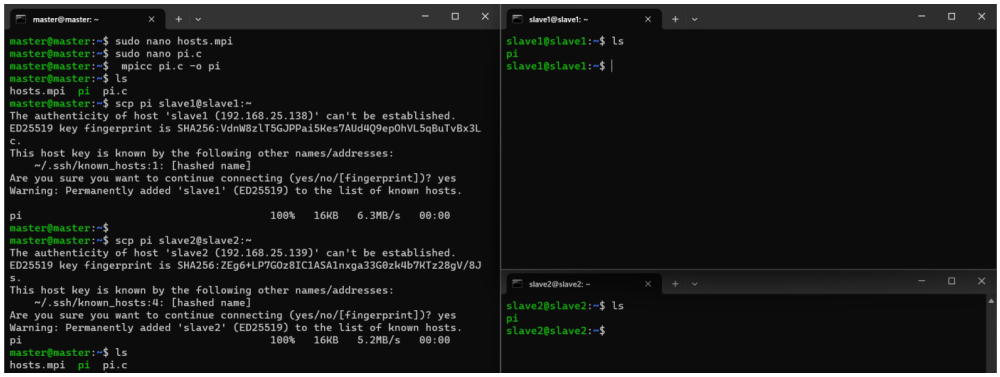

Figure 7: màn hình termial của máy master sau khi gửi file pi đến các máy slave

- Dùng lệnh ls ~ để xem nhận được file pi từ máy master chưa

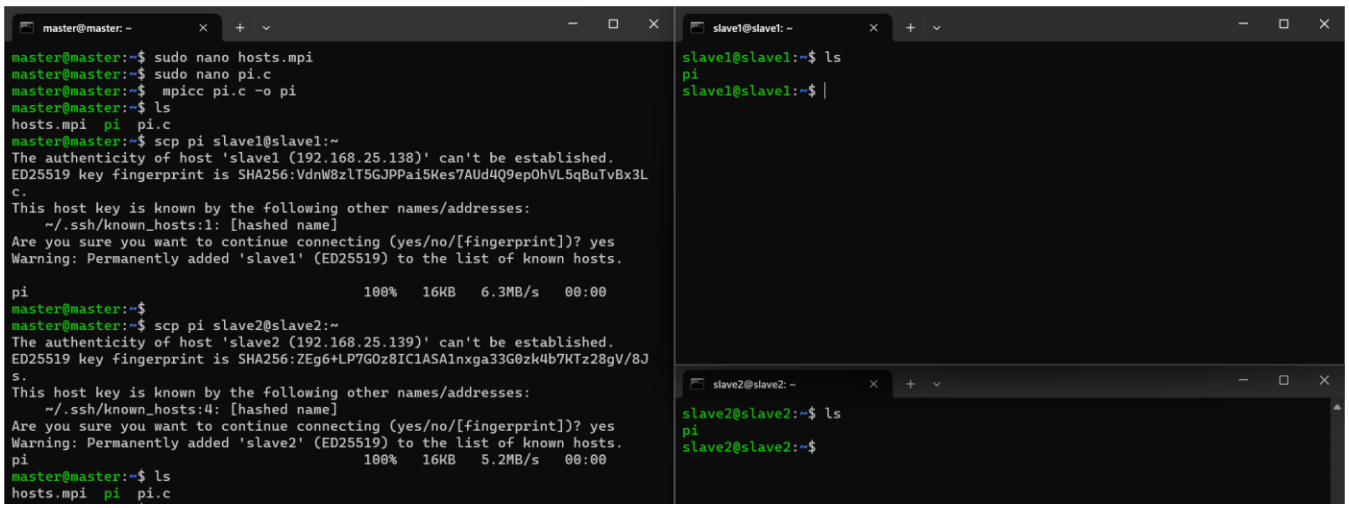

Figure 8: kiểm tra đã nhận được file pi từ máy master chưa

8. Chạy code tính số pi
   - Dùng lệnh time mpirun -np 6 --hostfile hosts.mpi piđể chạy code tính số pi trên các máy đã cấu trong file hosts.mpi
   - Dùng lệnh top trên các máy đã được cấu hình trong hosts.mpi xem nó có đang chạy code pi không.

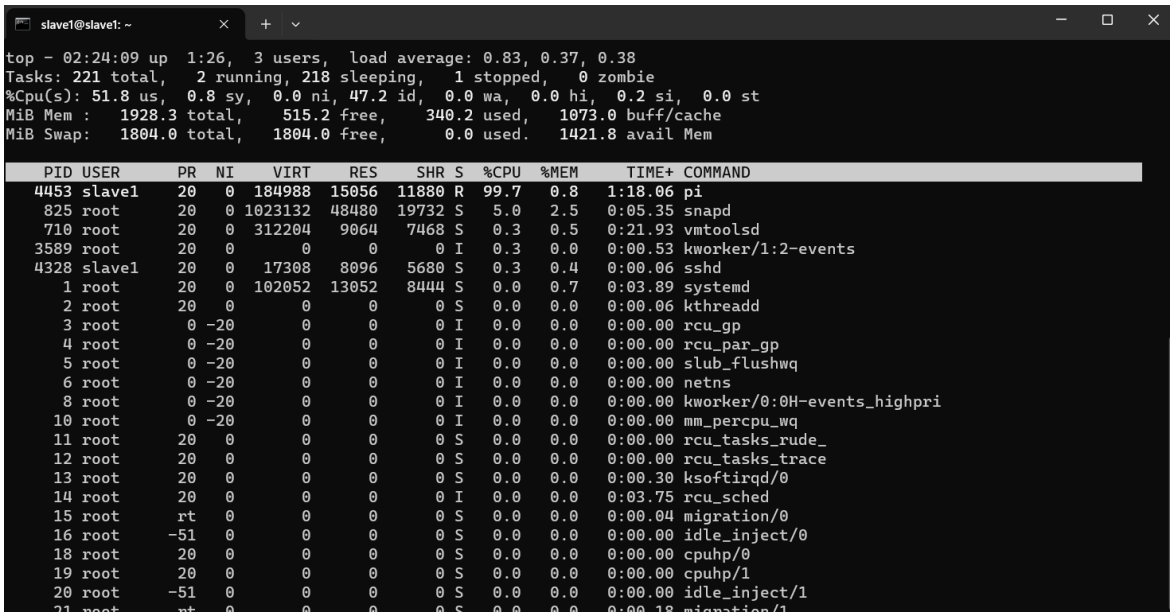

Figure 9: Kết quả khi thực hiện câu lệnh top trên máy slave

9. Kết quả sau khi tính số pi

- Thời gian tiết kiệm được khi sử dụng 2 slave node trong bài toán tính pi:

$$\frac{(2000 - 900)}{2000} \times 100 = 55\%$$

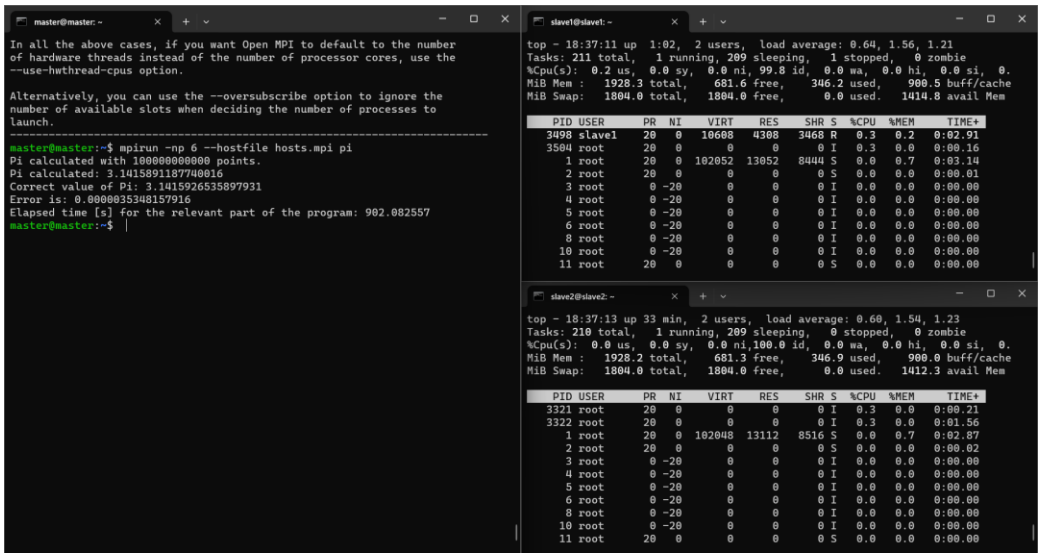

Figure 10: Kết quả sau khi tính số pi với 2 slave node

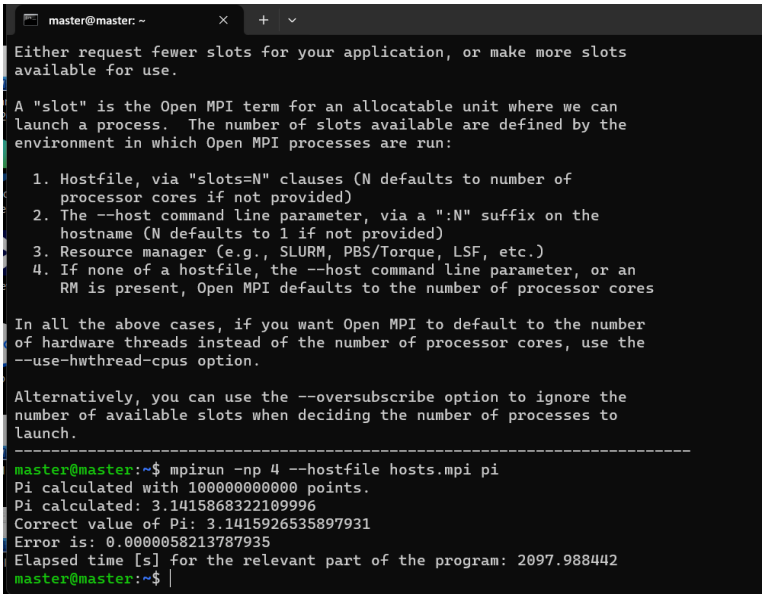

Figure 11: Kết quả sau khi tính số pi với 1 slave node

- Thời gian tiết kiệm được khi sử dụng 2 slave node trong bài toán word count:

$$\frac{(0.19 - 0.18)}{0.19} \times 100 = 5.26\%$$

Figure 12: Kết quả sau khi tính word count với 2 slave node

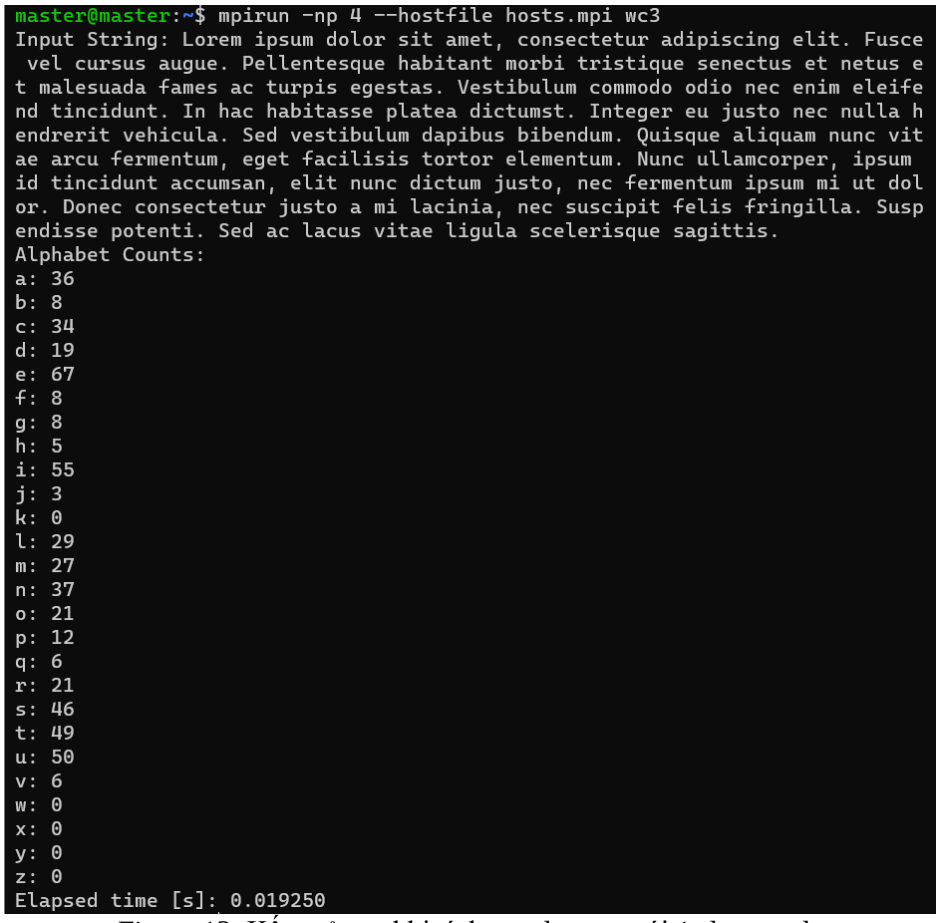

Figure 13: Kết quả sau khi tính word count với 1 slave node