Tugas Laporan
Pemrosesan Paralel
“Message Passing Interface pada Ubuntu Dekstop menggunakan Bahasa Python”

Nama : **Farca Rizqi Aqillah**

NIM : **09011282126082**

Jurusan : **Sistem Komputer**

Dosen : **Ahmad Heryanto, S.Kom, M.T.**

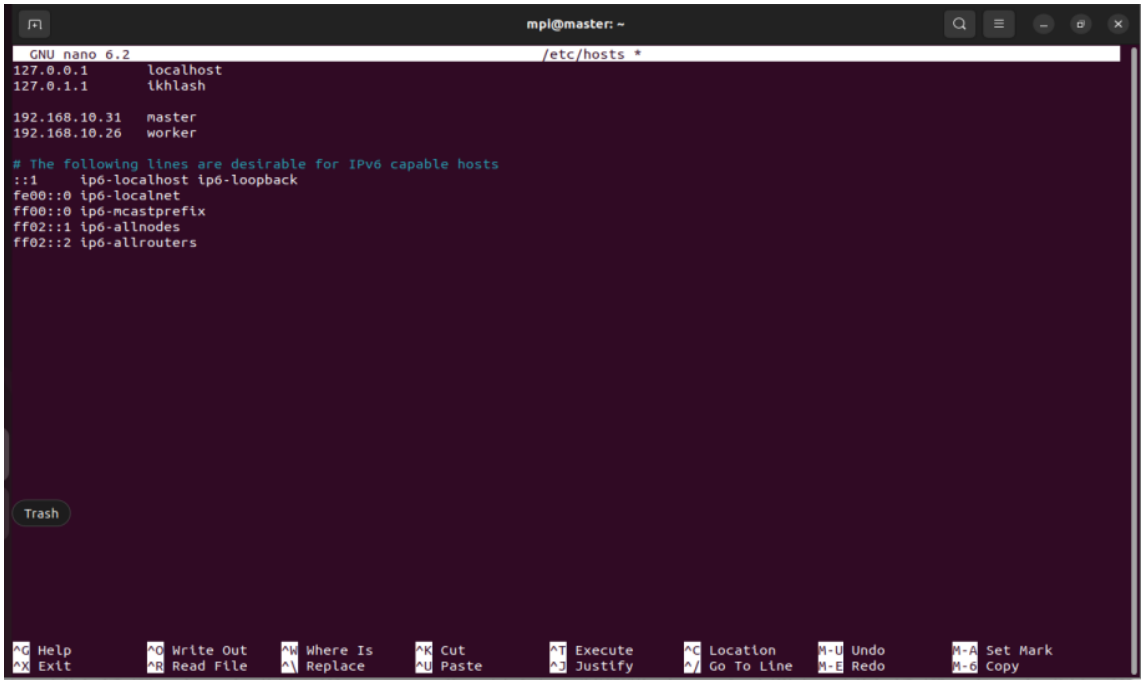
Adi Hermansyah, S.Kom., M.T.

Jurusan Sistem Komputer
Fakultas Ilmu Komputer
Universitas Sriwijaya

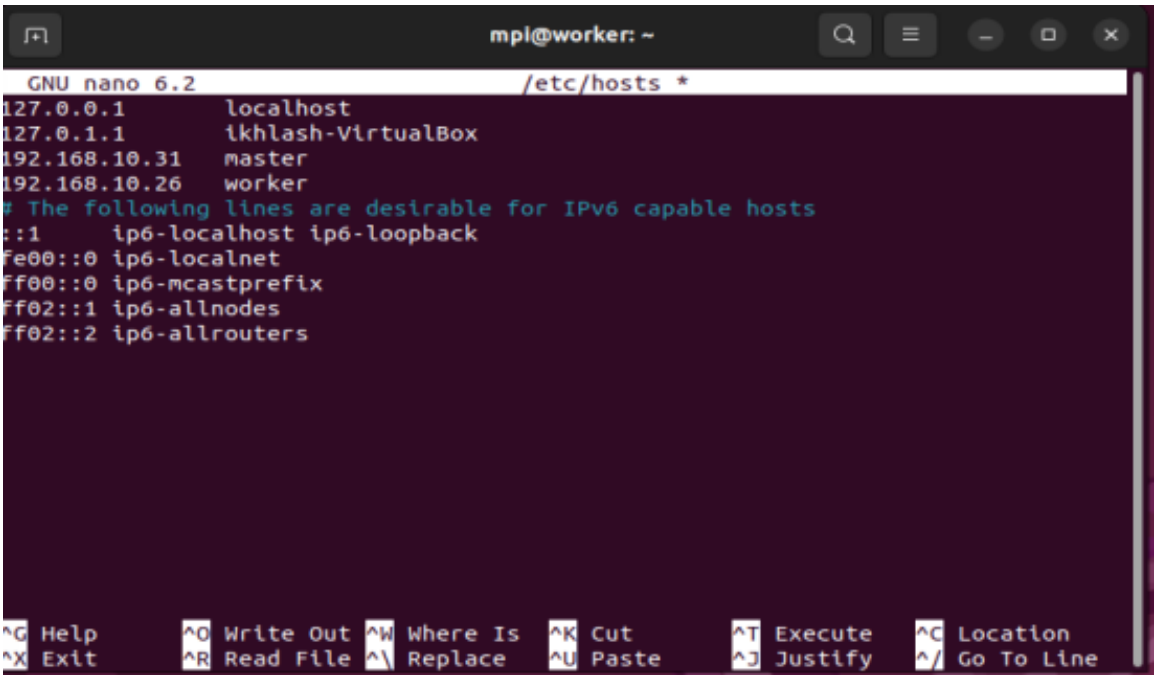
Message Passing Interface pada ubuntu dekstop menggunakan bahasa python

1. Konfigurasi file /etc/hosts

   Server

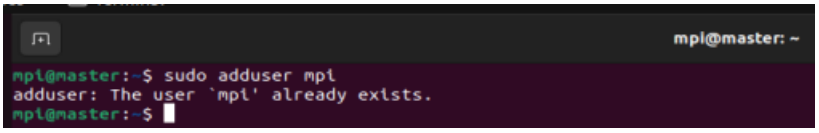


   Worker

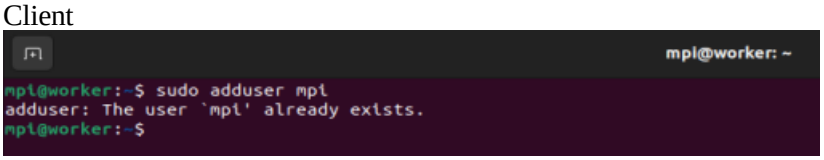


2. Menambahkan user

   Server

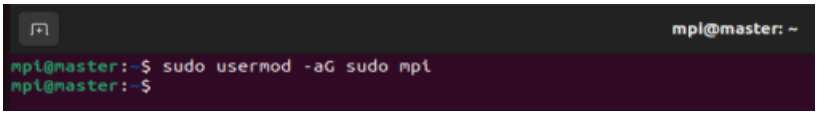


   Client

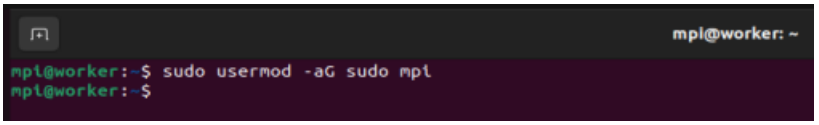


3. Memberikan akses root kepada user

   Server

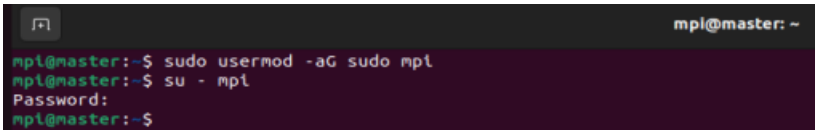


   Client

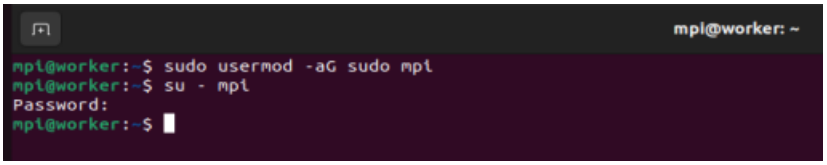


4. Melakukan login akun user

   Server

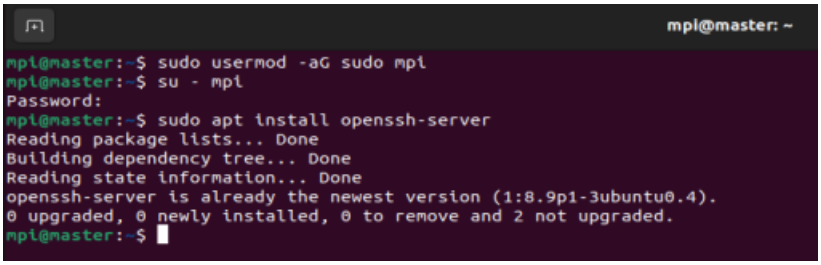


   Client



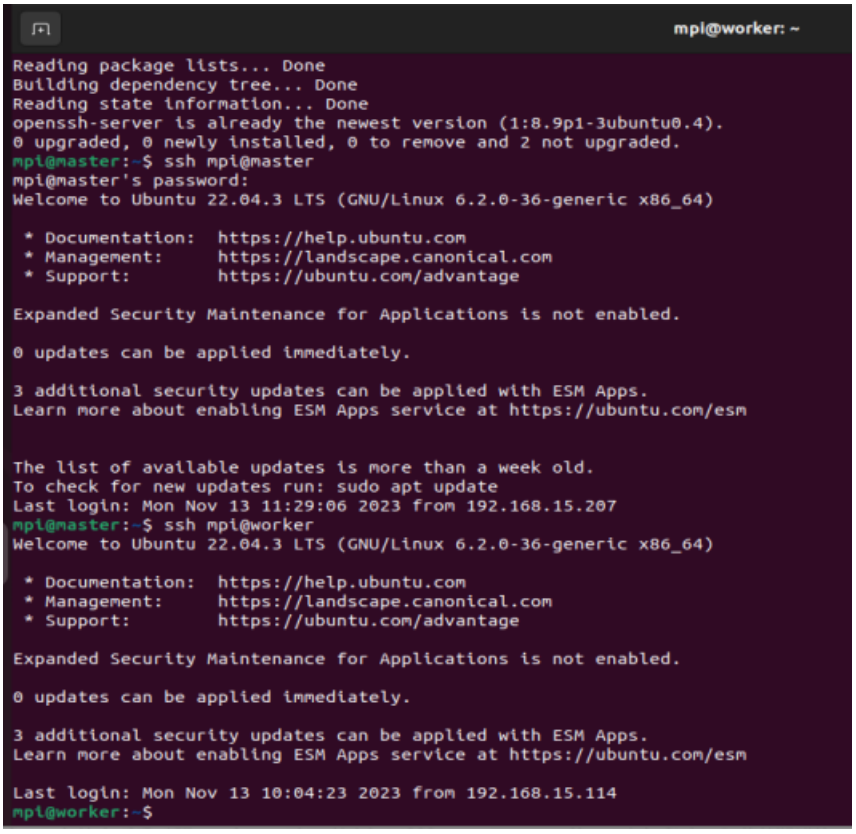
5. Instalai paket openssh-server

Server

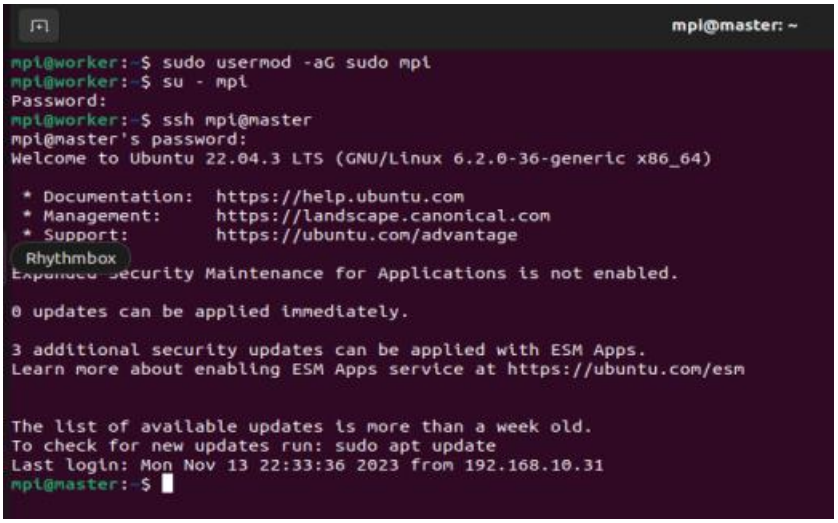


6. Pengecekan ssh

Server



Client



7. Membuat keygen

   Server

```
mpi@master:~$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/mpi/.ssh/id_rsa):
/home/mpi/.ssh/id_rsa already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/mpi/.ssh/id_rsa
Your public key has been saved in /home/mpi/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:lIt8/Nh1eq9NynUiNWdlZV/0lqiBJBkfpx/z84r8Nbc mpi@master
The key's randomart image is:
+---[RSA 3072]----+
|       ooo .   .=|
|       .+.=   ..*|
|        o+ + . .*|
|     . + .. *  o.|
|      o S  o.o+ o|
|       . + . +o+ |
|        . o o o+=|
|           . .+oB=|
|            o.o+Eo|
+----[SHA256]-----+
mpi@master:~$
```

8. Menyalin keygen ke Server dan Client

   Server

```
mpi@master: ~/.ssh
mpi@master:~$ cd .ssh
mpi@master:~/.ssh$ cat id_rsa.pub | ssh mpi@worker "mkdir .ssh; cat >> .ssh/autorized_key"
mpi@worker's password:
mkdir: cannot create directory '.ssh': File exists
mpi@master:~/.ssh$
```

9. Membuat sharing file

   Server

```
mpi@master: ~
mpi@master:~$ mkdir cloud
mkdir: cannot create directory 'cloud': File exists
mpi@master:~$ ls
cloud  Desktop  Documents  Downloads  Music  Pictures  Public  snap  Templates  Videos
mpi@master:~$
```

   Client

```
mpi@worker: ~
mpi@worker:~$ mkdir cloud
mkdir: cannot create directory 'cloud': File exists
mpi@worker:~$ ls
cloud  Desktop  Documents  Downloads  Music  Pictures  Public  snap  Templates  Videos
mpi@worker:~$
```

10. Instalasi nfs server
    Server

```
mpi@master: ~
mpi@master:~$ sudo apt install nfs-kernel-server
[sudo] password for mpi:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
nfs-kernel-server is already the newest version (1:2.6.1-1ubuntu1.2).
0 upgraded, 0 newly installed, 0 to remove and 2 not upgraded.
mpi@master:~$
```

cl

11. Konfigurasi file /etc/exports
    Server

```
mpi@master: ~
mpi@master:~$ sudo nano /etc/exports
mpi@master:~$
```

```
mpi@master: ~
  GNU nano 6.2                                    /etc/exports
# /etc/exports: the access control list for filesystems which may be exported
#               to NFS clients.  See exports(5).
#
# Example for NFSv2 and NFSv3:
# /srv/homes       hostname1(rw,sync,no_subtree_check) hostname2(ro,sync,no_subtree_check)
#
# Example for NFSv4:
# /srv/nfs4        gss/krb5i(rw,sync,fsid=0,crossmnt,no_subtree_check)
# /srv/nfs4/homes  gss/krb5i(rw,sync,no_subtree_check)
#
/home/mpi/cloud *(rw,sync,no_root_squash,no_subtree_check)
```

12. Simpan dan restart nfs-kernel-server
    Server

```
mpi@master: ~
mpi@master:~$ sudo nano /etc/exports
mpi@master:~$ sudo exportfs -a
mpi@master:~$ sudo systemctl restart nfs-kernel-server
mpi@master:~$
```

13. Instalasi nfs client
    Client

```
mpi@worker: ~
mpi@worker:~$ sudo apt install nfs-common
[sudo] password for mpi:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
nfs-common is already the newest version (1:2.6.1-1ubuntu1.2).
0 upgraded, 0 newly installed, 0 to remove and 2 not upgraded.
mpi@worker:~$
```

14. Mounting sharing file pada Client

    Client

```
mpi@worker: ~
mpi@worker:~$ sudo mount master:/home/mpi/cloud /home/mpi/cloud
mpi@worker:~$
```

15. Instalasi python3 dan mpi

    Server

```
mpi@master: ~
mpi@master:~$ sudo apt install openmpi-bin libopenmpi-dev
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
libopenmpi-dev is already the newest version (4.1.2-2ubuntu1).
openmpi-bin is already the newest version (4.1.2-2ubuntu1).
0 upgraded, 0 newly installed, 0 to remove and 2 not upgraded.
mpi@master:~$
```

    Client

```
mpi@worker: ~
mpi@worker:~$ sudo apt install openmpi-bin libopenmpi-dev
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
libopenmpi-dev is already the newest version (4.1.2-2ubuntu1).
openmpi-bin is already the newest version (4.1.2-2ubuntu1).
0 upgraded, 0 newly installed, 0 to remove and 2 not upgraded.
mpi@worker:~$
```

16. Menjalankan program bubble sort pada file bubble.py secara multi computing

    Server

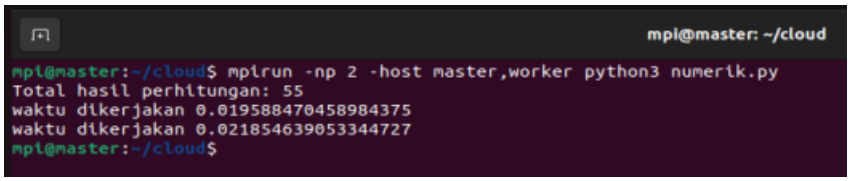
```
mpi@master: ~/cloud
mpi@master:~/cloud$ mpirun -np 2 -host master,worker python3 bubblesort.py
Waktu dikerjakan 0.017549991607666016
Sorted Data: [1, 1, 2, 2, 5, 5, 5, 5, 6, 6, 9, 9]
Waktu dikerjakan 0.01981496810913086
mpi@master:~/cloud$
```

    Kodingan :

```
from mpi4py import MPI

def bubble_sort_parallel(data):
    comm = MPI.COMM_WORLD
    rank = comm.Get_rank()
    size = comm.Get_size()

    local_data = data[rank::size]
    local_data.sort()

    for step in range(1, size):
        if rank % 2 == 0:
            if rank < size - 1:
                comm.send(local_data, dest=rank+1)
                received_data = comm.recv(source=rank+1)
                local_data = merge(local_data, received_data)
        else:
            comm.send(local_data, dest=rank-1)
            received_data = comm.recv(source=rank-1)
            local_data = merge(local_data, received_data)

    sorted_data = comm.gather(local_data, root=0)
    if rank == 0:
        sorted_data = merge_sorted_arrays(sorted_data)
```

```
        return sorted_data
    else:
        return None

def merge(arr1, arr2):
    merged_array = []
    i = j = 0
    while i < len(arr1) and j < len(arr2):
        if arr1[i] < arr2[j]:
            merged_array.append(arr1[i])
            i += 1
        else:
            merged_array.append(arr2[j])
            j += 1
    merged_array.extend(arr1[i:])
    merged_array.extend(arr2[j:])
    return merged_array

def merge_sorted_arrays(arrays):
    merged_array = []
    for array in arrays:
        merged_array = merge(merged_array, array)
    return merged_array

if _name_ == "_main_":
    data = [5, 2, 9, 1, 5, 6]
    comm = MPI.COMM_WORLD
    rank = comm.Get_rank()

    if rank == 0:
        sorted_data = bubble_sort_parallel(data)
        print("Sorted Data:", sorted_data)
    else:
        bubble_sort_parallel(data)
```

17. Menjalankan program numeric pada file numerik.py secara multi computing Server



Kodingan :

```
from mpi4py import MPI

import time


start = time.time()


def main():

    comm = MPI.COMM_WORLD

    rank = comm.Get_rank()

    size = comm.Get_size()
```

```
    # Data yang akan dihitung
    data = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

    # Bagi data di antara proses
    chunk_size = len(data) // size
    start = rank * chunk_size
    end = (rank + 1) * chunk_size

    if rank == size - 1:
        # Pastikan semua data terhitung jika
panjang data tidak habis dibagi oleh jumlah
proses
        end = len(data)

    local_sum = sum(data[start:end])

    # Kumpulkan hasil dari semua proses
    total_sum = comm.reduce(local_sum,
op=MPI.SUM, root=0)

    if rank == 0:
        print("Total hasil perhitungan:",
total_sum)

if _name_ == '_main_':
    main()
end = time.time()
print("waktu dikerjakan", end-start)
```