LAPORAN PRAKTIKUM
MESSAGE PASSING INTERFACE (MPI)

Disusun Oleh:

MUHTADIN

09011182227102

DOSEN PENGAMPU:

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

PROGRAM STUDI SISTEM KOMPUTER

FAKULTAS ILMU KOMPUTER

UNIVERSITAS SRIWIJAYA

2023

LAPORAN PRATIKUM

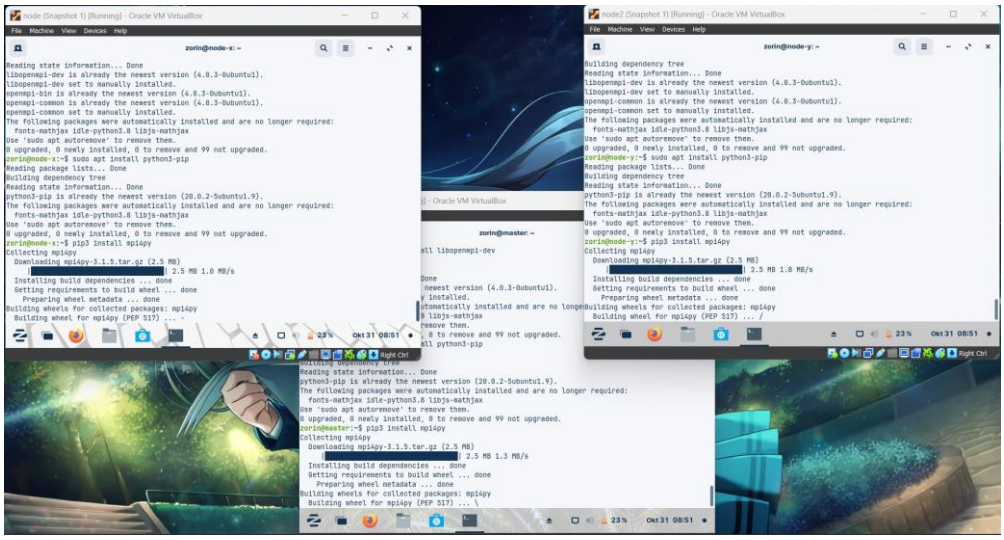
Dalam laporan praktikum ini, kami akan menjelaskan cara instalasi MPI, menghubungkan VM melalui jaringan host only dan bridged adapter via ssh, dan menjalankan program python bubble sort di MPI secara paralel via layanan ssh localhost dari openssh-server.

1. Set up MPI

A. MPI via Host Only Adapter

Dalam kebanyakan kasus, lebih dari satu virtual machine (VM) yang aktif terletak di satu komputer dalam jaringan adapter host only. Kelebihan adapter host only adalah dapat disesuaikan dengan berbagai situasi, bahkan ketika komputer utama tidak aktif atau mati. Kami akan menggunakan ZorinOS sebagai VM utama yang telah digandakan dua kali sebagai wadah praktikum. Cara menginstal dan menjalankan pemrosesan paralel MPI dengan adapter hanya host adalah sebagai berikut:

- Ubah hostname pada ketiga VM dan update sistem. Install MPI dengan mengetik perintah “sudo apt install openmpi-bin openmpi-common libopenmpi-dev”. Install juga library python untuk MPI dengan “pip3 install mpi4py.”



Gambar 1. 1 Install library python untuk mpi

- Jika telah terinstall, cek MPI dengan “mpirun --version”.

```
zorin@master:~/Desktop$ mpirun --version
mpirun (Open MPI) 4.0.3

Report bugs to http://www.open-mpi.org/community/help/
```

Gambar 1. 2 cek version

- Pastikan untuk mengecek versi mpi4py dengan perintah python dibawah ini.

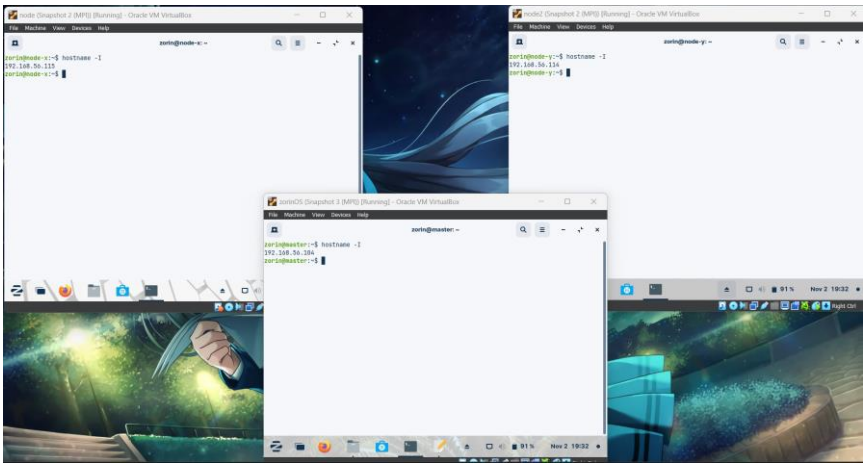
```
zorin@master:~/Desktop$ python3
Python 3.8.10 (default, May 26 2023, 14:05:08)
[GCC 9.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import mpi4py
>>> print (mpi4py.__version__)
3.1.5
```

Gambar 1. 3 melihat version

- Untuk mengetahui apakah mpi berjalan, ketik perintah benchmark seperti "mpiexec -n <jumlah core proci> python3 -m mpi4py.bench helloworld.".

```
zorin@master:~/Desktop$ mpiexec -n 1 python3 -m mpi4py.bench helloworld
Hello, World! I am process 0 of 1 on master.
```

- Install semua mpi dan openssh-server pada semua node yang ingin terhubung. Tentukkan 1 komputer yang berperan sebagai master. Perlihatkan semua ip koneksi menggunakan perintah "hostname -I".



Gambar 1. 4 cek hostname

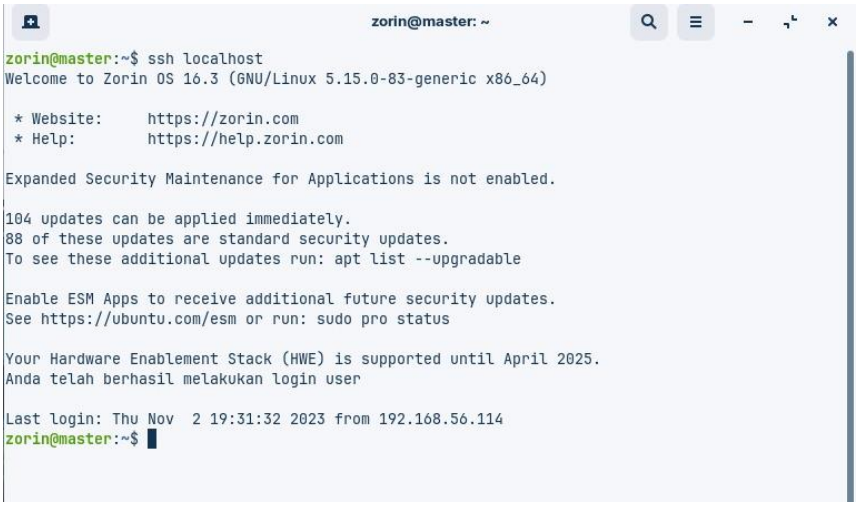
- Masuk dan tambahkan ip pada masing-masing VM dengan "sudo nano /etc/hosts".

Gambar 1. 5 Menambah IP pada file sistem hosts di slave_1

Gambar 1. 6 Menambah IP pada file sistem hosts di slave_2

Gambar 1. 7 Menambah IP pada file sistem hosts di master

- Simpan dan tutup semua file “/etc/hosts”. Kembali ke VM master, aktifkan ssh dan buka “ssh localhost”.



Gambar 1. 8 hasil perintah ssh localhost

- Ketik “ssh-keygen -t rsa” untuk mencetak ssh. Setelah itu, ketik perintah “ssh <user>@<hostname>” kepada semua VM. Input password dan tunggu hingga pesan konfirmasi telah muncul di terminal.

```
zorin@master:~$ ssh-copy-id zorin@master
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/zorin/.ssh/id_rsa.pub"
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that a
re already installed

/usr/bin/ssh-copy-id: WARNING: All keys were skipped because they already exist on the remote
system.
                (if you think this is a mistake, you may want to use -f option)

zorin@master:~$ ssh-copy-id zorin@node-x
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/zorin/.ssh/id_rsa.pub"
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that a
re already installed

/usr/bin/ssh-copy-id: WARNING: All keys were skipped because they already exist on the remote
system.
                (if you think this is a mistake, you may want to use -f option)

zorin@master:~$ ssh-copy-id zorin@node-y
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/zorin/.ssh/id_rsa.pub"
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that a
re already installed

/usr/bin/ssh-copy-id: WARNING: All keys were skipped because they already exist on the remote
system.
                (if you think this is a mistake, you may want to use -f option)
```

Gambar 1. 9 hasil percobaan

- Lakukan pengujian sekali lagi pada terminal di master, uji coba apakah mpi berjalan di multi-computer dengan memasukkan perintah “mpirun -n <jumlah_komputer> -host <nama_host1>,<nama_host2>,<nama_host_lainnya> python3 -m mpi4py.bench helloworld”. Proses dapat dikatakan berhasil berjalan secara paralel apabila muncul pesan output seperti ini.

```
zorin@master:~$ mpirun -n 3 -host master,node-x,node-y python3 -m mpi4py.bench helloworld
Hello, World! I am process 0 of 3 on master.
Hello, World! I am process 1 of 3 on node-x.
Hello, World! I am process 2 of 3 on node-y.
```

Gambar 1. 10 menjalankan 3 node

B. MPI via Bridged Adapter

Apabila ingin banyak virtual machine (VM) yang terhubung pada beberapa komputer, adapter bridged dapat menjadi opsi yang lebih baik. Pastikan komputer yang ingin dihubungkan memiliki koneksi jaringan yang sama, baik kabel maupun nirkabel, agar adapter bridged dapat digunakan. Kali ini, kami menggunakan server Ubuntu yang diremote oleh Command Prompt (CMD). Cara berikut untuk menyiapkan MPI dengan adapter bridged:

- Update sistem terlebih dahulu, perlihatkan hostname pada komputer yang ingin terhubung.

```
ubuntu@ubuntu:~$ hostname -I
10.8.142.80
```

Gambar 1. 11 cek ip pada machine

- Ketik “sudo nano /etc/hosts” untuk dapat mengedit jaringan host pada sistem yang terhubung di semua VM.

```
ubuntu@ubuntu:~$ hostname -I
10.8.142.80
ubuntu@ubuntu:~$ sudo nano /etc/hosts
[sudo] password for ubuntu:
```

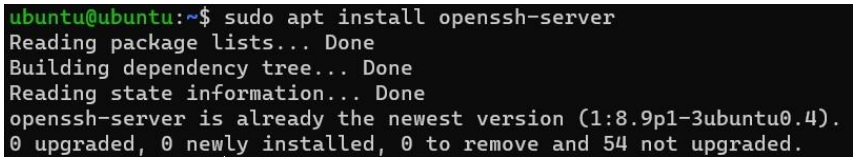
Gambar 1. 12 Perintah untuk mengedit file sistem host

```
  GNU nano 6.2                    /etc/hosts *
127.0.0.1 localhost
127.0.1.1 ubuntu

10.8.142.80 ubuntu
10.8.143.159 muhtadinserver
```

Gambar 1. 13 Menambahkan jaringan yang ingin disambungkan pada sistem

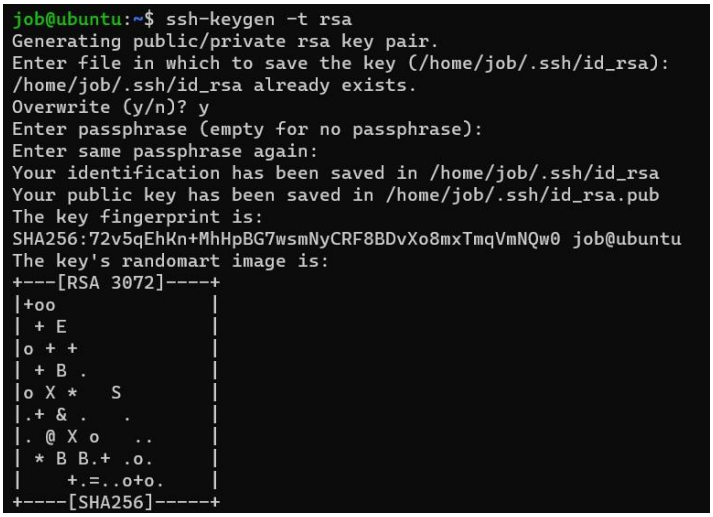
- Simpan dan tutup isi “/etc/hosts”. Install openssh-server untuk semua VM agar dapat dihubungkan. Setelah openssh-server diinstall, buat dan login sebagai user yang baru di semua VM. Untuk membuat user ketik “sudo adduser <nama_user>”, memberikan akses root “sudo usermod -aG sudo <nama_user>”, dan menggunakan user “su - <nama user>”.



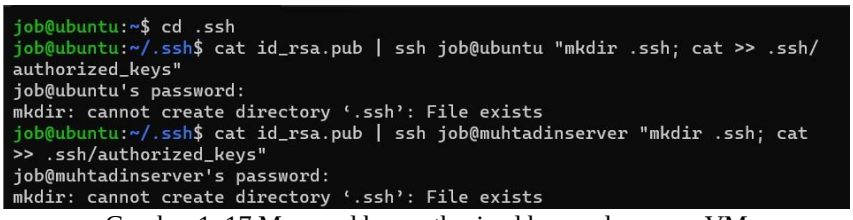
Gambar 1. 14 Menginstall openssh-server

Gambar 1. 15 Masuk sebagai user yang baru

- Ketik “ssh-keygen -t rsa” pada 1 VM master untuk dapat menghubungkannya dengan VM lain. Ganti direktori menjadi “/home/<nama_user>/.ssh”. dan hubungkan ssh mereka dengan mengetik “cat id_rsa.pub | ssh <nama_user>@<host> “mkdir .ssh; cat >> .ssh/authorzed_keys””. Jika hasil output setelah memasukkan password “… file exists”, artinya file telah dibuat sebelumnya.



Gambar 1. 16 Mengetik perintah kunci untuk ssh



Gambar 1. 17 Memasukkan authorized key pada semua VM

- Install MPI “sudo apt install openmpi-bin libopenmpi-dev” dan library python “pip3 install mpi4py” pada VM yang telah terhubung.

```
job@ubuntu:~$ 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 54 not upgraded.
job@ubuntu:~$ sudo apt install python3-pip
pip install mpi4py
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
python3-pip is already the newest version (22.0.2+dfsg-1ubuntu0.3).
0 upgraded, 0 newly installed, 0 to remove and 54 not upgraded.
Defaulting to user installation because normal site-packages is not writeabl
e
Requirement already satisfied: mpi4py in ./.local/lib/python3.10/site-packag
es (3.1.5)
```

Gambar 1. 18 Menginstal python3-pip

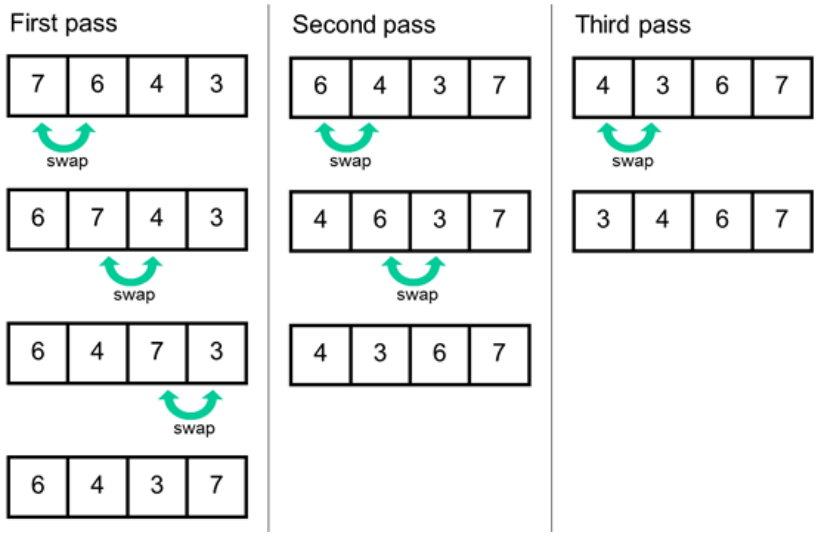
- Uji MPI setelah penginstalan berhasil dilakukan. Normalnya ia akan mengeluarkan output seperti dibawah ini.

```
job@ubuntu:~$ mpirun -n 2 -host ubuntu,muhtadinserver python3 -m mpi4py.benc
h helloworld
Hello, World! I am process 0 of 2 on ubuntu.
Hello, World! I am process 1 of 2 on muhtadinserver.
```

Gambar 1. 19 Menjalankan 2 node

2. Menjalankan program python bubble sort dengan MPI

Untuk menjalankan program MPI, dua program akan digunakan. Mereka dibuat menggunakan bahasa pemrograman python, dan konsep pengurutan bubble adalah kemampuan untuk mengurutkan data dengan menukar posisi dua nilai yang bersebelahan jika urutan data tidak sesuai.



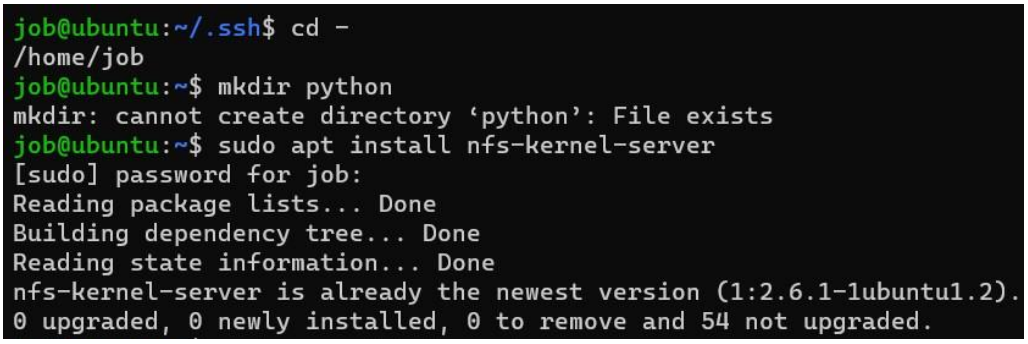
Gambar 2. 1 Proses eksekusi dari bubble sort

Sementara program pertama diuji pada dua kondisi jaringan, program kedua hanya diuji pada satu kondisi jaringan. Perlu diingat bahwa kedua program harus berada di lokasi yang sama. Jika Anda menggunakan adapter hanya host, kesamaan ini

mungkin dapat diakali dengan mudah. Namun, jika banyak virtual machine (VM) terhubung dalam satu jaringan, hal itu berbeda dengan adapter terhubung. Jadi, sebelum program python dijalankan, disarankan untuk membuat direktori yang dapat digunakan pada semua virtual machine (VM).

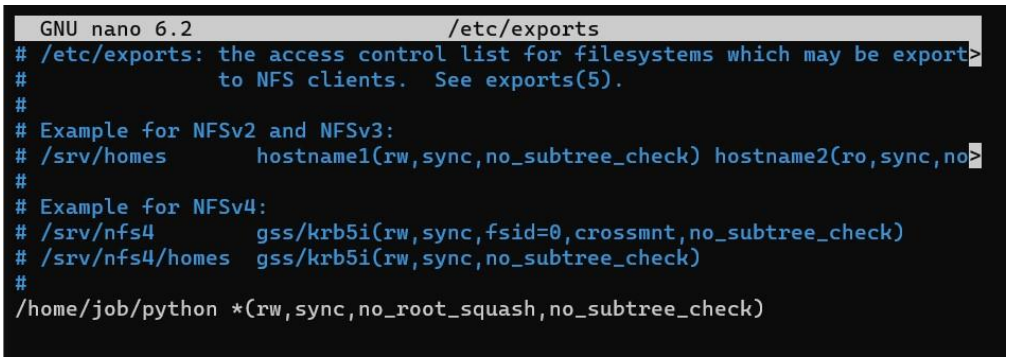
Ini berlaku jika menggunakan adapter hanya host. Namun, situasi berbeda dengan adapter bridged jika banyak virtual machine (VM) terhubung dalam satu jaringan. Oleh karena itu, sebelum menjalankan program python, disarankan untuk membuat direktori yang dapat digunakan pada semua VM.

Salah satu caranya adalah dengan menginstall NFS. Pertama, buat dulu direktori sebagai direktori bersama. Kemudian install nfs-kernel-server pada semua VM yang tersambung.



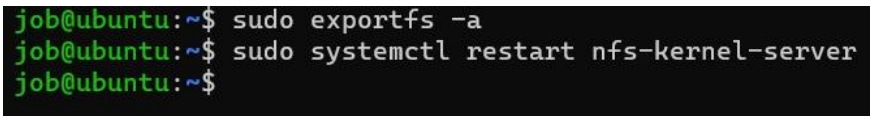
Gambar 2. 2 Membuat folder python

Ketik “sudo nano /etc/exports” dan berikan akses pada direktori yang telah dibuat dengan <lokasi_direktori> *(rw,sync,no_root_squash,no_subtree_check).



Gambar 2. 3 Lokasi direktori

Restart service NFS agar dapat digunakan.



Gambar 2. 4 Restart service NFS

Pada VM yang terhubung selain master, ketik “sudo mount <nama_host_master>:<direktori_master_yang _dimounting> <direktori_yang _jadi_wadah_mounting>”. Cek area mounting dengan “df -h”. jika muncul direktori master yang termounting, artinya perintah “mount” berhasil dieksekusi. Artinya setiap ada program baik dari master maupun node yang dimasukkan dalam direktori, orang lain dapat mengaksesnya melalui direktori mounting.

```
ubuntu:/home/job/python          9.8G  5.1G  4.2G  55% /home/job/python
```

Gambar 2. 5 Ukuran file

Contoh program 1. Program ini akan mensortir data inputan berbentuk list yang diacak menjadi urutan data dari yang tertinggi hingga data terendah. Jika dieksekusi secara paralel, ia akan muncul sebanyak 3 kali dengan urutan list yang sama.

```
def bubble_sort(arr):
    arr_len = len(arr)
    for i in range(arr_len-1):
        flag = 0
        for j in range(0, arr_len-i-1):
            if arr[j] > arr[j+1]:
                arr[j+1], arr[j] = arr[j], arr[j+1]
                flag = 1
                if flag == 0:
                    break
    return arr

arr = [5, 3, 4, 1, 2, 10, 83, 86, 83, 76, 64, 52, 64, 86]
print("List sorted with bubble sort in ascending order: ", bubble_sort(arr))
```

Gambar 2. 6 Program python

```
zorin@master:~$ mpirun -n 3 -host master,node-x,node-y python3 /home/python/bubble_up.py
List sorted with bubble sort in ascending order:  [1, 2, 3, 4, 5, 10, 52, 64, 64, 76, 83, 83, 86,
86]
List sorted with bubble sort in ascending order:  [1, 2, 3, 4, 5, 10, 52, 64, 64, 76, 83, 83, 86,
86]
List sorted with bubble sort in ascending order:  [1, 2, 3, 4, 5, 10, 52, 64, 64, 76, 83, 83, 86,
86]
```

Gambar 2. 7 Output dari program python

```
def bubble_sort(arr):
    arr_len = len(arr)
    for i in range(arr_len-1):
        flag = 0
        for j in range(0, arr_len-i-1):
            if arr[j] > arr[j+1]:
                arr[j+1], arr[j] = arr[j], arr[j+1]
                flag = 1
                if flag == 0:
                    break
    return arr

arr = [5, 3, 4, 1, 2]
print("List sorted with bubble sort in ascending order: ", bubble_sort(arr))
```

Gambar 2. 8 Isi dari program 1 di Ubuntu Server

```
job@ubuntu:~$ mpirun -n 2 -host ubuntu,muhtadinserver python3 /home/job/pyth
on/cobain.py
List sorted with bubble sort in ascending order:  [1, 2, 3, 4, 5]
List sorted with bubble sort in ascending order:  [1, 2, 3, 4, 5]
```

Gambar 2. 9 Output dari python setelah di eksekusi secara paralel via bridged adapter

Contoh program 2. Program ini akan menerima inputan berupa angka. Pengguna akan menentukkan berapa banyak elemen yang akan diinput. Kemudian, pengguna akan memasukkan data tersebut hingga akhir. Contoh jika elemennya adalah 8, maka pengguna harus memasukkan inputan angka hingga 8 elemen. Begitu juga seterusnya. Elemen yang diinput akan disortir dan ditampilkan dalam bentuk list sesuai dengan urutan dari angka terendah hingga angka tertinggi.

```python
from mpi4py import MPI
import numpy as np

def parallel_bubble_sort(data, comm):
    rank = comm.Get_rank()
    size = comm.Get_size()

    local_data = np.array_split(data, size)[rank]

    n = len(local_data)
    for i in range(n):
        for j in range(0, n - i - 1):
            if local_data[j] > local_data[j + 1]:
                local_data[j], local_data[j + 1] = local_data[j + 1], local_data[j]

    sorted_data = comm.gather(local_data, root=0)

    if rank == 0:
        merged_data = np.concatenate(sorted_data)
        merged_data.sort()
        return merged_data
    else:
        return None

if __name__ == "__main__":
    comm = MPI.COMM_WORLD
    rank = comm.Get_rank()

    if rank == 0:
        try:
            num_elements = int(input("Masukkan jumlah elemen dalam array: "))
            data = []
            for i in range(num_elements):
                element = int(input(f"Masukkan elemen array bilangan {i + 1}: "))
                data.append(element)
        except ValueError:
            print("Input harus berupa angka.")
            exit(1)
```

Gambar 2. 10 Isi codingan program 2 (1)

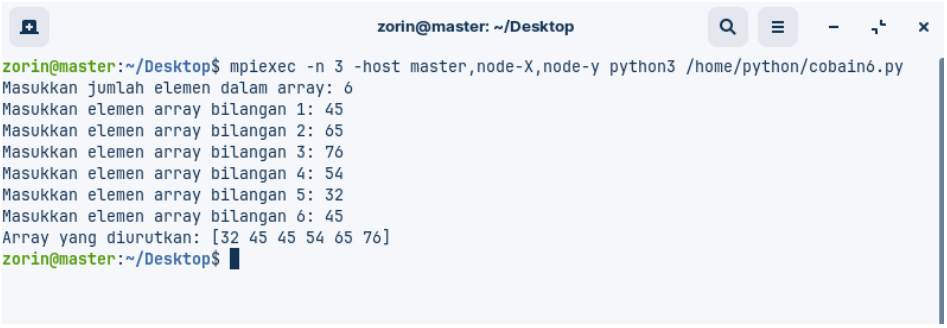
```python
    else:
        data = None

    data = comm.bcast(data, root=0)

    sorted_data = parallel_bubble_sort(data, comm)

    if rank == 0:
        print("Array yang diurutkan:", sorted_data)
```

Gambar 2. 11 Isi codingan program 2 (2)



Gambar 2. 12 Output program 2 setelah dijalankan secara paralel