

## Task 2 Report and MPI Commands Explanation

### c) Output from master and worker nodes

```
root@00b9972a4a4d:/mpi x root@42526d804c8f:/ x WORKER 3 root@7b1a6b1e21f6:/ x WORKER 1
hwest Airlines Co. 0.29
root@00b9972a4a4d:/mpi# mpiexec -n 4 -machinefile ~/
machinefile python t3q1.py
Slave Container Id: 7b1a6b1e21f6 with rank :1
Slave finished processing with : {1}
Slave Container Id: 26a772e60f8f with rank :2
Slave finished processing with : {2}
Slave Container Id: 42526d804c8f with rank :3
Slave finished processing with : {3}
Master Container Id: 42526d804c8f has distributed ch
unks to all slaves
100%| 3/3 [00:00<00:00, 41391.16it/s]
Southwest Airlines Co. 0.22
root@00b9972a4a4d:/mpi# mpiexec -n 4 -machinefile ~/
machinefile python t3q1.py
Slave Container Id: 7b1a6b1e21f6 with rank :1
Slave finished processing with : {1}
Slave Container Id: 26a772e60f8f with rank :2
Slave finished processing with : {2}
Slave Container Id: 42526d804c8f with rank :3
Slave finished processing with : {3}
Master Container has distributed chunks to all slave
s
100%| 3/3 [00:00<00:00, 37227.55it/s]
Southwest Airlines Co. 0.22
root@00b9972a4a4d:/mpi# MASTER

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
inet 127.0.0.1 netmask 255.0.0.0
loop txqueuelen 1000 (Local Loopback)
RX packets 0 bytes 0 (0.0 B)
RX errors 0 dropped 0 overruns 0 fram
e 0
TX packets 0 bytes 0 (0.0 B)
TX errors 0 dropped 0 overruns 0 carri
er 0 collisions 0
root@42526d804c8f:/# service ssh start
* Starting OpenBSD Secure Shell server sshd
[ OK ]
root@42526d804c8f:/# nano etc/host
root@42526d804c8f:/# nano /etc/hosts
root@42526d804c8f:/#

ck)
RX packets 0 bytes 0 (0.0 B)
RX errors 0 dropped 0 overruns 0
frame 0
TX packets 0 bytes 0 (0.0 B)
TX errors 0 dropped 0 overruns 0 c
arrier 0 collisions 0
root@7b1a6b1e21f6:/# service ssh start
* Starting OpenBSD Secure Shell server sshd
[ OK ]
root@7b1a6b1e21f6:/# nano etc/hosts
root@7b1a6b1e21f6:/# service ssh start
* Starting OpenBSD Secure Shell server sshd
[ OK ]
root@7b1a6b1e21f6:/#

RX packets 0 bytes 0 (0.0 B)
RX errors 0 dropped 0 overruns 0 fra
me 0
TX packets 0 bytes 0 (0.0 B)
TX errors 0 dropped 0 overruns 0 carr
ier 0 collisions 0
root@26a772e60f8f:/# service ssh start
* Starting OpenBSD Secure Shell server sshd
[ OK ]
root@26a772e60f8f:/# nano etc/hosts
root@26a772e60f8f:/# service ssh start
* Starting OpenBSD Secure Shell server sshd
[ OK ]
root@26a772e60f8f:/#
```

### d) The IP of each node are as follows:

Master Node : 172.17.0.2

Workers : 172.17.0.3, 172.17.0.4, 172.17.0.5

There are no shared paths as each container runs their own processes with the data given from the Master container and each of them are isolated from each other.

### e) Commands used for task 2 and its explanation:

#### SERVER:

1. docker pull ozxx33/mpi4py-cluster-base

-> gets the base image for the assignment in order to build containers on top of it

2. `docker run -it --mount "type=bind,src=%cd%,target=/mpi" -it --name mpi_prep`  
-> creates and starts a container with the name `mpi_prep`. The mount command is used to mount the host directory with the container's file system. It copies the file present in the current working directory into the `mpi` folder in the container.
3. `passwd`  
-> used to set the password for the container. In order to connect with the containers using passwordless ssh
4. `apt-get update`  
-> updates the container with latest packages
5. `apt-get install nano net-tools iputils-ping openssh-client openssh-server`  
-> installs necessary packages in order to connect to the nodes using ssh
6. `pip install pandas==1.5.0`  
-> installs specified pandas version
7. `docker commit T2Server mpi_cluster_t2`  
-> Creates a new intermediate base image with name `mpi_cluster_t2` from `T2Server` container
8. `docker run -it --mount "type=bind,src=%cd%,target=/mpi" -it --name T2Server mpi_cluster_t2 bash`  
-> Creates a new container with name `T2Server` from the intermediate base image
9. `ssh-keygen -t rsa`  
-> Used to store private keys for connecting with containers
10. `ssh-copy-id -i ~/.ssh/id_rsa.pub root@<container_ip>`  
-> Stores the public key to connect with all the slave container using ssh
11. `ssh root@<container_ip>`  
-> Used to connect to the specified container by its IP address for testing
12. `exit`  
-> To exit from the container shell
13. `import tqdm`  
-> Install tqdm package
14. `Nano ~/machinefile`  
-> Used to add the IP addresses of the container nodes to use as worker machines

15. `mpiexec -n 4 -machinefile ~/machinefile python t3q1.py`  
-> Run MPI program with machine file

### **WORKER 1:**

1. `docker run -it --mount "type=bind,src=%cd%,target=/mpi" -it --name T2SlaveOne mpi_cluster_t2 bash`  
-> Creates a new container using the intermediate image with the name T2SlaveOne
2. `service ssh start`  
-> Start the ssh service for master nodes to connect with worker1
3. `nano /etc/hosts`  
-> Add the IP Address of the master container for worker1 to accept connection from Master container

### **WORKER 2:**

1. `docker run -it --mount "type=bind,src=%cd%,target=/mpi" -it --name T2Slave2 mpi_cluster_t2 bash`  
-> Creates a new container using the intermediate image with the name T2Slave2
2. `service ssh start`  
-> Start the ssh service for master nodes to connect with worker2
3. `nano /etc/hosts`  
-> Add the IP Address of the master container for worker2 to accept connection from Master container

### **WORKER 3:**

1. `docker run -it --mount "type=bind,src=%cd%,target=/mpi" -it --name T2Slave3 mpi_cluster_t2 bash`  
-> Creates a new container using the intermediate image with the name T2Slave2
2. `service ssh start`  
-> Start the ssh service for master nodes to connect with worker3
3. `nano /etc/hosts`  
-> Add the IP Address of the master container for worker3 to accept connection from Master container

g) Yes, I have used an intermediate image to create the final containers.  
This image was created with all the necessary installments ready.