10/25/2015

MPI Instructions

For Ryerson EE Network

Muhammad Ismail Sheikh
DR. NAGI MEKHIEL

Mpich-3.1.4 software is already installed on Ryerson EE network and anyone using the following instructions can run this software on one computer or on the network of computers.

This document is divided in three parts,

1. MPI Setup
2. Running MPI on one computer
3. Running MPI on the network of computers

1. MPI Setup:

Since the MPI is already installed on Ryerson EE network so an individual don't have to manually install it on their account. However, they have to add some directories in their account to successfully run MPI software.

There are two ways to add directories to run MPI

1. Save the directory in a file:

    1.1 Create a new empty file in your home directory and copy the following path in that file.

    ```
    LD_LIBRARY_PATH=/usr/local/mpich-3.1.4/lib
    export   LD_LIBRARY_PATH
    ```

    1.2 Change the file name to **.myzshrc** (dot is a hidden file so after renaming the file it should disappear)
    You can verify the file through the terminal by using **ls –alt** on your home directory. Similarly, you can use **pico .myzshrc** to open to the file and verify the path in the file.

    1.3 After adding LD_LIBRARY_PATH in your home directory, you should be able to run MPI software.

2. Create directories before execution of program.
    2.1 The MPI environment can be created through the terminal command **mpichenv**

    Note: Mpichenv command creates all the required directories for you. However, after you logout all those directories are deleted. Thus, if you use this option, you must create these directories every time you login to Ryerson EE network.

3. Running MPI on one computer:

For section 2 and 3 of this document, we will be referring to the following example for MPI software. It is a simple HelloWorld example which outputs the computer process number and hostName of the computer.

```
#include "/usr/local/mpich-3.1.4/include/mpi.h"
#include <stdio.h>
#include <unistd.h>

int main(int argc, char **argv){
        int rank;
        char hostname[256];

        MPI_Init(&argc,&argv);
        MPI_Comm_rank(MPI_COMM_WORLD, &rank);
        gethostname(hostname,255);

        printf("Hello world!  I am process number: %d on host %s\n", rank, hostname);
        MPI_Finalize();

        return 0;
}
```

Note: you probably notice that we have added the complete path for mpi.h file. So make sure for every single example you run on Ryerson EE network you specify this path, otherwise mpich builder will not be able to find MPI.h file.

3.1 Copy this example into an empty file and save it as **ex1.c**
3.2 Go to ex1.c directory on terminal
3.3 Use **mpicc -L/usr/local/mpich-3.1.4/lib ex1.c** instruction to build ex1.c
Note: after executing the above instruction, the mpicc generates a new file **a.out**
3.4 Use **mpirun a.out** to run ex1.c

```
brampton:/home/grad/[redacted]/Desktop> mpicc -L/usr/local/mpich-3.1.4/lib ex1.c
brampton:/home/grad/[redacted]/Desktop> mpirun a.out
Hello world!  I am process number: 0 on host brampton.ee.ryerson.ca
brampton:/home/grad/[redacted]/Desktop>
```

4. Running MPI on the network of computers:

In this section we will run ex1 on the network of computers. For the network of computers we will be using two EE computers (brampton and Guelph).
In order to run MPI on the network of computers we first have to make a trust relationship between the computer we want to use in the network.

4.1 Building Trust Relationship between two computers:

4.1.1 From first computer (Brampton) use **ssh Guelph**
4.1.2 Enter your EE password to go in Guelph computer
4.1.3 From second computer (Guelph) use **ssh brampton**
4.1.4 Enter your EE password to go in brampton computer
4.1.5 Use **exit** in both computers to go back.

At this point both computers have a trust relationship between them.

4.2 Linking both computers

In this section, we will link both computer together so we can run MPI on both computers. We will use brampton as a Master computer and Guelph as Slave.

4.2.1 Go to your home directory on brampton computer.
4.2.2 Use **cd .ssh** to access the trust relationship we built in section 3.1
4.2.3 Use **ssh-keygen –t dsa** to link these computers

This will create three files in the .ssh folder **id_dsa id_dsa.pub known_hosts**

4.2.4 Use **cp id_dsa.pub authorized_keys** to copy id_dsa.pub file and rename it to authorized_keys

At this point you have linked both computers and you can test it by using ssh Guelph. You should be able to go in Guelph without entering your password. If it asks for the password, that's mean you have not linked both computers, go back and start again from section 3.1

Note: For security reasons, authorized_keys file gets deleted after every 10 mins. However, all the stored public and private keys of trusted computers is still save in id_dsa.pub file. You can copy and rename that file again to authorized_keys.

You can also use the screenshort on next page as a reference.

```
brampton:/home/grad/[redacted]> ssh guelph
The authenticity of host 'guelph (172.16.57.1)' can't be established.
RSA key fingerprint is b3:3f:77:c5:60:05:24:88:ed:dd:2e:e9:ee:be:96:92.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'guelph,172.16.57.1' (RSA) to the list of known hosts
.
[redacted]@guelph's password:
Last login: Sun Oct 25 16:47:35 2015 from pascal.ee.ryerson.ca
guelph:/home/grad/[redacted]> exit
Connection to guelph closed.
brampton:/home/grad/[redacted]> cd .ssh
brampton:/home/grad/[redacted]/.ssh> ls
known_hosts
brampton:/home/grad/[redacted]/.ssh> ssh-keygen -t dsa
Generating public/private dsa key pair.
Enter file in which to save the key (/home/grad/[redacted]/.ssh/id_dsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/grad/[redacted]/.ssh/id_dsa.
Your public key has been saved in /home/grad/[redacted]/.ssh/id_dsa.pub.
The key fingerprint is:
bc:1f:75:f2:58:26:17:49:4f:61:83:69:a7:4e:8f:98 [redacted]@brampton.ee.ryerson.ca
The key's randomart image is:
+--[ DSA 1024]----+
|              ++o|
|             =.=.|
|            . = .|
|       .     o . |
|        S   O B  |
|         . E % . |
|        . . . .  |
|         . .     |
|          .      |
+-----------------+
brampton:/home/grad/[redacted]/.ssh> ls
id_dsa  id_dsa.pub  known_hosts
brampton:/home/grad/[redacted]/.ssh> cp id_dsa.pub authorized_keys
brampton:/home/grad/[redacted]/.ssh> ls
authorized_keys  id_dsa  id_dsa.pub  known_hosts
brampton:/home/grad/[redacted]/.ssh> ssh guelph
Last login: Sun Oct 25 16:48:28 2015 from brampton.ee.ryerson.ca
guelph:/home/grad/[redacted]>
```

4.3 Running the example ex1on the network of computers:

At this point if everything was successful, you can run MPI on the network of computer

4.3.1 First built the **ex1** again using **mpicc -L/usr/local/mpich-3.1.4/lib ex1.c**

4.3.2 Run the ex1 by specifying hosts, master and slaves by using the following instruction

mpirun –np #ofProcessors -host masterIp,slaveIp1,slaveIp2…. a.out
mpirun –np 2 –host brampton,guelph a.out

3.3.3 If everything was correct, you should see the following output.

```
brampton:/home/grad/[redacted]/.ssh> ls
id_dsa  id_dsa.pub  known_hosts
brampton:/home/grad/[redacted]/.ssh> cp id_dsa.pub authorized_keys
brampton:/home/grad/[redacted]/.ssh> ls
authorized_keys  id_dsa  id_dsa.pub  known_hosts
brampton:/home/grad/[redacted]/.ssh> cd..
brampton:/home/grad/[redacted]> cd Desktop
brampton:/home/grad/[redacted]Desktop> mpicc -L/usr/local/mpich-3.1.4/lib ex1.c
brampton:/home/grad/[redacted]/Desktop> mpirun -np 2 -host brampton,guelph a.out
Hello world!  I am process number: 0 on host brampton.ee.ryerson.ca
Hello world!  I am process number: 1 on host guelph.ee.ryerson.ca
brampton:/home/grad/[redacted]/Desktop>
```