Instalación de OpenMPI

José Luis Quiroz Fabián

Abirl 2016

1 OpenMPI

MPI ("Message Passing Interface", Interfaz de Paso de Mensajes) es un estándar que define la sintaxis y la semántica de las funciones contenidas en una biblioteca de paso de mensajes. Existen diferentes implementaciones de MPI, la más conocida es OpenMPI. En el cluster *mathcluster* la instalación de OpenMPI se realizó en una carpeta compartida que se encuentra en el servidor: */usr/local/herramientas*. Los pasos de su instalación son los siguientes:

1.1 Instalación OpenMPI (como root)

1. Descargar OpenMPI:

   [root@localhost]: *wget www.open-mpi.org/software/ompi/v1.10/downloads/openmpi-1.10.2.tar.gz*

2. Crear la carpeta /usr/local/herramientas

   [root@localhost]: *mkdir /usr/local/herramientas*

3. Mover el archivo a */usr/local/herramientas*:

   [root@localhost]: *mv openmpi-1.10.2.tar.gz /usr/local/herramientas*

4. Crear el directorio openmpi

   [root@localhost]: *mkdir /opt/openmpi*

5. Descomprimir el archivo descargado

   [root@localhost]: *tar -xzvf openmpi-1.10.2.tar.gz*

6. Ingresar a la carpeta generada

   [root@localhost]: *cd openmpi-1.10.2*

7. Preparar la configuración configuración:

   [root@localhost]:*./configure –prefix=/opt/openmpi –without-hwloc –without-openib*

Usando Java:

```
./configure  --prefix=/opt/openmpi --without-hwloc --with-verbs --without-openib
--enable-mpi-java --with-jdk-bindir=/opt/jdk1.8.0_25/bin
--with-jdk-headers=/opt/jdk1.8.0_25/include
```

8. Para la creación de los ejecutables:

   [root@localhost]:*make install all*

9. Agregar en el archivo /etc/bashrc (en sistemas basados en RedHat) o en /etc/bash.bashrc (para sistemas basados en Debian, como Ubuntu) al final:

```
export OPENMPI_LIB=/opt/openmpi/lib:/opt/openmpi/lib/openmpi
export OPENMPI_BIN=/opt/openmpi/bin
export PATH=$OPENMPI_BIN:$PATH
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$OPENMPI_LIB
```

   Lo anterior se tiene que realizar en cada nodo del cluster y en el servidor.

10. Realizar

    [root@localhost]:*source /etc/bashrc*

11. Para limpiar un make previo:

    [root@localhost]:*make distclean*

12. Instalar (si es que no esta instalado) el ssh server:

    Para sistemas basados en Red-Hat
    [root@localhost]:*yum install openssh-server*

    Para sistemas basados en Debian
    [root@localhost]:*apt-get install openssh-server*

Desde los usuarios que utilizarán MPI (no root)

1. Ejecutar (presionar enter hasta que la consola se libere)

   [usuario@localhost]:*ssh-keygen -t rsa*

```
Generating public/private rsa key pair.
Enter file in which to save the key (/home/usuario/.ssh/id_rsa):
Created directory '/home/usuario/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/usuario/.ssh/id_rsa.
Your public key has been saved in /home/usuario/.ssh/id_rsa.pub.
```

```
The key fingerprint is:
5e:04:22:44:93:11:b1:1b:e4:8b:9f:23:b3:f7:78:79 usuario@dhcppc3
The key's randomart image is:
+--[ RSA 2048]----+
|   oX=. .        |
|   o.+ . .       |
|    +     .      |
|   . +   .       |
|  . o   S .      |
|   . . . .       |
|  o +  ..        |
|   +.oo E        |
|  ...o..         |
+-----------------+
```

2. Ingresar a la carpeta .ssh

   [usuario@localhost]:*cd .ssh/*

3. (En la carperta .ssh) Copiar el contenido del archivo id_rsa.pub a authorized_keys

   [usuario@localhost]:*cp id_rsa.pub authorized_keys*

4. (En la carperta .ssh) Asignar los permisos adecuados

   [usuario@localhost]:*chmod 600 **

1.2 Ejemplo en OpenMPI (como usuario)

Para ejecutar en el cluster, los usuarios deben ingresar a cualquier nodo del mismo ($mathcluster1 - mathcluster9$). Los pasos para compilar y ejecutar un programa en OpenMPI son los siguientes:

Creación y compilación:

1. Escribir un programa en C; por ejemplo $informacion.c$

   [usuario@localhost]:*vim informacion.c (vim u otro editor de texto)*

```
#include <stdio.h>
#include <mpi.h>
int main(int argc,char **argv){

    int mi_id, numprocs,len;
    char name[MPI_MAX_PROCESSOR_NAME];

    MPI_Init(&argc,&argv);
    MPI_Comm_size(MPI_COMM_WORLD,&numprocs);
    MPI_Comm_rank(MPI_COMM_WORLD,&mi_id);
    MPI_Get_processor_name(name, &len);
```

```
    printf("Proceso %d  en la maquina %s  de un total de  %d\n",mi_id,name,numprocs);

    MPI_Finalize();
}
```

2. Compilar usando *mpicc*

   [usuario@localhost]:*mpicc informacion.c -o informacion*

Ejecución

1. Escribir un archivo con la información de las máquinas del cluster; por ejemplo *maquinas*

   [usuario@localhost]:*vim maquinas*

```
mathcluster1
mathcluster2
mathcluster3
mathcluster4
mathcluster5
mathcluster6
mathcluster7
mathcluster8
mathcluster9
```

2. Ejecutar el programa $informacion$ por medio de *mpirun*

   [usuario@localhost]:*mpirun -np 9 -hostfile maquinas informacion*

   donde, $-np$ es el número de procesos a ejecutar y $-hostfile$ indica en que nodos del cluster ejecutar.

3. La salida será similar a:

```
Proceso 0  en la maquina mathcluster1  de un total de  9
Proceso 1  en la maquina mathcluster1  de un total de  9
Proceso 2  en la maquina mathcluster1  de un total de  9
Proceso 3  en la maquina mathcluster1  de un total de  9
Proceso 5  en la maquina mathcluster2  de un total de  9
Proceso 8  en la maquina mathcluster3  de un total de  9
Proceso 6  en la maquina mathcluster2  de un total de  9
Proceso 7  en la maquina mathcluster2  de un total de  9
Proceso 4  en la maquina mathcluster2  de un total de  9
```