Práctico 0: ClusterUY – Linux

Descripción / Configuraciones / Tutorial

13/04/2023

1. Tutorial de Linux

Se recomienda seguir el Curso Básico de Linux, http://iie.fing.edu.uy/personal/vagonbar/sample-page/curso-basico-de-linux/.

Comandos utilizados		
Comando/Sintaxis	Descripción	Ejemplos
cat *arch1 [...archN*	Concatena y muestra archivos	**cat** /home/passwd **cat** dict1 dict2 dict
cd [dir]	Cambia de directorio	**cd** /home/mesoescala
chmod *permisos fich*	Cambia los permisos de un archivo	**chmod** +x miscript
cp *arch1...archN dir*	Copia archivos	**cp** foo foo.backup
du *[-sabr] fich*	Reporta el tamaño del directorio	**du** -s /home/
file *arch*	Muestra el tipo de un archivo	**file** arc_desconocido
find *dir test acción*	Encuentra archivos	**find** . -name “.bak” – print
grep *[-cilnv] expr archivos*	Busca patrones en archivos	**grep** mike /etc/passwd
head -count fich	Muestra el inicio de un archivo	**head** prog1.c
mkdir *dir*	Crea un directorio.	**head** *temp*
mv *fich1 ...fichN dir*	Mueve un archivo(s) a un directorio	**mv** a.out prog1
mv *fich1 fich2*	Renombra un archivo.	**mv** .c prog_dir
ls	Lista el contenido del directorio	**ls** -l /usr/bin
pwd	Muestra la ruta del directorio actual	**pwd**
rm *fich*	Borra un fichero.	**rm** foo.c
rm -r *dir*	Borra un todo un directorio	**rm -rf** prog_dir
rmdir *dir*	Borra un directorio vacío	**rmdir** prog_dir
tail *-count fich*	Muestra el final de un archivo	**tail** prog1.c
vi *arch*	Edita un archivo.	**vi** .profile
ln *[-s] fich acceso*	Crea un acceso directo a un archivo	**ln** -s /users/mike/.profile
cal [[mes] año]	Muestra un calendario del mes/año	**cal** 1 2025
date [mmddhhmm] [+form]	Muestra la hora y la fecha	**date**
echo *string*	Escribe mensaje en la salida estándar	**echo** “Hola mundo”
man *comando*	Ayuda del comando especificado	**man** gcc **man** -k printer
exit	terminar la sesión, cierra el terminal.	**exit**
who / **rwho**	Muestra información de los usuarios conectados al sistema.	**who**
diff *[-e]arch1 arch2*	Encuentra diferencia entre archivos	**diff** foo.c newfoo.c

2. ClusterUY

El Centro Nacional de Supercomputación (ClusterUY), https://cluster.uy/, es una plataforma de computación de alto desempeño que posee la capacidad de gestionar en forma coordinada múltiples recursos de cómputo. Cuenta con 1120 núcleos Intel Xeon-Gold 6138 2.00GHz y 96 núcleos AMD EPYC 7642 2.30GHz, 3,8 TB de memoria RAM y 100.352 núcleos de cómputo GPU Nvidia Tesla P100 con 12Gb de memoria interconectados por una red de alta velocidad Ethernet de 10 Gbps.

2.1. Cómo acceder al ClusterUY

Si se inscribió al curso, **Modelos numéricos de Mesoescala aplicados a Ingeniería**, sitio EVA: https://eva.fing.edu.uy/course/view.php?id=1480, se le ha asignado un usuario nuevo, *mesoescalaXX*.

La conexión con el clusterUY se realiza a través de SSH, *secure shell*, y para la autenticación se necesitan un par de claves, pública y privada, (en ningún caso el usuario manejará password).

2.1.1. ¿Cómo acceder desde Linux?

Abrir una terminal, y ejecutar las siguientes líneas de comando:

chmod *600 **

ssh *-i* mesoescalaXX_key mesoescalaXX@cluster.uy

Para ingresar a clusteruy con salidas gráfica se agrega la opción *-X*.

ssh *-i* mesoescalaXX_key *-X* mesoescalaXX@cluster.uy

Practique los siguientes comandos: **pwd**, **ls**, **vi**, **cd** .

2.1.2. ¿Cómo acceder desde Windows?

En este caso se debe seguir el instructivo publicado en el EVA. Se utilizará *MobaXterm*, https://mobaxterm.mobatek.net/

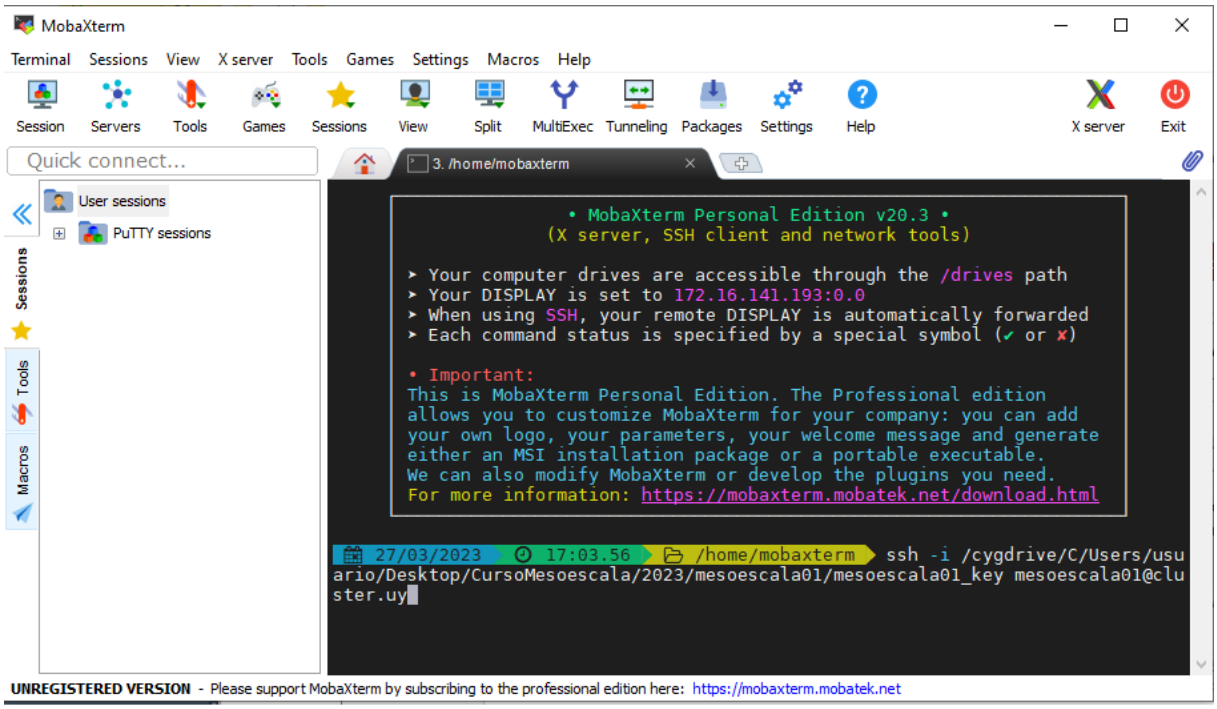

Figura 1: Acceso a clusterUY desde Windows.

Practique los siguientes comandos: **pwd**, **ls**, **vi**, **cd**.

En consola ejecute las líneas de código presentadas en la Figura 2 para diferentes formatos de fechas.

```
date +"Year: %Y, Month: %m, Day: %d"
```

```
date "+DATE: %D%nTIME: %T"
```

```
date -d '16 Dec 1974' +'%A, %d %B %Y'
```

```
date -d "last week"
```

```
date_now=$(date "+%F-%H-%M-%S")
```

```
echo $date_now
```

Figura 2: Líneas de código para visualizar fechas.

2.2. Comandos específicos para el ClusterUY

Al acceder al ClusterUY se va encontrar en la máquina **login.cluster.uy**. La misma se puede utilizar para el manejo de archivos, despacho de trabajos al gestor, tareas livianas como la edición de archivos y en general para el acceso a todos los servicios del cluster. Esta máquina no dispone de herramienta de compilación ni desarrollo de software y **no se deben ejecutar procesos de computo intensivo en esta máquina bajo ningún concepto**.

Para las tareas de compilación y las relacionadas con el desarrollo de software se deben realizar solicitando un trabajo interactivo y se debe solicitar el tiempo estimado de ejecución con el parámetro **–time=** .

```
srun --job-name=mitrabajo --time=05:20:00 --ntasks=1 --partition=normal --qos=normal --mem=512 --pty bash -l
```

Figura 3: Comando srun.

2.2.1. Cómo ejecutar un trabajo en ClusterUY

Antes de iniciar un trabajo en el ClusterUY es necesario determinar los recursos de cómputo que serán necesarios para su ejecución. Los principales recursos que deben especificarse son:

- Tiempo total de ejecución del trabajo
- Cantidad de núcleos utilizados
- Cantidad de memoria utilizada
- Cantidad de GPUs requeridas

Los trabajos en el cluster se ejecutan fuera de línea. Esto permite al usuario desconectarse del cluster y recibir una notificación cuando su trabajo termina su ejecución. La ejecución de este tipo de trabajos es iniciada con el comando **sbatch**. Este comando recibe como argumento un **script** que contiene la especificación del trabajo a ejecutar y los comandos necesarios para su ejecución. La especificación del trabajo se realiza mediante comentarios especiales que comienzan con **#SBATCH**.

```
#!/bin/bash
#SBATCH --job-name=mitrabajo
#SBATCH --ntasks=1
#SBATCH --mem=2048
#SBATCH --time=6:00:00
#SBATCH --tmp=9G
#SBATCH --partition=normal
#SBATCH --qos=normal
#SBATCH --mail-type=ALL
#SBATCH --mail-user=mi@correo

source /etc/profile.d/modules.sh

cd ~/miaplicacion
./mibinario arg1 arg2
```

Figura 4: Ejemplo de script con las especificaciones del trabajo que se quiere ejecutar y los comandos necesarios para su ejecución. Extraído de https://cluster.uy/

El trabajo es cargado en el sistema con el comando **sbatch** *miscript.sh*. Se le asignará un idéntificador único (ID) y quedará en espera hasta que se encuentren disponibles los recursos de cómputo necesarios para su ejecución. Una vez disponibles, el sistema asigna los recursos al trabajo y lanza su ejecución ejecutando secuencialmente el contenido del *script* de especificación.

Una vez finalizada su ejecución, SLURM dejará disponibles el archivo **slurm-<ID>.out** con el contenido de la salida a pantalla de la ejecución donde <ID> es el ID asignado al trabajo. Este archivo incluye la salida estándar y la salida de error.

Se recuerdan los siguientes comandos de utilidad para cuando se ejecuten los trabajos del curso:

- squeue -u <usuario>
- scancel <ID>

2.2.2. Trabajo interactivo

Un trabajo interactivo permite conectarse directamente a un nodo de cómputo y trabajar con una consola. Para iniciar un trabajo como este es necesario usar el comando **srun** con la directiva *–pty bash -l*. Por ejemplo:

srun −−job-name=mitrabajo −−time=00:20:00 −−ntasks=1 −−partition=normal −−qos=normal −−mem=512 −−pty bash -l

3. Scripts de bash

El uso del shell Bash como lenguaje de programación resulta muy útil para realizar tareas repetidas con frecuencia. Es muy eficiente en la ejecución de ”scripts”, sucesiones de comandos escritos en un archivo de texto.

Por ejemplo, en un editor de texto escribiremos lo siguiente y lo guardaremos con el nombre *hola.sh*

```
#!/bin/bash
# Este es nuestro primer progrma
echo Hola Mundo
```

Figura 5: Ejemplo de script bash.

A continuación iremos a la terminal y lo ejecutaremos:

```
~$ ./hola.sh.
```

Figura 6: Ejecutar script bash.

La primera línea del script le indica al sistema que tiene que usar la shell BASH. La segunda línea es un comentario para consumo humano, todas las líneas que comiencen por # son ignoradas por la máquina y nos sirven para incluir comentarios destinados a programadores o usuarios. En la tercera línea tenemos el comando **echo** que sirve para imprimir texto en la pantalla.

Para copiar el script a la carpeta mesoescalaXX de su pc personal, utilice el **scp -i**. También podrá copiar carpetas con el comando **scp -r**. A continuación se deja un ejemplo:

scp -i mesoescalaXX_key mesoescalaXX@cluster.uy:/clusteruy/home/mesoescalaXX/hola.sh .