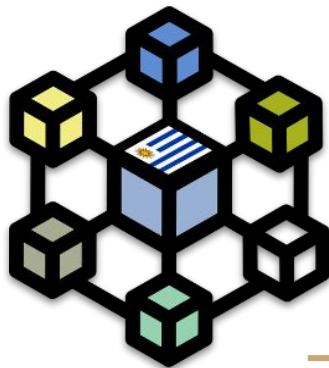


Centro nacional de supercomputación

-- cluster.uy --



Centro Nacional de Supercomputación

- Iniciativa académica para proporcionar poder de cómputo en el ámbito nacional.
- Arquitectura cluster: agregación de recursos de cómputo.
- *Abierto* a estudiantes, investigadores, técnicos, empresas e instituciones del país y del mundo.
- Sin fines de lucro, con una gestión *autosustentable*.



Especificaciones de hardware



- Un servidor de 96 cores AMD EPYC 7642 con 256 GB de ram
- 28 servidores HPE DL380 Gen10:
 - 2 procesadores Intel Xeon Gold 6138 CPU.



Cada uno con 20 núcleos @ 2.00 GHz. (40 cpus por servidor)

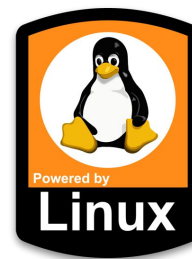
- GPU Nvidia Tesla P100 con 12 GB de RAM.
- Conectividad 10 GbE.
- Linux CentOS 7.



Capacidad de almacenamiento: 300 TB.

Especificaciones de hardware

NOMBRE	PROCESADORES	# NÚCLEOS	MEM.	GPU/NODE	DISCO
node[01-14]	Xeon Gold 6138	40	128 GB	NVIDIA P100	300 GB SSD
node[15][26-28]	Xeon Gold 6138	40	128 GB	-	300 GB SSD
node[17-22]	Xeon Gold 6138	40	128 GB	NVIDIA P100 x 2	300 GB SSD
node23	Xeon Gold 6138	40	128 GB	NVIDIA P100 x 3	300 GB SSD
node[24-25]	Xeon Gold 6138	40	512 GB	-	300 GB SSD
node31	AMD EPYC 7642	96	256 GB	-	150 GB SSD



Solicitar un usuario

Para solicitar el alta de un usuario es necesario completar el formulario en <http://cluster.uy/registro>

- E-mail institucional (*muy importante!*).
- Nombre completo.
- Descripción de uso, docente responsable.
- Clave pública OpenSSH: RSA o ECDSA para autenticación.

Acceder a cluster.uy



A través de cualquier cliente de SSH que posea una clave privada generada previamente:

```
$ ssh usuario@cluster.uy  
[usuario@login ~]$
```

“casi todas las distribuciones de Linux traen cliente OpenSSH de forma nativa”



Cuota de almacenamiento

- El espacio de almacenamiento en **cluster.uy** es limitado.
- Cada usuario tiene una cuota de 300 GB de almacenamiento asociada a su **HOME**.
- Se pueden crear **otros grupos** asociados a los usuarios.
 - Estos grupos funcionan como directorios compartidos con una cuota independiente.
 - Se solicitan a **soporte@cluster.uy**

Gestor de recursos

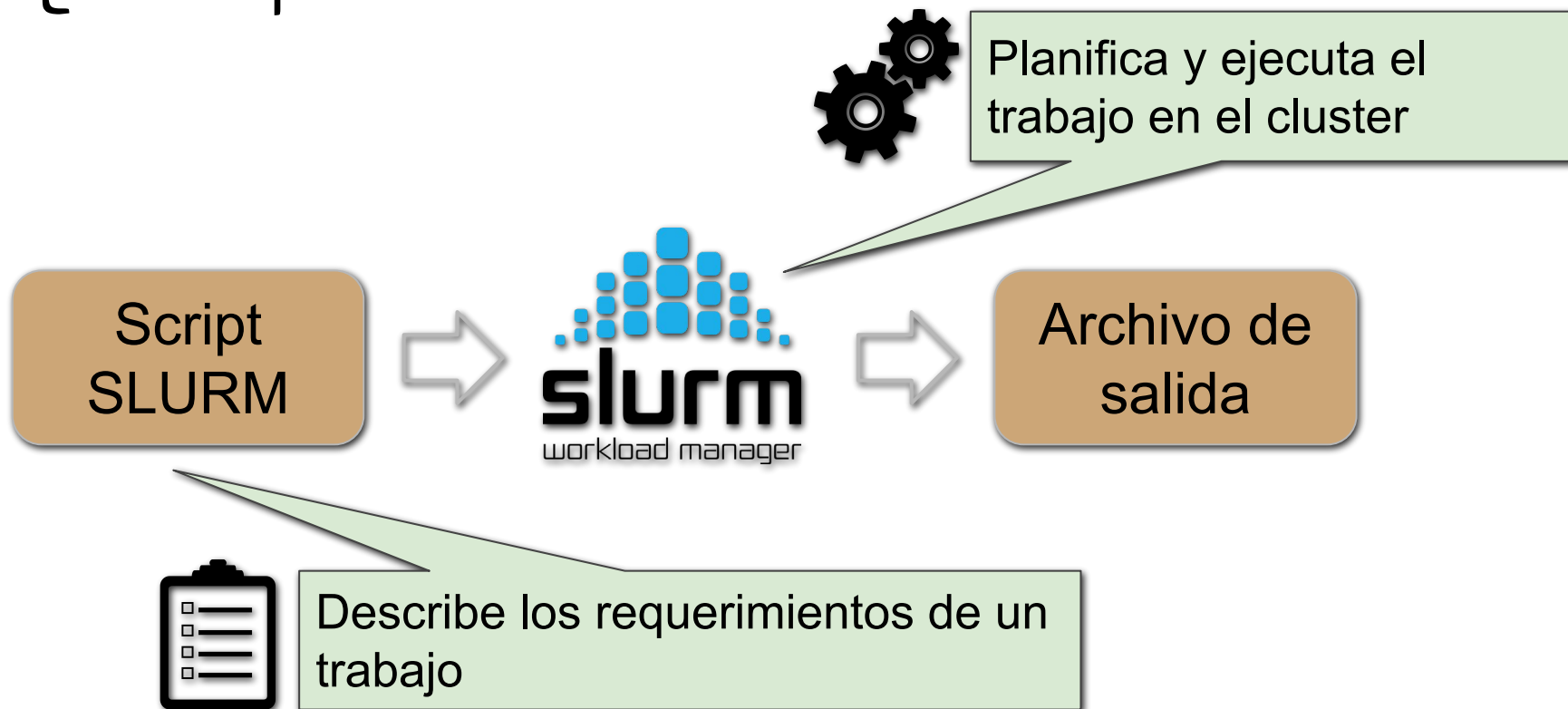


Simple Linux Utility for Resource Management (SLURM):

- Organiza la ejecución de los trabajos.
- Tipos de trabajos: interactivos y por lotes.
- Orientado al trabajo por lotes.



¿Cómo funciona SLURM?



Tipos de trabajos



- **Trabajo interactivo**

El usuario debe estar ***en línea***.

Se lanzan comandos interactivamente en el nodo de cómputo.

Ideales para compilar, testear y debuggear trabajos.

Son necesariamente cortos (hasta 30 minutos).

- **Trabajo por lotes (batch)**

El usuario trabaja ***fuera de línea***.

Se obtiene toda la salida de la ejecución junta al finalizar.

Pueden ser de larga duración (hasta 5 días).

Trabajos interactivos

Se le asigna un nodo al usuario con un procesador. El usuario debe estar en línea y se trabaja directamente sobre el nodo de cómputo. El comando interactivo genera una instancia básica de 30 minutos, con un procesador solo.

```
$ interactive -g
```

Trabajo interactivo (avanzado)

- Se **solicita** su inicio con el comando **srun**.
 - El comando se bloquea hasta que el trabajo pueda comenzar.
- Se debe especificar el tiempo de uso del cluster.
 - Transcurrido este tiempo el trabajo será **terminado**.
- Inicio de un trabajo interactivo:

```
$ srun --time=00:30:00 --pty bash -l
```

Tiempo máximo

Abrir una sesión de
bash en una terminal

Ejemplo interactivo en R



```
$ ssh [usuario]@cluster.uy

[usuario@login ~]$ interactive -g

[usuario@node15 ~]$ R

> mtscaled <- as.matrix(scale(mtcars))

> pdf("heatmap.pdf")

> heatmap(mtscaled, Colv=F, scale='none')

> dev.off()

> q()
```

Ejemplo interactivo en R



```
[usuario@node15 ~]$ ls *.pdf
```

```
heatmap.pdf
```

```
[usuario@node15 ~]$ exit
```

```
[usuario@login ~]$ exit
```

```
$ scp usuario@cluster.uy:heatmap.pdf .
```

```
heatmap.pdf
```

```
100% 7699 346.1KB/s 00:00
```

el archivo pdf generado
debe copiarse localmente
para su visualización

Trabajos por lotes (batch)

El usuario genera un script solicitando recursos y el pipeline a ejecutar. No requiere que el usuario esté conectado mientras se ejecuta.

```
$ sbatch script.sbatch
```

```
1  #!/bin/bash
2  #SBATCH --job-name=charla
3  #SBATCH --ntasks=1
4  #SBATCH --cpus-per-task=1
5  #SBATCH --mem=1G
6  #SBATCH --time=0:01:00
7  #SBATCH --partition=normal
8  #SBATCH --qos=normal
9  #SBATCH --reservation=charla
10 #SBATCH --mail-type=ALL
11 #SBATCH --mail-user=siturria@fing.edu.uy
12 R CMD BATCH heatmap.r
```

Trabajo batch: ejemplo script sbatch

```
#!/bin/bash -l
#SBATCH --job-name=heatmap
#SBATCH --ntasks=1
#SBATCH --mem=3G
#SBATCH --time=00:20:00
#SBATCH --partition=normal
#SBATCH --qos=normal
#SBATCH --mail-type=ALL
#SBATCH --mail-user=mail
```

3 Gb de mem. física
por nodo

20 min. de ejecución

Partición y QoS
donde ejecutará el
trabajo

Programa a ejecutar

```
r CMD BATCH heatmap.r
```


Particiones utilizables y límites

NOMBRE	# NÚCLEOS DISPONIBLES	MAX. TIEMPO	MAX. TRABAJOS	OTROS LÍMITES
normal	560	5 días	30	60 núcleos, 512 GB RAM
besteffort	1120	5 días	60	60 núcleos, 1 TB RAM

Calidad de servicio (QOS, Quality of Service)

QOS	MAX. RECURSOS	MIN. RECURSOS	PARTICIÓN COMPATIBLE
gpu	60 núcleos, 512 GB RAM, 4 GPU	1 GPU	normal
rapida	8 núcleos, 8GB RAM	-	normal
normal	60 núcleos, 512 GB RAM, 0 GPU	-	normal
besteffort	60 núcleos, 1 TB RAM	-	besteffort

Calidad de servicio (QoS, Quality of Service)

- QoS **normal**:
 - Tiempo máximo de ejecución 5 días.
 - Máximo 40 cpus y 500gb de RAM.
- QoS **rapida**:
 - Tiempo máximo de ejecución: 30 mins.
 - Prioridad superior a **normal** y **gpu**.
- QoS **gpu**:
 - Es necesaria cuando se solicita el uso de algún gpu.
- QoS **besteffort**:
 - Prioridad mínima.
 - Es expropiable por trabajos de otras QoS.

Control del espacio: quota

Se lista la cuota de **todos** los grupos asociados al usuario actual.

```
$ quota -gs
Disk quotas for group clusterusers (gid 10002): none
Disk quotas for group usuario (gid 10115):
Filesystem space quota limit grace files quota limit grace
fileserver:/home
                218G  300G  301G                8552      0      0
```

Espacio usado

Máximo utilizable

Máximo de gracia

Cantidad de archivos
creados

Límite de cantidad
de archivos

Control de colas: squeue

Tiempo de ejecución hasta el momento. Formato: DD-HH:MM:SS

Razón por la que no está ejecutando

```
[arnoldo@login ~]$ squeue -u arnoldo
```

JOBID	PARTITION	NAME	USER	ST	TIME	NODES	NODELIST (REASON)
493442	normal	FrCaso	arnoldo	PD	0:00	1	(Resources)
493443	normal	FrCaso	arnoldo	PD	0:00	1	(Priority)
493444	normal	FrCaso	arnoldo	PD	0:00	1	(ReqNodeNotAvail)
493445	normal	FrCaso	arnoldo	PD	0:00	1	(QOSMaxJobsPerUserLimit)
491643	gpu	pi	arnoldo	R	0:06	1	node18
493492	normal	cbo006	arnoldo	R	46:59	2	node[15-16]

Estado: Pending (PD),
Running (R)

Nodos asignados
para la ejecución

Control de trabajos: scontrol show job

```
[arnoldo@login ~]$ scontrol show job 22644
JobId=22644 JobName=ejemplo
  UserId=arnoldo(10074) GroupId=users(10002) MCS_label=N/A
  Priority=1 Nice=0 Account=udelar QOS=normal
  JobState=RUNNING Reason=None Dependency=(null)
  Requeue=1 Restarts=0 BatchFlag=1 Reboot=0 ExitCode=0:0
  RunTime=9-14:55:24 TimeLimit=9-23:50:00
  SubmitTime=2018-11-27T17:40:50
  EligibleTime=2018-11-27T17:40:50
  StartTime=2018-11-27T18:29:21 EndTime=2018-12-07T18:19:21
  PreemptTime=None SuspendTime=None Partition=normal
  NodeList=node26 BatchHost=node26
  NumNodes=1 NumCPUs=1 NumTasks=1 CPUs/Task=1 ReqB:S:C:T=0:0:*:*
  TRES=cpu=1,mem=2G,node=1,billing=1
  Socks/Node=* NtasksPerN:B:S:C=0:0:*:* CoreSpec=*
  MinCPUsNode=1 MinMemoryNode=2G MinTmpDiskNode=0
  (...)
```

Copiar archivos desde y hacia cluster.uy

- Usando el comando **scp** puede copiar archivos desde cluster.uy a su máquina local o viceversa.

```
$ scp usuario@cluster.uy:heatmap.pdf .
```

```
heatmap.pdf                100% 7699  346.1KB/s  00:00
```

- **scp** ejemplo de copia de un archivo local hacia cluster.uy

```
$ scp heatmap.r usuario@cluster.uy:
```

```
heatmap.r                  100% 169  346.1KB/s  00:00
```

Copiar archivos desde y hacia cluster.uy

- Con **rsync** puede copiar y sincronizar directorios enteros de forma incremental desde cluster.uy a su máquina local y viceversa.

```
$ rsync -zvhr [directorio_local]/ cluster.uy:[directorio_remoto]
```

```
$ rsync -zvhr cluster.uy:[directorio_remoto] [directorio_local]
```

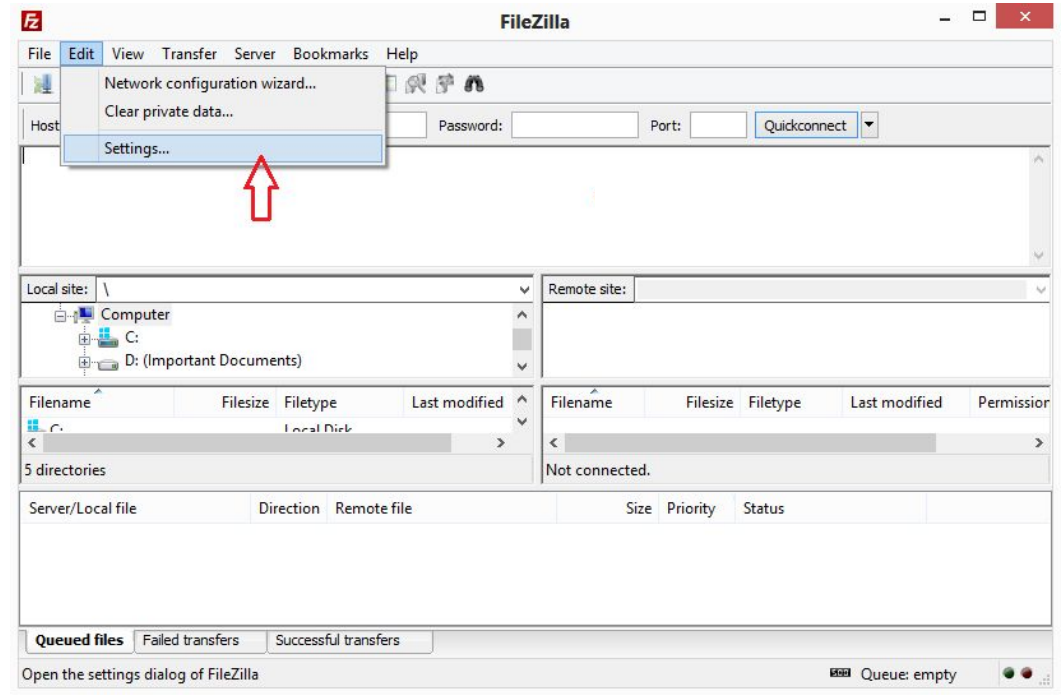

Copiar archivos desde y hacia cluster.uy

- Desde windows es recomendable que instale FileZilla si desea tener una alternativa gráfica para manejar archivos entre cluster.uy y su máquina local. También es compatible con otros sistemas operativos.



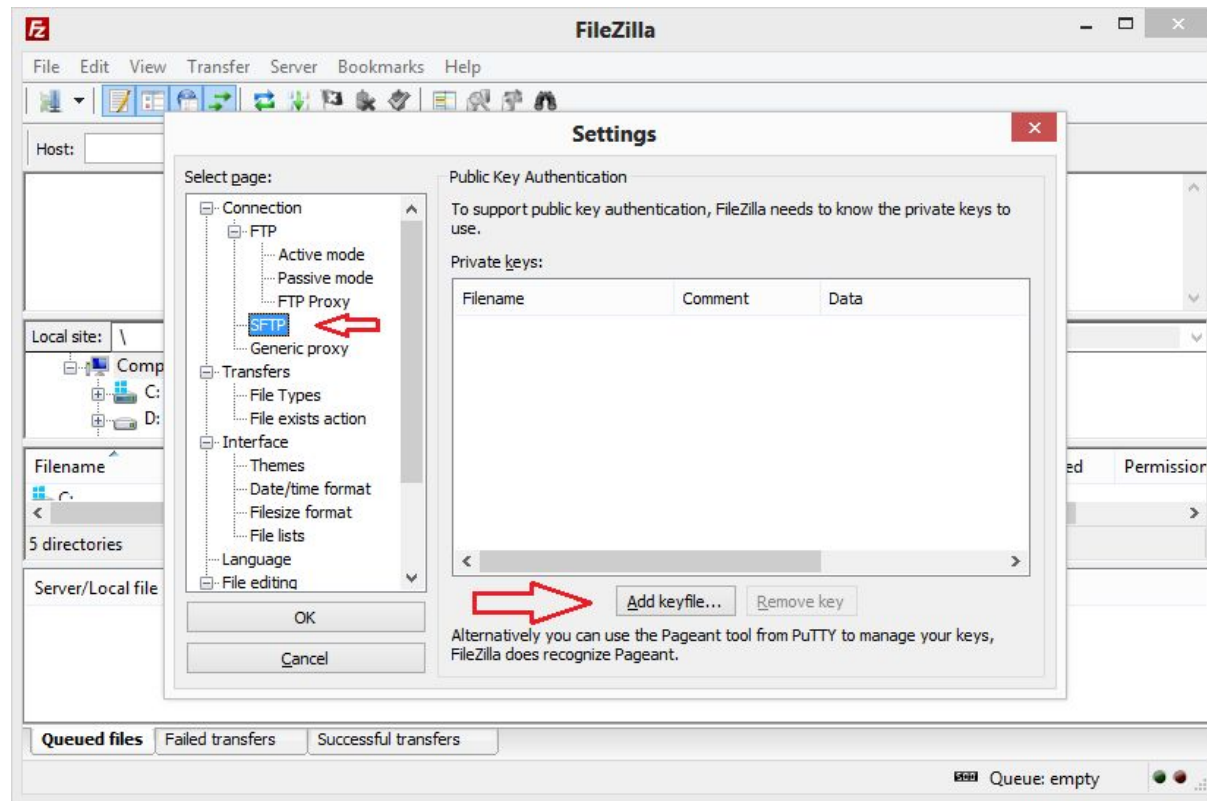
Copiar archivos desde y hacia cluster.uy

- Para usar FileZilla debe importar la clave privada generada.



Copiar archivos desde y hacia cluster.uy

- Para usar FileZilla debe importar la clave privada generada.

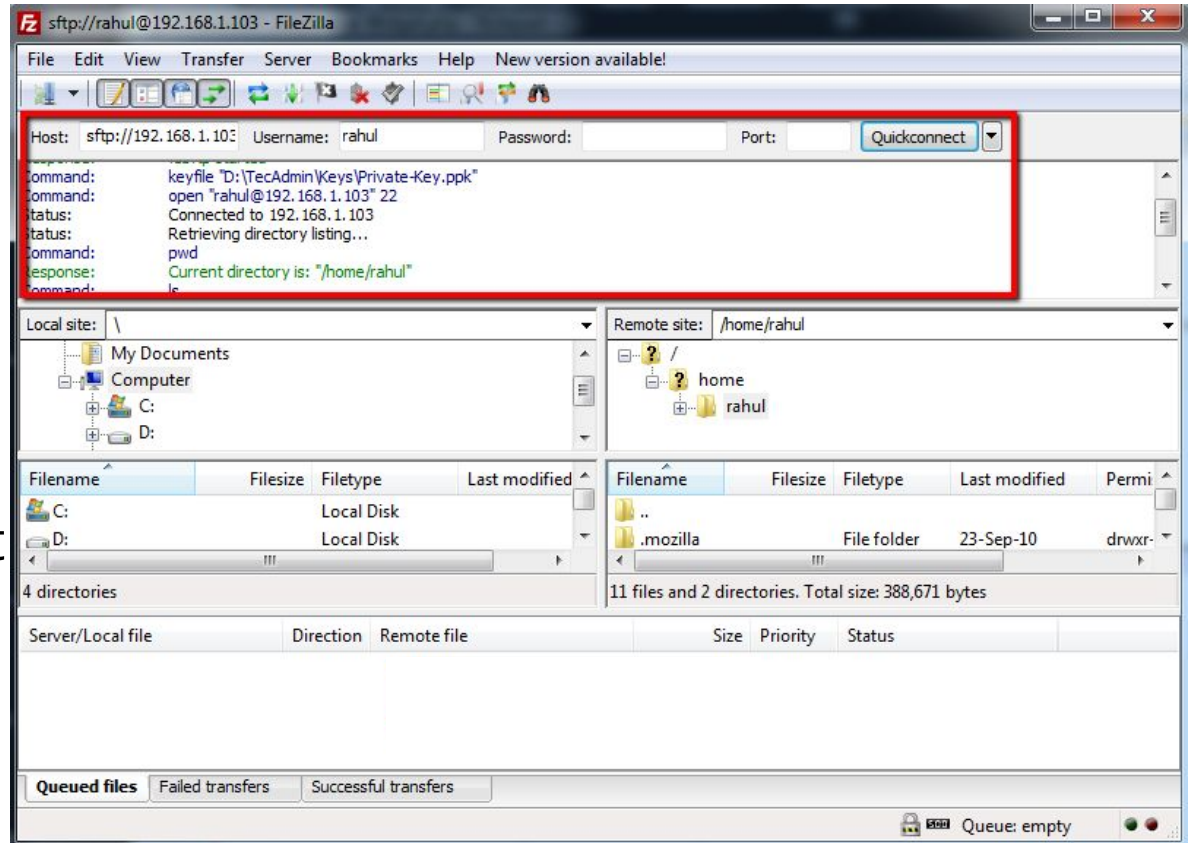


Copiar archivos desde y hacia cluster.uy

- En el **Host** hay que ingresar:

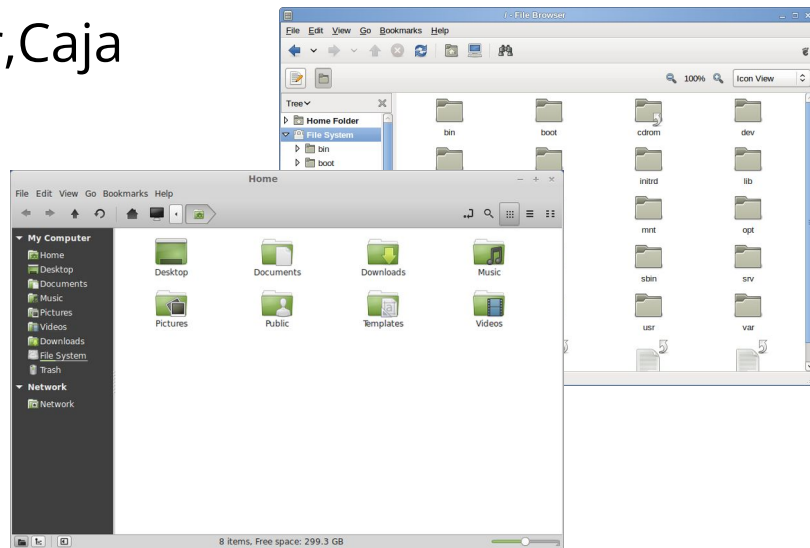
sftp://cluster.uy

- Debe completar la casilla **Username**
- Click en Quickconnect



Copiar archivos desde y hacia cluster.uy

- Los gestores de archivos de linux generalmente traen un cliente sftp incorporado. Por ejemplo:
Nautilus, Nemo, Thunar, Konqueror, Caja



Copia de gran volumen de datos

- Cuando se necesita copiar un gran volumen de datos y con el fin de ser solidario y no consumir ancho de banda en login se debe utilizar el puerto 10022 de cluster.uy.
- *scp*

```
$ scp -P 10022 [archivo origen] cluster.uy:[carpeta destino]
```

- *rsync*

```
$ rsync --port=10022 [archivo origen] cluster.uy:[carpeta destino]
```

Parámetros de configuración

- SLURM tiene una gran cantidad de parámetros que permiten configurar la ejecución de trabajos.
- Existen parámetros para configurar:
 - **Tiempo:** tiempo máximo, tiempo de inicio y fin, etc.
 - **I/O:** archivo de entrada o de salida, etc.
 - **Tareas:** cantidad, localización, etc.
 - **CPU:** total, por tarea, por gpu, localización, etc.
 - **Memoria:** total, por tarea, por cpu, por gpu, etc.
 - **Partición, QoS y account**
 - **Notificaciones:** casilla de correo y tipos de alertas a notificar.

Parámetros de configuración: Partición y QoS

- **--partition=<nombre>** especifica una partición.
 - Es posible indicar una lista de particiones separadas por coma.
 - El trabajo ejecutará en la primera partición de la lista que pueda ejecutarlo.
- **--qos=<nombre>** especifica una calidad de servicio.
- **--account=<nombre>** especifica una account trabajo.

Parámetros: tiempo

- **--time=<time>** límite de tiempo total de ejecución del trabajo.
 - Formato: **DD-HH:MM:SS**
 - Este parámetro es **obligatorio!**
- **--begin=<time>** el sistema garantiza que el trabajo no iniciará antes del momento especificado. Por ejemplo:
 - **--begin=16:00**
 - **--begin=now+1hour** ○ **--begin=now+60**
 - **--begin=2010-01-20T12:34:00**
- **--deadline=<OPT>** límite para la finalización de un trabajo.
 - El trabajo falla si el sistema no puede finalizarlo antes de su deadline, i.e. **start > deadline - time**

Parámetros: recursos de cómputo

- **--ntasks=<número>** cantidad de tareas a ejecutar.
 - Una tarea es indivisible y se ejecuta en un único nodo.
 - Este parámetro es **obligatorio!**
- **--ntasks-per-node=<número>** cantidad de tareas por nodo.
- **--cpus-per-task=<número>** cantidad de CPU por tarea.
 - Por defecto: **--cpus-per-task=1**
- **--gres=gpu:<número>** para solicitar un número de GPUs por nodo. Para solicitar una GPU por nodo: **--gres=gpu:1**

Parámetros: entrada/salida

- **--output=<archivo>** archivo donde se enviará la salida estándar de la ejecución. Por defecto: **slurm-ID.out**
- **--error=<archivo>** archivo donde se enviará la salida de error de la ejecución. Por defecto **--output** contiene ambas salidas.

Variables de entorno en SLURM

SLURM exporta un conjunto de variables que pueden ser usadas durante la ejecución de un trabajo. Por ejemplo:

- ID del trabajo: `$SLURM_JOBID`
- Nombre del trabajo: `$SLURM_JOB_NAME`
- Nombre de la account donde ejecuta el trabajo: `$SLURM_JOB_ACCOUNT`
- Directorio de lanzamiento del trabajo: `$SLURM_SUBMIT_DIR`
- Número de nodos asignados: `$SLURM_NNODES`
- Nombres de los nodos asignados: `$SLURM_JOB_NODELIST`
- Número de cores asignados: `$SLURM_NPROCS`
- Número de tareas creadas: `$SLURM_NTASKS`
- Id de GPU asignada: `$CUDA_VISIBLE_DEVICES`
- Muchas más...

Política de uso

- Cada usuario debe **respaldar** su información.
- Se recomienda no almacenar información *sensible*.
- Los datos y aplicaciones almacenados en el cluster son *responsabilidad* de cada usuario.
- El cluster está destinado a la *investigación e innovación*.

Políticas completas en:

https://www.cluster.uy/ayuda/politica_uso/

Política de uso



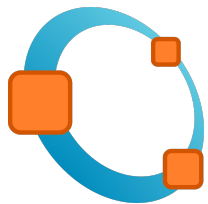
Las ejecuciones en login están PROHIBIDAS. Rogamos solicite un trabajo interactivo para realizar tareas como descomprimir archivos, copiar, editar, instalar o compilar.

¡Cuidado con VSCode!



Visual Studio Code

Aplicaciones disponibles en cluster.uy



Lista (*incompleta*) de aplicaciones en:

https://www.cluster.uy/ayuda/lista_software/