2º curso / 2º cuatr.

Grado en Ing. Informática

Arquitectura de Computadores

Seminario 0. Entorno de programación: atcgrid y gestor de carga de trabajo

Material elaborado por los profesores responsables de la asignatura:
Mancia Anguita – Julio Ortega

ugr Universidad de Granada

ETSIIT Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación

ATC Departamento de Arquitectura y Tecnología de Computadores UNIVERSIDAD DE GRANADA

Contenidos

- Cluster de prácticas (atcgrid)
- Gestor de carga de trabajo (*workload manager*)
- Ejemplo de script

Contenidos

- Cluster de prácticas (atcgrid)
  - Componentes
  - Placa madre
  - *Chips* de procesamiento (procesadores)
  - Acceso
- Gestor de carga de trabajo
- Ejemplo de script

Cluster de prácticas (atcgrid): componentes

Nodos de cómputo (atcgrid[1-4]):

➔ Tres servidores rack SuperMicro SuperServer 6016T-T (atcgrid[1-3])
http://www.supermicro.com/products/system/1U/6016/SYS-6016T-T.cfm

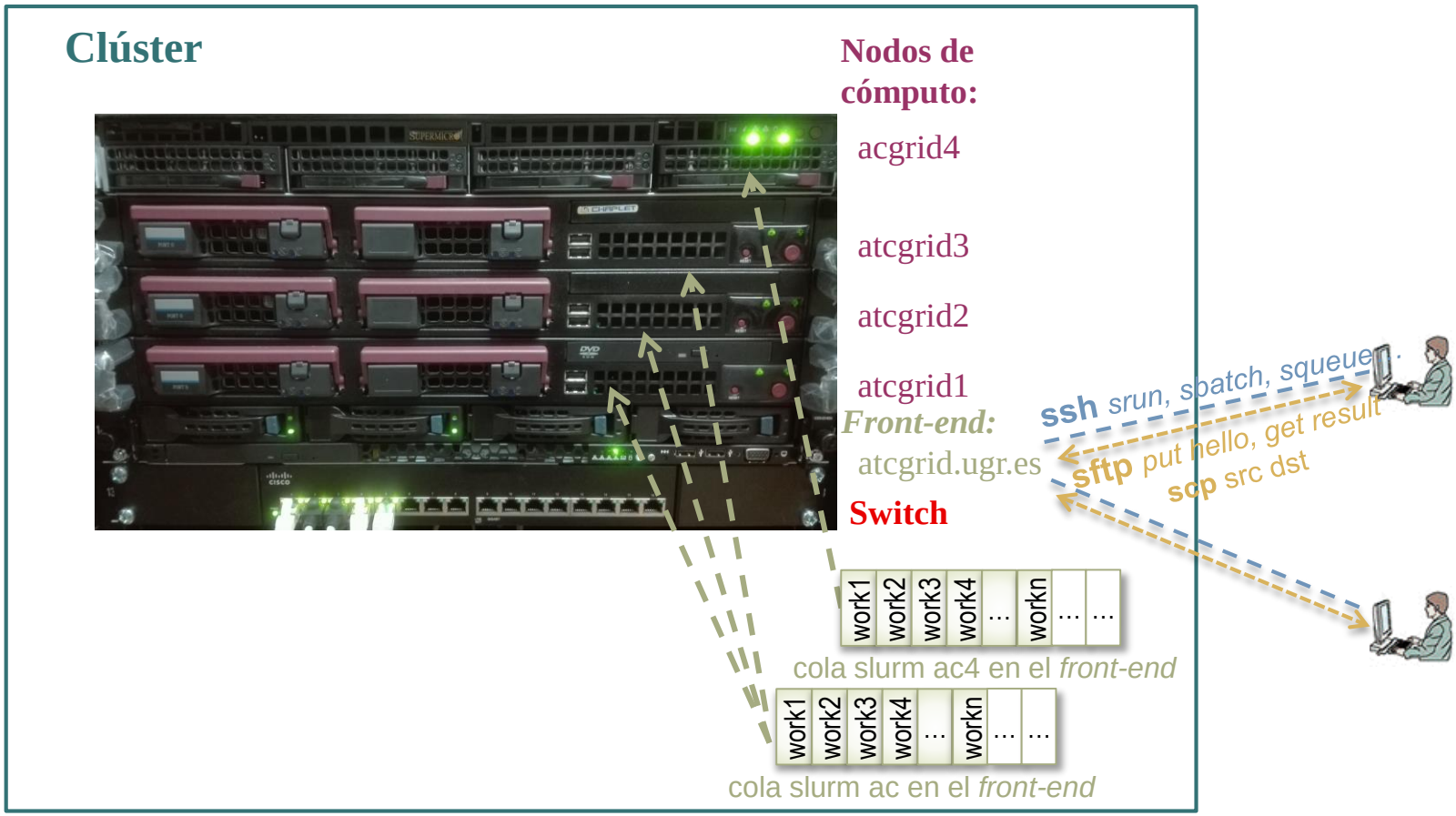
➔ Un servidor rack SuperMicro SYS-6019U-TR4 1U (atcgrid4)
https://www.supermicro.com/products/system/1u/6019/SYS-6019U-TR4.cfm

Cables

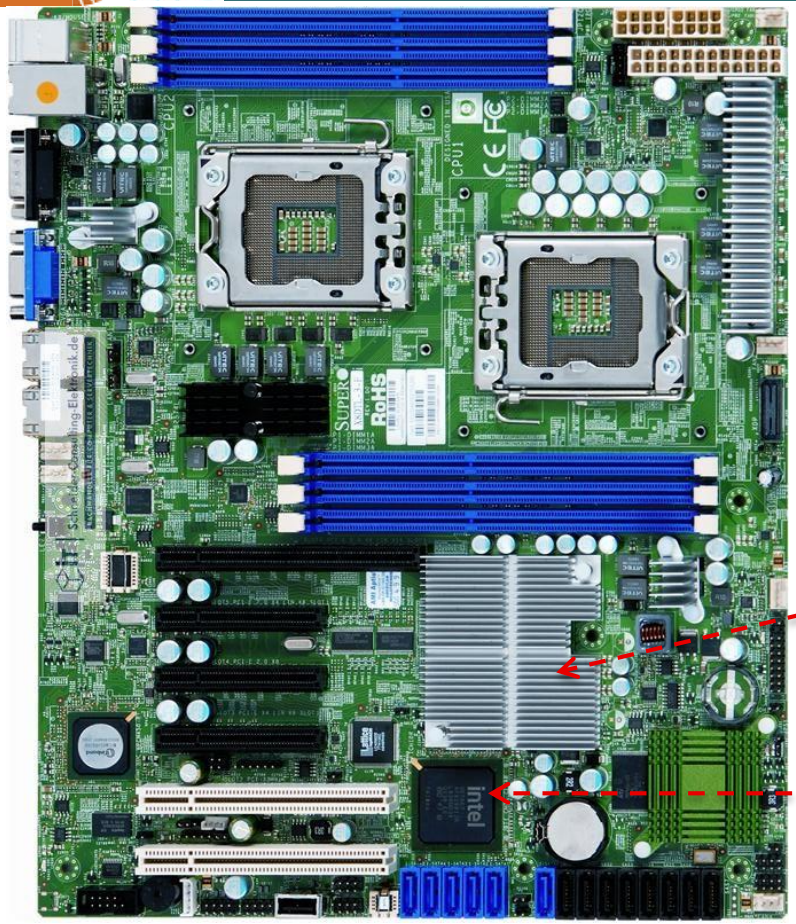
Switch

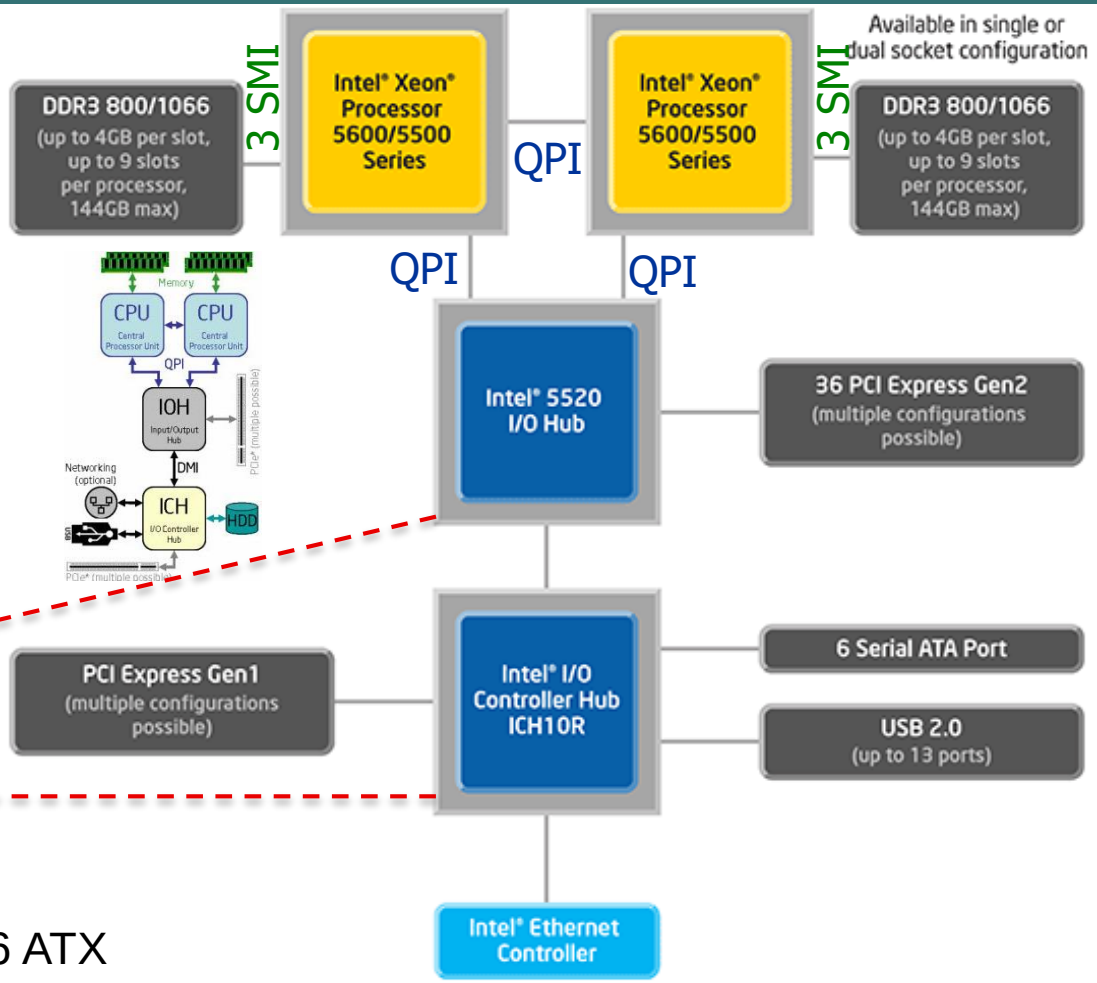
Nodo *front-end* (*host*, *master*):
Asus RS300-E9-PS4

Cluster de prácticas (atcgrid): componentes



Cluster de prácticas>nodos atcgrid[1-3]: placa madre

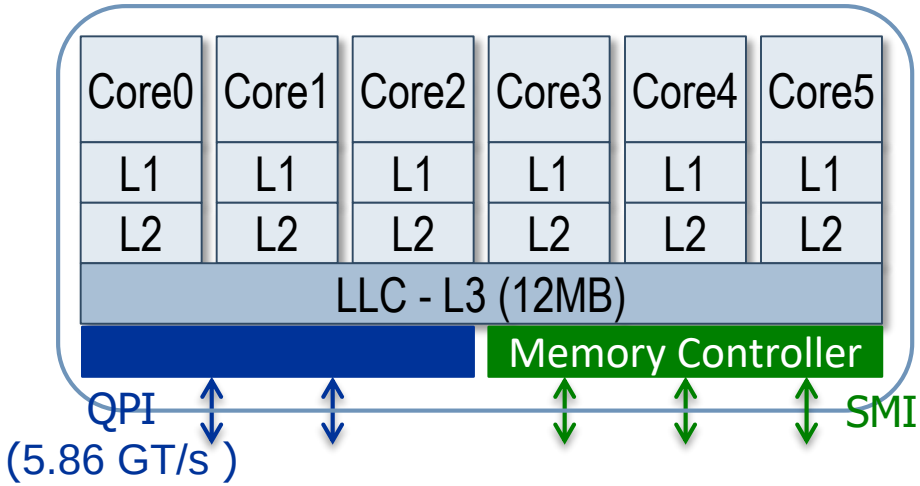




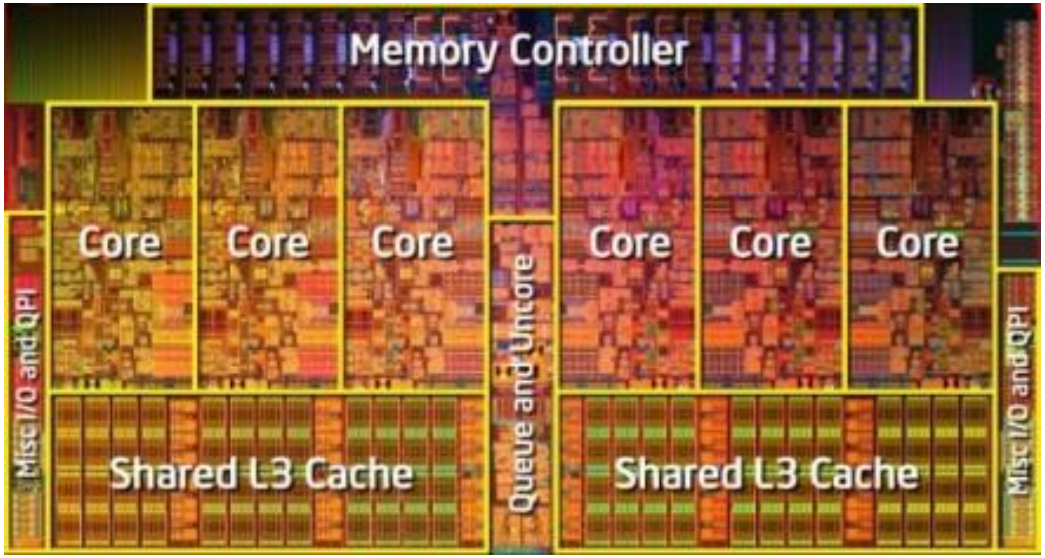
Supermicro X8DTL-i Dual Socket 1366 ATX Server Mainboard Intel 5520 chipset
http://www.supermicro.com/products/motherboard/QPI/5500/X8DTL-i.cfm

Intel® 5520 Chipset
http://www.intel.com/p/en_US/embedded/hwsw/hardware/xeon-5600-5500/overview

Cluster de prácticas>nodos atcgrid[1-3]: chip de procesamiento

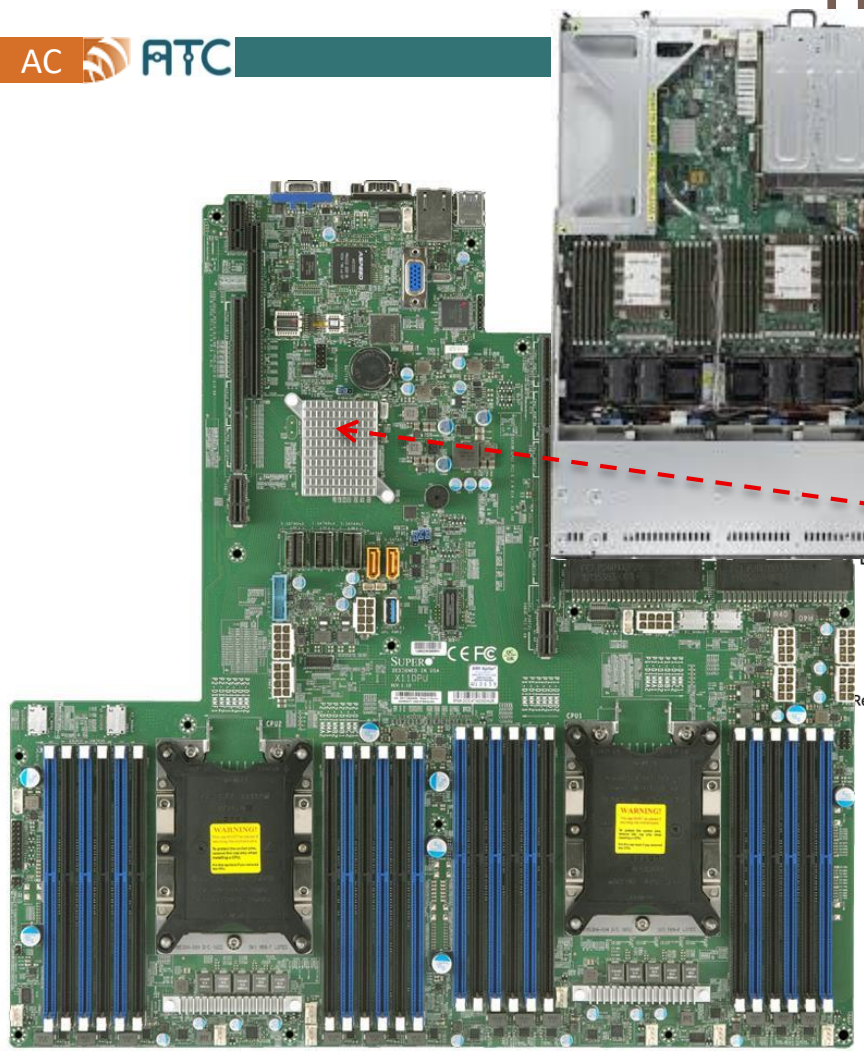


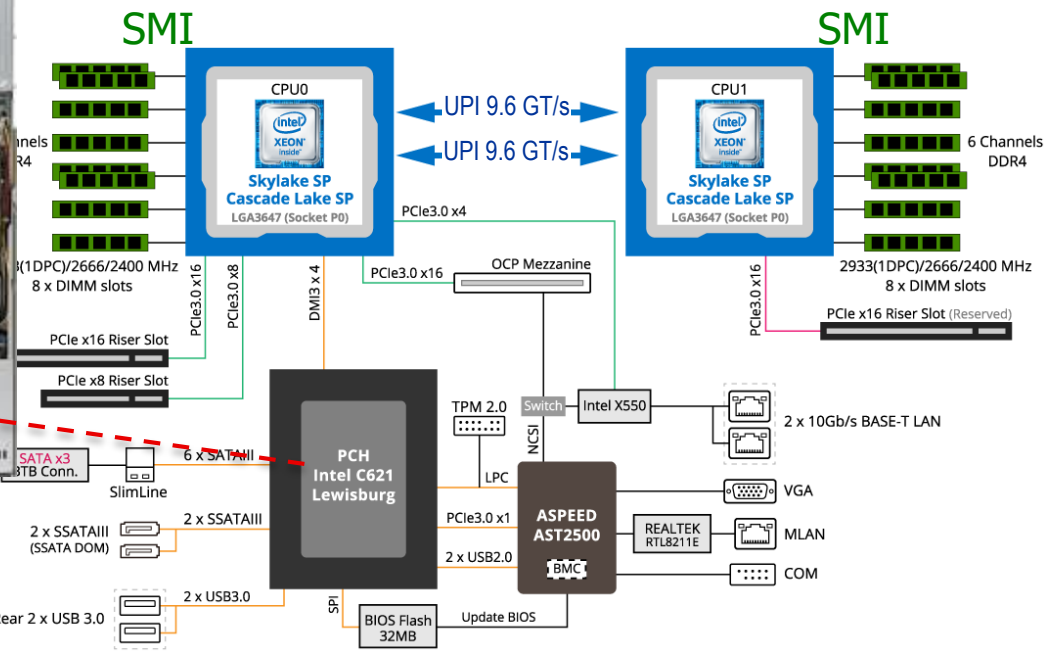
6 cores hyperthreading 2.4 GHz (12 threads)

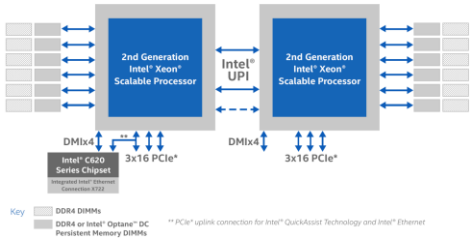


Intel Xeon E5645 (6 cores/12 threads, 12M L3 Cache compartida, 2.40 GHz cada core, 5.86 GT/s Intel® QPI) http://ark.intel.com/products/48768?wapkw=(E5645)

Cluster de prácticas> nodo atcgrid4: placa madre







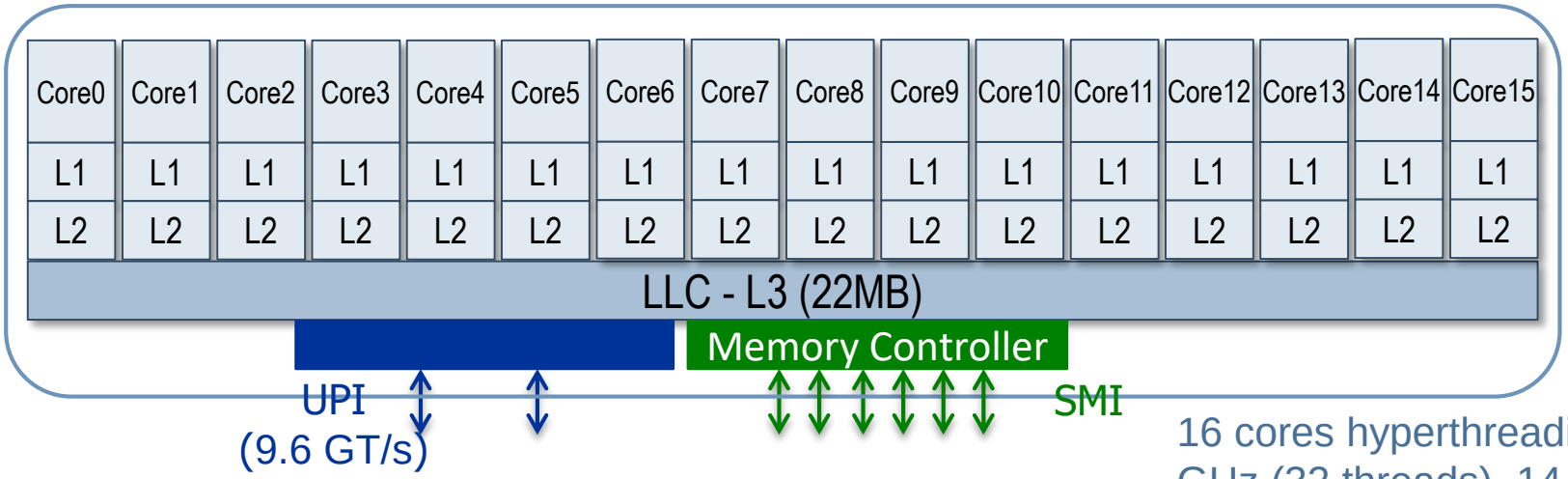
Supermicro SYS-6019U-TR4 1U

https://www.supermicro.com/en/products/motherboard/X11DPU

Intel® C621 Chipset

Intel® C621 Chipset Product Specifications

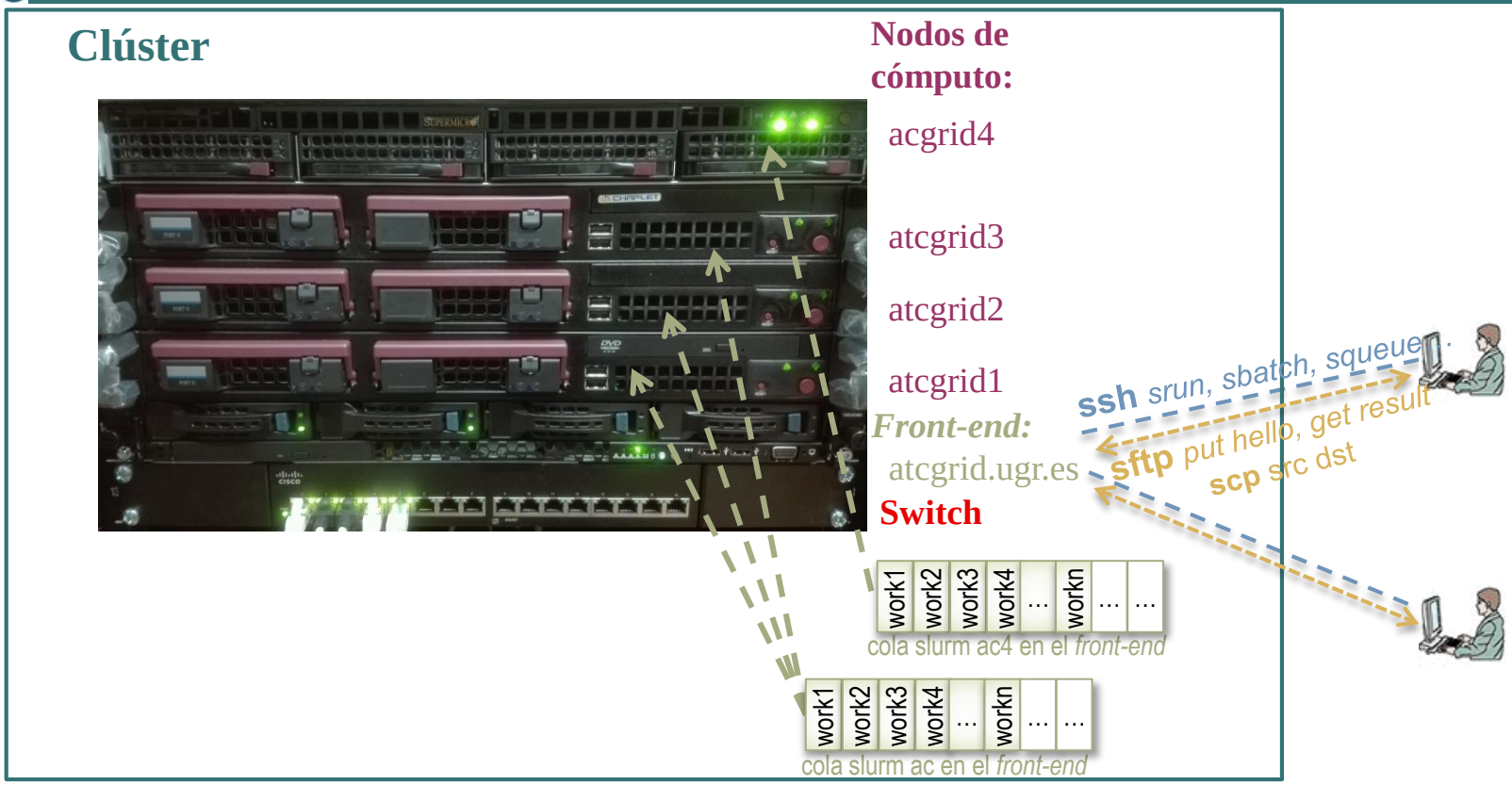
Cluster de prácticas (atcgrid4): chip de procesamiento de la CPU



16 cores hyperthreading 2.1 GHz (32 threads), 14 nm

Intel® Xeon® Silver 4216 (16 cores/32 threads, 22MB L3 Cache compartida, 2.10 GHz máxima 3,2 GHz, cada core, 9.6 GT/s Intel® UPI)

Cluster de prácticas (atcgrid): acceso



- Cada **usuario** tiene un home en el nodo ***front-end*** del **clúster atcgrid**. Se puede acceder al home:
  - Para ejecutar comandos (`srun`, `sbatch`, `squeue`…), con un cliente **`ssh`** (***secure shell***):
    - Linux: `$ ssh -X username@atcgrid.ugr.es` (pide *password* del usuario "username")
  - Para cargar y descargar ficheros (`put hello`, `get slurm-9.out`, …), con un cliente **`sftp`** (***secure file transfer protocol***)
    - Linux: `$ sftp username@atcgrid.ugr.es` (pide *password* del usuario "username")

Contenidos

- Cluster de prácticas (atcgrid)
- Gestor de carga de trabajo
- Ejemplo de script

Ejemplos con comandos slurm

- Se ejecutarán en el *front-end* con conexión **ssh**

Ejemplo	Explicación
srun -pac -Aac ./hello **srun** -p ac4 -A ac lscpu	`srun` envía a ejecutar un trabajo (en los ejemplos, el ejecutable hello, y lscpu) a través de una cola slurm. Si aparece `-p`, se envía a nodos de la cola especificada con `-p` (un trabajo solo puede usar un nodo de la cola `ac`).
sbatch -p ac script.sh **sbatch** -p ac --wrap "echo Hola" **sbatch** -p ac --wrap "./hello" **sbatch** -p ac --wrap "echo Hola ; ./hello"	`sbatch` envía a ejecutar un *script* (en este caso script.sh, "echo Hola", "./hello" y "echo Hola ; ./hello") a través de una cola slurm. La salida se devuelve en un fichero. La ejecución con `srun` es *interactiva*, con `sbatch` es en *segundo plano*. Se recomienda usar `sbatch`
squeue	Muestra todos los trabajos en ejecución y los que están encolados
scancel jobid	Elimina el trabajo con identificador "jobid"
sinfo	Lista información de las particiones (colas) y de los nodos
sinfo -pac -o"%10D %10G %20b %f"	Lista los nodos (D), los recursos (G), y las características activas (b) y disponibles (f) en la partición especificada (-p) (%[[.]size]type[suffix])

Tabla resumen: https://slurm.schedmd.com/pdfs/summary.pdf
Páginas de manual: https://slurm.schedmd.com/man_index.html

Particiones slurm (colas) en atcgrid

AC ATC

```
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
ac*          up       1:00      3   idle atcgrid[1-3]
ac4          up       1:00      1   idle atcgrid4
aapt         up       2:00      3   idle atcgrid[1-3]
acap         up       1:00      3   idle atcgrid[1-3]
```

- * significa que `ac` es la cola utilizada por defecto, es decir, cuando no se usa `-p`

Contenidos

- Cluster de prácticas (atcgrid)
- Gestor de carga de trabajo
- Ejemplo de script

Ejemplo hello OpenMP

- Cada *thread* imprime su identificador
  - El identificador se obtiene con la función OpenMP `omp_get_thread_num()`

HelloOMP.c

```
/* Compilar con:
 gcc -O2 -fopenmp -o HelloOMP HelloOMP.c
*/
#include <stdio.h>
#include <omp.h>

int main(void) {

#pragma omp parallel
  printf("(%d:!!!Hello world!!!)",
            omp_get_thread_num());

return(0);

}
```

Script para la ejecución del ejemplo HelloOMP en atcgrid

script_helloomp.sh

```
#!/bin/bash
#Órdenes para el Gestor de carga de trabajo:
#1. Asigna al trabajo un nombre
#SBATCH --job-name=helloOMP
#2. Asignar el trabajo a una partición (cola)
#SBATCH --partition=ac
#2. Asignar el trabajo a un account
#SBATCH --account=ac

#Obtener información de las variables del entorno del Gestor de carga de trabajo:
echo "Id. usuario del trabajo: $SLURM_JOB_USER"
echo "Id. del trabajo: $SLURM_JOBID"
echo "Nombre del trabajo especificado por usuario: $SLURM_JOB_NAME"
echo "Directorio de trabajo (en el que se ejecuta el script): $SLURM_SUBMIT_DIR"
echo "Cola: $SLURM_JOB_PARTITION"
echo "Nodo que ejecuta este trabajo:$SLURM_SUBMIT_HOST"
echo "Nº de nodos asignados al trabajo: $SLURM_JOB_NUM_NODES"
echo "Nodos asignados al trabajo: $SLURM_JOB_NODELIST"
echo "CPUs por nodo: $SLURM_JOB_CPUS_PER_NODE"
#Instrucciones del script para ejecutar código:
echo -e "\n 1. Ejecución helloOMP una vez sin cambiar nº de threads (valor por defecto):\n"
srun ./HelloOMP
echo -e "\n 2. Ejecución helloOMP varias veces con distinto nº de threads:\n"
for ((P=12;P>0;P=P/2))
do
   export OMP_NUM_THREADS=$P
   echo -e "\n  - Para $P threads:"
   srun ./HelloOMP
done
```

Órdenes para Gestor de carga de trabajo

Para imprimir variables del Gestor de carga de trabajo

Instrucciones del script

No olvidar poner **srun** delante del ejecutable

Utilidades

- Formateo de código a insertar en los cuadernos:
  - https://pinetools.com/syntax-highlighter,
  - https://highlight.hohli.com/index.php,
  - https://tohtml.com/