# VMGRID (ATLAS TIER2 ESPAÑA)

# Uso de recursos HPC para ATLAS Red Española de Supercomputación ** Lusitania **


**Esteban Fullana, Santiago González, Javier Sánchez**

**Instituto de Física Corpuscular (IFIC)**
**Universitat de València - CSIC**

# Red Española de Supercomputación

- Siguiendo el ejemplo de Andreu, Santiago solicitó horas de cálculo en la Red Española de Supercomputación (RES)
- Asignación de tiempo de uso:
  - Andreu
    - BSC (Barcelona)
    - Cibeles (Madrid)
  - Santiago
    - Lusitania (Extremadura)
- No hemos obtenido tiempo en Valencia (Tirant) porque la Universitat de València está reformando su CPD y no está todavía listo

# Concepto general de HPC

- Infraestructura con un número muy alto de procesadores potentes con mucha memoria y un sistema de comunicación de alto ancho de banda y muy baja latencia.
- Optimizadas para cálculo paralelo.
- Suelen tener algunas limitaciones para nuestro uso:
  - Arquitecturas exóticas
  - No tienen almacenamiento local en los nodos
  - No tienen acceso a la red
  - Sistemas operativos distintos a SLC 6

# Lusitania II

- Lusitania es una infraestructura de HPC en extremadura que participa en la RES

  http://www.cenits.es/cenits/lusitania/caracteristicas-lusitania

- Tenemos acceso a
  - 15 nodos
    - 2 x Intel Xeon E5-2660v3 10 cores/c.u.
    - 80 GB RAM
    - RHEL 6.6 (Santiago) 2.6.32-504.8.1.el6.x86_64
  - 5 TB + 5TB (scratch) en disco compartido (Lustre)
  - 2 head nodes para ssh login (bajo un alias único)
  - Sistema de colas SLURM
- Los nodos tiene comunicación externa via NAT

# Primera aproximación

- Puesto que los nodos tienen conectividad externa y tienen instalado RHEL 6, pensamos en cvmfs
  - No podemos montar directamente cvmfs (kernel)
  - Podemos correr en UserSpace => PARROT
- PARROT funciona correctamente pero genera mucho tráfico de red. No disponemos de un SQUID local
  - Aun así pudimos generar MC sin problemas
  - Al uso de la CPU hay que añadir el tiempo de importar los ficheros de datos y programas y el tiempo de exportar el output.
- Pensamos en correr el harvester en un cron en el head-node para la gestión de los trabajos de Panda

# Primera aproximación

- Preocupación de la comunidad HPC
  - No quieren mucho I/O
  - Quieren eficiencias altas en el uso de la CPU
  - Quieren programas optimizados
- Esto nos lleva a una aproximación más clásica:
  - Modo HPC ATLAS puro
  - Todo el sw de ATLAS en una única imagen local
  - La entrada y salida queda fuera del tiempo del trabajo
  - La experiencia y el setup es usable en otros HPCs

# Solución ensayada

- ARC-CE
  - Instalado un servidor ARC-CE en el IFIC
  - Recibe los trabajos de Panda.
  - Obtiene los ficheros de entrada y los scripts y los coloca en el sistema HPC
  - Cuando lo tiene todo envía el trabajo a la cola de la HPC
  - Al final recoge el output del HPC y lo envía a Panda y a los SEs
- Singularity
  - La imagen que contiene el software de ATLAS + SO es de este tipo.
  - Ocupa 450 GB que han de estar accesibles para los trabajos en el sistema HPC

# Singularity

- Permite correr un SO dentro de otro en un entorno aislado

  https://singularity.lbl.gov/

- Este entorno usa los ficheros que tiene dentro de la imagen (container) y puede acceder a algunos directorios externos que previamente han sido ligados
  - Algunos directorios y ficheros son montados o ligados (binding) automáticamente, como por ejemplo:
    - /tmp, /var/tmp
    - $HOME
    - /etc/hosts

# Singularity

- Singularity controla y restringe que puntos son accesibles
  - No puedo montar directorios externos en puntos del contenedor que no existen
- Para crear o modificar las imágenes hay que ser root
- Problema 1:
  - Pedimos que instalaran singularity en Lusitania, pero no funcionaba porque el kernel es antiguo
  - No es posible que actualicen el kernel
  - Tras mucho probar conseguimos parchear una versión de singularity y que la instalasen 4.2.5, eliminando algunos checks de seguridad que no son posibles en esa versión del kernel

# Singularity

- Andreu nos proporcionó una imagen del SW de ATLAS con SLC 6 (450 GB)
- Es necesario modificar la imagen para añadir directorios que no existen o actualizar ficheros importantes:
  - HOME: /lusitania_homes
  - SLURM: /var/spool/slurmd
  - LUSTRE: /lustre
  - Actualizar: /cvmfs/atlas.cern.ch/repo/sw/local/etc/
- Para cada modificación que hay que hacer en el IFIC como root, es necesario transferir la imagen de nuevo (450 GB)
  - Es posible usar blocksync.py que sólo transfiere los cambios, pero es lento porque hay que leer en cada lado 450 GB y comparar checksums

# ARC-CE

- ARC-CE es el sistema empleado para controlar los trabajos y el flujo de ficheros de entrada y salida
- Después de configurar y comprobar que tenemos un ARC-CE funcional podemos integrar el sistema remoto
  - Mapping de usuarios con LCMAPS
  - Certificados, etc
- ARC-CE y Lusitania comparten el directorio “session” mediante “sshfs”. Este directorio pertenece físicamente al almacenamiento de Lusitania, pero es visible desde el ARC-CE.
  - Este directorio ha de ser accesible por el trabajo en el sistema de batch local del HPC

# ARC-CE

- Asegurarse de las opciones de sshfs:
  - allow_other,nonempty,reconnect,ServerAliveInterval=15,ServerAliveCountMax=3
  - Asegurarse que el uid y gid remoto es trasladado al usuario y grupo arc del ARC-CE con las opciones uid=xxxx gid=xxxx
- Para que sea un proceso automático sin password, creamos una clave de ssh y la registramos en el sistema remoto
- No queremos perder la comunicación con el HPC y el punto de montaje de “session”
- Ejemplo:

```
sshfs -o
allow_other,nonempty,reconnect,ServerAliveInterval=15,Server
AliveCountMax=3 -o uid=1000 -o gid=1000
uv23819@lusitania2.cenits.es:/lusitania_homes/uv23/uv23819/a
rc/session /opt/arc/session/
```

# ARC-CE

- Configuramos ARC-CE como si el sistema de batch de Lusitania (SLURM) fuera local
- Creamos las herramientas impostoras de los comandos de SLURM de manera que ejecuten comandos remotos en Lusitania
- La receta no es nuestra, esta copiada de una presentación de CSCS HPC que a su vez lo llama “Bern aproach”
- Creamos un script llamado sshslurm y el resto de comandos SLURM con enlaces simbólicos a éste
- Nos aseguramos que el comando puede ejecutarse sin password mediante la clave de ssh

```
#!/bin/bash
source /opt/sshslurm/config/sshslurm-config
SBINARY=$(basename "$0")
SARGS=""
for token in "$@"; do
   SARGS="$SARGS '$token'"
done
echo $(date) - $SBINARY "$SARGS" >> /tmp/sshslurm.log
POST_EXEC=""
[[ "$SBINARY" == "squeue" ]] && SBINARY="squeue_cenits"
if [[ "$SBINARY" == "sbatch" && "$1" != "" ]]; then
   batchscript=$(mktemp)
   cp "$1" $batchscript
   sed -i 's#/opt/arc/session#/lustre/lusitania_homes/uv23/uv23819/arc/session#ig' $batchscript
   sed -i 's#/opt/arc/runtime#/lustre/lusitania_homes/uv23/uv23819/arc/runtime#ig' $batchscript
   SARGS=$REMOTE_TEMP_PATH/$(basename "$1")
   $SCP_CMDLINE -q $batchscript "$SSHSLURM_HOST:$SARGS"
   POST_EXEC="rm -f $SARGS"
fi
$SSH_CMDLINE $SSHSLURM_HOST -- $REMOTE_SLURM_PATH/$SBINARY "$SARGS" \; $POST_EXEC
```

# ARC-CE

- Ahora ARC-CE debería de ser capaz de enviar trabajos a lusitania y recibir el ouput.
- El flujo a grandes rasgos es:
  - El cliente envía el trabajo al ARC-CE
  - El ARC-CE prepara el trabajo en un directorio de “session” obteniendo los ficheros de input
  - El ARC-CE submite el trabajo al sistema BATCH local, pero en realidad es un ssh que ejecuta el comando en el sistema remoto
  - El sistema SLURM de lusitania ejecuta el trabajo que tiene sus scripts en “session”
  - El ARC-CE pregunta al sistema local de colas por el estado del trabajo, pero en realidad lo esta haciendo en el sistema remoto por ssh
  - Cuando acaba el trabajo, el output que está en session es “local” para el ARC-CE al estar montado por sshfs

# ARC-CE

- Con algún detalle más ya está preparado el sistema para recibir trabajos de ATLAS.
- Añadir un directorio “runtime” que puede ser compartido entre el ARC-CE y Lusitania o simplemente copiar el contenido ya que éste es estático:

  arc/runtime/APPS/HEP/ATLAS-SITE

- Contiene variables de entorno necesarias para los trabajos de ATLAS
- Lo siguiente es que los trabajos en Lusitania se ejecuten dentro de singularity para que dispongan del entorno de SW apropiado para ATLAS

# ARC-CE

- Para ello, se modifica el script que genera los trabajos para SLURM del ARC-CE /usr/share/arc/submit-SLURM-job añadiendo algo como:

```
# force execution in container

echo 'SINGULARITY_IMAGE="$HOME/singularity/centos6-cvmfs-
v2.atlas.cern.ch.img"' >> $LRMS_JOB_SCRIPT

echo 'SINGULARITY_BIND="-B /lustre,/var/slurm/"'      >> $LRMS_JOB_SCRIPT

echo 'if [ -z $SINGULARITY_CONTAINER ]; then'         >> $LRMS_JOB_SCRIPT

echo '  exec /usr/bin/singularity exec $SINGULARITY_BIND $SINGULARITY_IMAGE
$0'  >> $LRMS_JOB_SCRIPT

echo 'fi'                                             >> $LRMS_JOB_SCRIPT
```

- De forma que el trabajo se ejecuta dentro de singularity desde el primer momento

# ARC-CE

- Una vez que el entorno de ARC-CE – SLURM – Singularity funciona, es el momento de crear una cola de test de PANDA en AGIS
  - IFIC_ARC_TEST
- Ajustar todos los parámetros de forma correcta en especial los “file movers”
- Después de actualizar AGIS, hay que refrescar dentro de la imagen el directorio:

  /cvmfs/atlas.cern.ch/repo/sw/local/etc/

  Asegurándose que la fuente en /cvmfs real está a su vez actualizada.
- Esperar los trabajos de HammerCloud y verificar que se ejecutan correctamente

# Status

- Ya tenemos la cadena completa
- Algunos trabajos fallan porque:
  - No existe en la imagen de Singularity la release pedida.
  - No existe en la imagen la versión de gcc pedida
  - Son trabajos de derivation
  - Hay algún problema en la trasferencia de los ficheros o en el ARC-CE
- Ya tenemos muchos trabajos que se ejecutan bien
- Estamos muy cerca que tener el sistema listo para la producción de MC en lusitania.