Batch Service at CERN

- Service used for both grid and local submissions, HPC on the way
- Local public queue open to all CERN users
- Wider range of requirements than grid submissions
- Migration to HTCondor underway, majority still LSF
  - Some Grid migrated to HTCondor (see Iain’s talk)
## LSF likes/dislikes

<table>
<thead>
<tr>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Defined queues based on job length</td>
<td>Slow reconfigure to add/remove machines</td>
</tr>
<tr>
<td>Ability to backfill whilst draining</td>
<td>Limit of 6500 nodes</td>
</tr>
<tr>
<td>Ability to encourage shorter jobs</td>
<td>Slow query / submission</td>
</tr>
</tbody>
</table>
Current Capacity

~80K cores vs ~10K cores

~5.4K nodes vs ~1.4K nodes

HTCondor nodes 8 core, vs most LSF are 16 core
Some background

- 400-600K job submissions per day
- ~60K running jobs, ~110K pending
- “local” submissions range between 40-60% of total
- Primarily Vanilla Universe
Some background

- 400-600K job submissions per day
- ~60K running jobs, ~110K pending
  - Implies around 10-15 schedds
- “local” submissions range between 40-60% of total
  - Local submissions less predictable
- Primarily Vanilla Universe
  - Planning on Docker universe
  - Parallel universe to be evaluated
Schedds

- Need to map users to schedds (currently)
- Want to make it easy & cheap to query, so needs to be static assignment
- Currently using zookeeper as the k/v store
  - Previous experience with zk, we like the availability, has kerberos support
- znode contains schedd
  - `/htcondor/users/$username/{current,old}`
- Old kept for scenarios where we migrate
Scheddds

• LOCAL_CONFIG_FILE with piped script to contact zk on submit/query for schedd
• Remapping of users to scheddds for failures, also to rebalance heavy users
• Specific scheddds for DAG
• Trying to decouple use of $HOME for local batch users
  • Starting to migrate away from AFS homedirs
  • File stage in/out necessary for cloud resources
  • May provide our own stage in/out from CEPH S3 and/or EOS
Kerberos

• Local batch users use/expect kerberos
  • Submission, authentication with other services (yes, including AFS)
• Existing CERN service to renew kerberos tickets for job lifetime
• Testing HTCondor integration via Credential Monitor
  • Touchpoints with condor_submit, condor_credd to maintain, condor_starter to copy to sandbox
Accounting Group management

- Preferred workflow: resource coordinator sets overall shares, experiments manage lower down
- Using Group Quota tool from BNL (github.com/fubarwrangler/group-quota)
- Extending to provide REST service to add/remove group membership
- SUBMIT_REQUIREMENT checks if user authorized for a group
# EZEditor of group

## EZEditor for group_u_ALICE

<table>
<thead>
<tr>
<th>Name</th>
<th>Fix</th>
<th>Quota</th>
<th>Slider</th>
</tr>
</thead>
<tbody>
<tr>
<td>group_u_ALICE.u_z2</td>
<td></td>
<td>24</td>
<td>0</td>
</tr>
<tr>
<td>group_u_ALICE.grid_ALICESGM</td>
<td></td>
<td>35</td>
<td>0</td>
</tr>
<tr>
<td>group_u_ALICE.grid_ALICEPLT</td>
<td></td>
<td>20</td>
<td>0</td>
</tr>
<tr>
<td>group_u_ALICE.grid_ALICEPRD</td>
<td></td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>group_u_ALICE.grid_ALICE</td>
<td></td>
<td>20</td>
<td>0</td>
</tr>
</tbody>
</table>

- Move the sliders above to adjust what proportion of the total quota each group gets. Select the Fix box to fix that group's quota at the current value (you can't leave fewer than 2 free sliders). Click on the quota field or the *edit* pencil next to it to manually input a quota — the up and down arrows increment and decrement by one while hitting enter or clicking off sets the value.
Jobflavours

- Current LSF service has defined queues for local
  - Defined as “normalised” minutes/hours/weeks
  - Slot limits for individual queues (ie limit long queues)
- Use SYSTEM_PERIODIC_REMOVE & Classads to achieve similar with HTCondor
- Try to keep it reasonably simple for users, and easy for admins to manage
JobFlavours

- Instead of 8nm, 1nh, 8nh, 1nw, 2nw… espresso, lunchbreak, mañana, nextweek
- Either way “normalised” time is hard for users to understand (“why was my job killed?)
- Job Classad for JobFeature, SYSTEM_PERIODIC_REMOVE to kill jobs over threshold
- Machine Classad for which JobFlavours to accept
Espresso JobFlavour eg

## Espresso Definitions

# Remove espresso jobs after 600 seconds wallclock
Remove_Espresso_WallClock = ((JobFlavour == "espresso")
&& (RemoteWallClockTime >= 600))

# Remove espresso jobs after 500MB Resident Set Size used
Remove_Espresso_RSS = ((JobFlavour == "espresso") &&
(ResidentSetSize >= 488281))

# Combine Expressions
Remove_Espresso_Constraints = $(Remove_Espresso_WallClock) || $(Remove_Espresso_RSS)
SYSTEM_PERIODIC_REMOVE = $(Remove_Espresso_Constraints)
Draining with backfill

- Necessary when deleting / rebooting worker nodes
- Whilst waiting for long jobs to finish, backfill with shorter jobs

BackfillDraining = False
UnixShutdownTime = 0

STARTD_ATTRS = $(STARTD_ATTRS), BackfillDraining, UnixShutdownTime
STARTD.SETTABLE_ATTRS_ADMINISTRATOR = $(STARTD.SETTABLE_ATTRS_ADMINISTRATOR), BackfillDraining, UnixShutdownTime
ENABLE_PERSISTENT_CONFIG = TRUE
PERSISTENT_CONFIG_DIR = /etc/condor/persistent

START = $(START) && ((BackfillDraining ||= False) || (BackfillDraining ||= True && (time() + ExpectedJobTime) <= UnixShutdownTime))
Docker

- Some user enquiries already
- Nascent CERN registry
  - docker.cern.ch
- Plan to use docker universe when some CentOS 7 worker nodes
- Possible extra tool for heterogeneous worker pools (alongside compat)