Migrating from CREAM-CE/LSF to HTCondor-CE/HTCondor

by Stefano Dal Pra

CDG 22 Nov 2019,

Email: stefano.dalpra@cnaf.infn.it

INFN-T1, Current Status (Production)

- $\sim$ 400 KHS06, 35000 slots, 850 physical hosts
- $5 \times \mathrm{CREAM-CE/LSF}\ 9.1.3$
- ~ 40 User groups: 24 Grid VOs, ~ 25 local

Moving to HTC-CE/HTC

We have two HTC clusters right now

Testbed (HTC-CE 3.2.2, HTC 8.8.4)

- $1 \times$ HTC-CE, 1 SN $+1 \times$ CM, $3 \times$ WN, 16 slot each
- Run pilots from the 4 LHC experiments (Sep. 2018)
- Testing configurations / script / tools / manegement

Production (HTC-CE 3.2.2, HTC 8.8.4)

- $(2+1) \times$ HTC-CE, $1 \times$ CM, $68 \times$ WN, 16 slot each (1088 slots, 1.1 KHS06)
- One more WN, with $2 \times$ K-40 GPUs (to test Grid access through HTC-CE)
- $1 \times$ SN for Remote Submission (from local UI, `FS_REMOTE`)

started on May 2019. Gradually moving WNs from LSF to HTCondor.

Local users "only" should convert `bsub` to `condor_submit` from their UI.

1. LHC VOs: Ready
2. Grid VOs using a WMS (i.e. Dirac): Ready
3. Other VOs: In progress
4. Local submitters: In progress

Experience with HTC-CE

Early installations (Apr 2018) a bit tricky (rpm and docs OSG oriented, no automated setup). Help and assistance provided by the HTCondor mailing list and developers.

Now: improved docs and rpms (for non osg people)

- https://htcondor-ce.org
- `htcondor-ce-*` RPMs from the same repository of HTCondor
- https://github.com/cernops/puppet-htcondor_ce. Puppet modules from CERN

Troubleshooting and quirks

- `ui-htc ~]$ condor_ce_trace --debug ce01-htc` (Most frequent: GSI)
- Puppet modules not directly compliant with our puppet/foreman system

On first setup, main issues to deal with were about GSI auth* and, later, GIP.

Note: Mostly because the CE setup was done manually (manual CREAM setup wouldn't have beeen any easier)

voms. The same as with CREAM-CE, except for default name and location of a few files (voms-mapfile, `x509` host certificates)

condor-mapfile. Adding regexp to map allowed certs at your site

After this, the CE should be able to deal with first job submissions

Argus. Set up one or configure an existing one. (Note: no support for TLS1.2)

bdii. two configuration files from `htcondor-ce-bdii` rpm

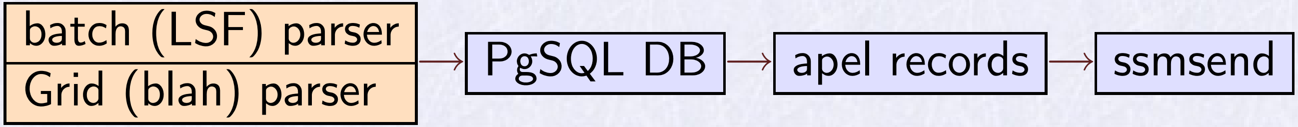
Note: they are in the condor config dir, not condor-ce. Glue2 only.

Notes collected for interested INFN sites: http://wiki.infn.it/progetti/htcondor-tf/home

Accounting

We use a custom accounting system for CREAM-CE/LSF (no APEL). We did some work to adapt it

Accounting with LSF

- 

- User DN and FQAN are the main Grid-side info collected from Blah.
- We collect a few more data for internal use: job exit status, WN name (this is then mapped to HS06 of the node), job exit status,...
- We need to collect the same data from HTC-CE, then we can re-use the other components.

Accounting with HTC-CE

Initially: python bindings to query HTC for job history. It works, but a few timeouts were experienced. Defining `PER_JOB_HISTORY_DIR` turns out to be a safer choice.

- `PER_JOB_HISTORY_DIR=/var/lib/gratia/data/`
- One accounting text file per job, `history.<jobid>` with <key> = <value> pairs, one per line.
- Each file is *complete* (have both grid and batch data: no need for blah records, no need to lookup for matches between sets of grid and batch records).
- python: `jobfile2dict(fn)` read a log file into a python dict. We transform the wanted ones and INSERT INTO our accounting DB. We collect the same set of keys that the apel HTCondor parser collects, and a few more for internal use.
- After parse & insert, the file is archived to a backup directory (prevents double counting, enable further and deeper inspection, in case of doubts).

Apel records obtained as a SQL VIEW:

```
acct=> select * from apelhtjob where "Processors"=8 limit 1;
+-----------------------------------------------------------
Site                          | INFN-T1
SubmitHost                    | ce02-htc.cr.cnaf.infn.it#7737.0#1555345220
MachineName                   | htc-2.cr.cnaf.infn.it
Queue                         | cms
LocalJobId                    | 7737
LocalUserId                   | pilcms006
GlobalUserName                | /[...]CN=cmspilot04/vocms080.cern.ch
FQAN                          | /cms/Role=pilot/Capability=NULL
VO                            | cms
VOGroup                       | /cms
VORole                        | Role=pilot
WallDuration                  | 41848
CpuDuration                   | 40549
Processors                    | 8
NodeCount                     | 1
StartTime                     | 1555345239
EndTime                       | 1555350470
InfrastructureDescription     | APEL-HTC-CE
InfrastructureType            | grid
ServiceLevelType              | HEPSPEC
ServiceLevel                  | 10.000
```

Enabling GPU access to Grid users

Successful usage tests from ATLAS and VIRGO

At client side: In the condor submit file:

```
request_GPUs = 1
requirements = (target.CUDACapability >= 1.2) &&\
(target.CUDADeviceName =?= "Tesla K40m") &&\
$(requirements:True)
```

HTC-CE side: In the HTCondor-CE `JOB_ROUTER_ENTRIES`:

```
[name = "condor_pool_atlas_cuda";
 TargetUniverse = 5;
 Requirements = (...) && (target.queue =?= "atlas_cuda");
 eval_set_WantGPU = true;
 set_requirements = (target.HasGPU =?= true);
...]
```

Accounting GPU usage

there are keys in the job history file:

```
AssignedGPUs = "CUDA0"
GPUsProvisioned=1
```

Tracking these (or newer) keys

```
acct=> SELECT COUNT(*) AS "N", sum(runtime) as "WCT", username, exechosts
acct-> FROM htjob WHERE gpu=1 GROUP BY username,exechosts;
```

Which yields:

N	WCT	username	exechosts
5	4	atlas220	hpc-200-06-07
5	5	dteam039	hpc-200-06-07
3	3	sdalpra	hpc-200-06-07
17	28	virgo050	hpc-200-06-07

Managing HTCondor

- LSF: configure everything on a small set of files
- Puppet + Foreman: provisioning and main setup. Good for semi-static *known to work* configurations. Not easy to achieve a desired level of flexibility (example: temporarily excluding a VO from working on an arbitrary set of WNs)
- `htconf.py`: simple tool to override puppet settings and provide granularity. Makes use of a shared filesystem across machines in the pool.

`/shared/fs/htconf.py|` declared in a main HTCondor config file. It injects a set of knobs to the machine running it, depending on the *role*, *group* and *name* of the machine.

- More similar to the way LSF is configured (a small set of config files)
- several different configurations can be tested and compared quickly
- Example: temporarily adding a classAd to an arbitrary set of WNs is a matter of defining the hostnames in a file and the classad in a related file

Command line tools

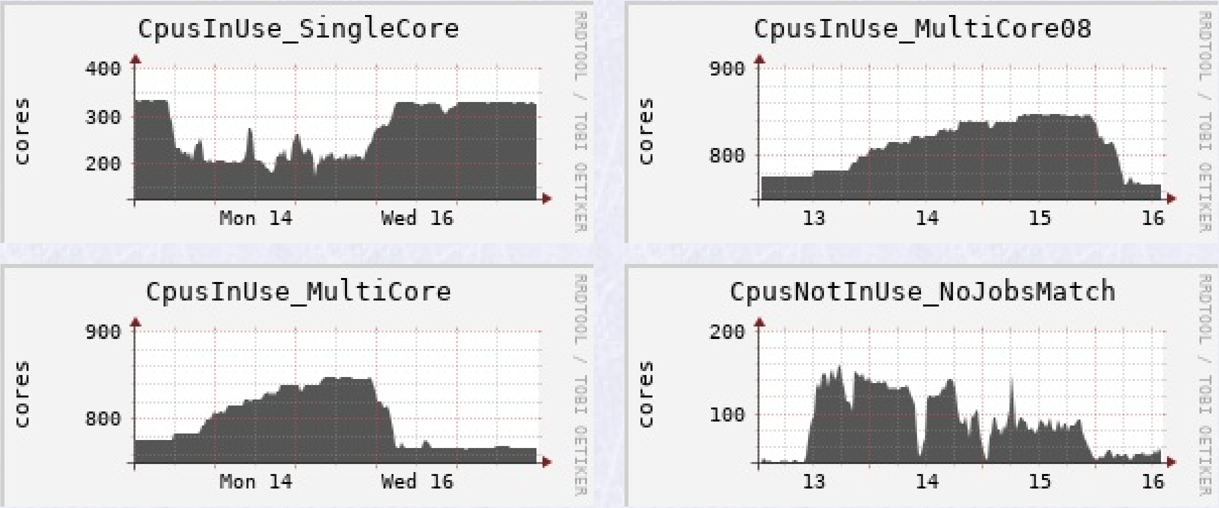
- condor_status, condor_q extremely powerful to inspect job and pool status, yet easy to get cumbersome. Most frequent LSF commands are being emulated using python bindings:

LSF	HTC
bjobs	hjobs
bqueues	hqueues
bhosts	hhosts

```
[root@htc-2 ~]# hjobs.py
JobId RemoteOwner fromhost JobStart Cpus Machine TotalCpus CPUsUsage
25571.0 pagnes sn-01 2019-10-31:03:32:29 1 wn-201-07-15-01-a 16.0 0.99
25764.0 pagnes sn-01 2019-10-31:03:42:55 1 wn-201-07-37-04-a 16.0 0.97
```

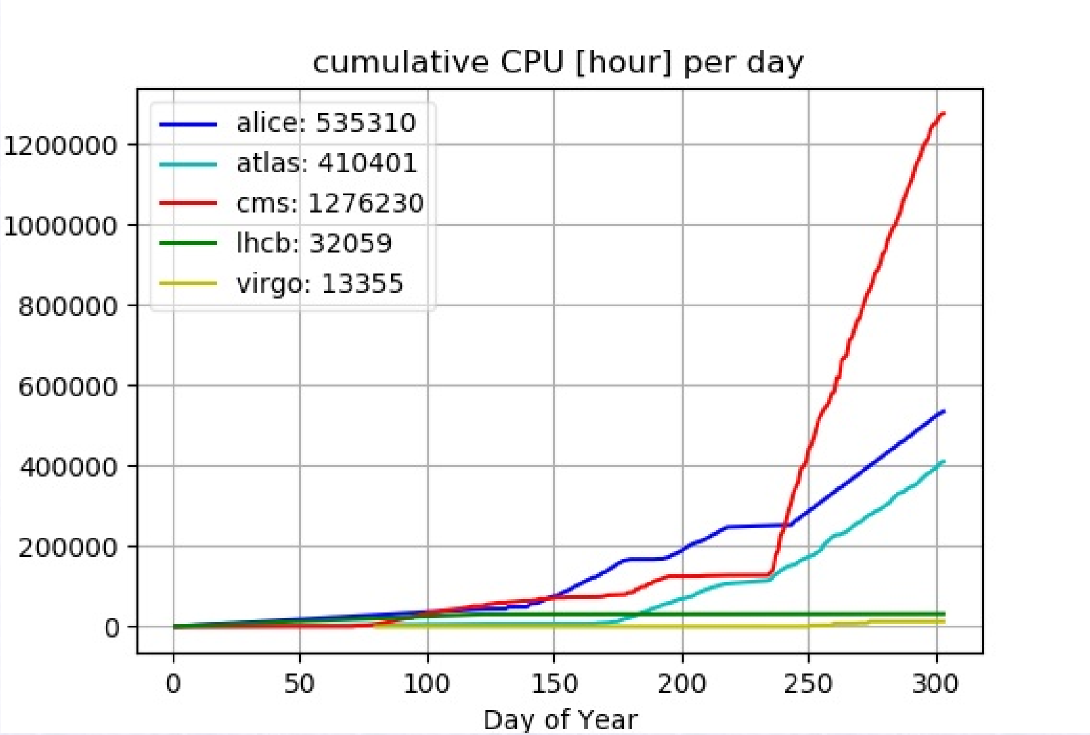
- dump_htc_shares.py injects GROUP_QUOTA_DYNAMIC_group_<name> values in the HTC conf. based on HS06 pledges of the user groups.

Monitoring (Ganglia)



- metrics defined in `/etc/condor/ganglia.d/` and collected by `gmetad` in the `CM`.
- In progress: integrate these data with our Monitoring System (Graphana / Sensu)
- HTCondor-CE comes with a small web tool (CEView) providing a simple interface for monitoring its activity.

JOBS so far...



Going to move more WNs soon and enable more VOs

Userland

- user-support group to assist local users to convert their `bsub` based scripts to use `condor_submit`.
- One common schedd for all local submissions (~ 10% of total)

Extension to remote/heterogenous/dynamic resources

- Dynamic Extension of INFN-T1 to opportunistic resources from several Cloud providers was realized over the past years using LSF [ISGC-2016, vol 13, nr. 18]
- A HTC-CE instance was setup recently and proved succesfull at interfacing with HPC resources (SLURM) to run typical HEP/LHC payloads (cfr. indi.to/Qn4Gf)
- Integration with dynamically provided HTC and HPC resources has also been exploited and is progressing (cfr. ndi.to/rpfVf)

Conclusion

- Gradually migrating most of our WNs from LSF to HTC (allow small groups to take their time).
- Testbed cluster precious to validate operations or troubleshoot problems.
- Monitoring enough for admin purposes. Some more effort required ($\rightarrow$grafana).
- HTC-CE proved versatility. We can migrate our usecases whilst delivering new ones (Grid access to GPUs, HPC clusters,...).
- Custom accounting can track usage of "new" resources (GPUs)
- More HTC-CEs being added as workload grows
- italian ml https://lists.infn.it/sympa/info/htcondor-support (Best Effort)
- Wiki: http://wiki.infn.it/progetti/htcondor-tf/ (Best Effort)

Backup slides

- Avvicendamento LSF $\rightarrow$ HTC in corso. Compresenza dei due sistemi per alcuni mesi.
- Supporto HTC: ml internazionale (altri siti, sviluppatori, altri) o “Task force locale”. Risposte e contributi utili, ma “Best Effort”, Response Time $w \in \mathcal{E}(\lambda) \cup \{+\infty\}$, $\lambda \sim O(\mathrm{days})$.
- $\sum_i \langle \vec{u}_i, \vec{R}\mathrm{isorsa}_i \rangle \cdot \Delta T_i = \mathrm{Percorso}$