Da LSF a HTCondor, da Cream-CE a HTCondor-CE

di Stefano Dal Pra

2019/06/06

Email: stefano.dalpra@cnaf.infn.it

INFN-T1, Situazione corrente

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

Passaggio a HTCondor-CE/HTCondor

Abbiamo due cluster attivi

Testbed cluster (HTC-CE 3.1.0, HTC 8.6.13; in aggiornamento)

- 1$\times$HTC-CE, 1$\times$CM/Collector, 3 $\times$ WN $\times$ 16 slot
- Lavora con job sottomessi dai 4 esperimenti LHC da Sep. 2018
- Una volta configurato: stabile, lavora senza richiedere interventi.

Preprod cluster (HTC-CE 3.2.1, HTC 8.8.2)

- 3×HTC-CE, 1×CM/Collector, 15 × WN × 16 slot
- un altro WN, con 2×K-40 GPUs (test in corso con VIRGO, ATLAS)
- latest stable HTCondor versions (migliore supporto GPU)

Abbiamo job di produzione dal 20 Maggio, e prevediamo di espandere il cluster HTC secondo quest'ordine:

1. VO LHC (ALICE, CMS, ATLAS, LHCb)
2. VO Grid che usano un WMS (es. Dirac)
3. Altre VO e utenti locali
4. Compresenza di LSF e HTC, graduale "trasloco" dei WN

Esperienza con HTC-CE

Installazione e setup iniziale

- `htcondor-ce-*` gli RPMs ora si trovano nello stesso repository di HTCondor
- https://github.com/cernops/puppet-htcondor_ce. Primi moduli puppet resi disponibili dal CERN (latest commit Dec. 2016)
- https://opensciencegrid.org/docs/compute-element/install-htcondor-ce/

  Documentazione e guide (abbastanza ben fatta e chiara, ma OSG–oriented)

Prima installazione

L'installazione iniziale è proceduta inizialmente per "prove ed errori". Aiuto, suggerimenti e assistenza sono stati forniti attraverso la HTCondor mailing list (anche dagli stessi sviluppatori).

– Moduli puppet condivisi dal CERN non direttamente applicabili alla nostra istanza puppet/foreman (sono stati integrati successivamente)

– La documentazione ufficiale fa riferimento a un tool `osg-configure` per completare il setup, che non ha corrispondente per siti non OSG.

- La configurazione è stata completata manualmente
- `ui-htc ~]$ condor_ce_trace --debug ce01-htc`

  Primo tool per troubleshooting di base.

principalmente si è trattato di sistemare la parte di GSI auth* e, successivamente, Grid Info Provider (GIP).

lcmaps, voms. Come per i CREAM-CE, tranne per il default name e path di alcuni files (voms-mapfile, x509 host certificates)

condor-mapfile. Aggiungere una regexp per il match dei certificati del sito

Argus. Configurarne uno esistente o installarne uno.

A questo punto il CE dovrebbe poter accettare le prime sottomissioni di job

bdii. due file di configurazione da `htcondor-ce-bdii.*.rpm`

NB: sono files nella condor config dir, non condor-ce. Glue2 only.

Dettagli raccolti qui: http://wiki.infn.it/progetti/htcondor-tf/

Esempi

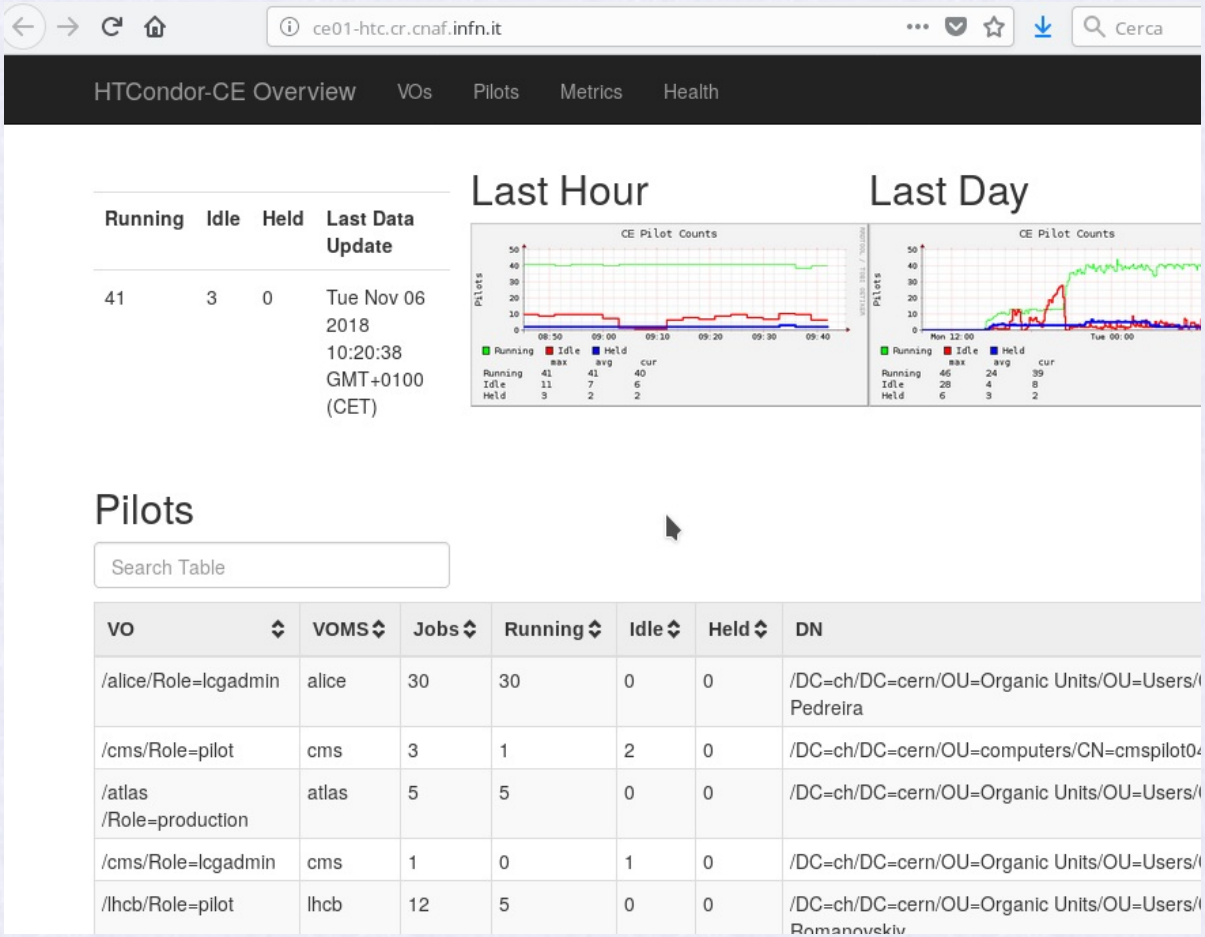
Da una UI con `condor_submit` e con un voms-proxy valido:

```
~]$ condor_submit -pool ce03-htc:9619 -remote ce03-htc -spool cetest.sub

Submitting job(s).

1 job(s) submitted to cluster 27416.

~]$ condor_q -name ce03-htc.cr.cnaf.infn.it -pool ce03-
htc.cr.cnaf.infn.it:9619 27416

-- Schedd: ce03-htc.cr.cnaf.infn.it : <131.154.192.51:9619?... @ 06/05/19
12:24:05

OWNER    BATCH_NAME      SUBMITTED   DONE   RUN    IDLE   HOLD TOTAL
JOB_IDS

dteam039 CMD: htcp308   6/5  12:22      _      _      _      1 1 27397.0

1 jobs; 0 completed, 0 removed, 0 idle, 0 running, 1 held, 0 suspended

~]$ condor_transfer_data -name ce03-htc -pool ce03-htc:9619 27377

Fetching data files...
```

Monitoring

Con HTCondor-CE è fornito built–in un semplice web tool (CEView) che espone un'interfaccia per il monitoring dell'attività of the CE.

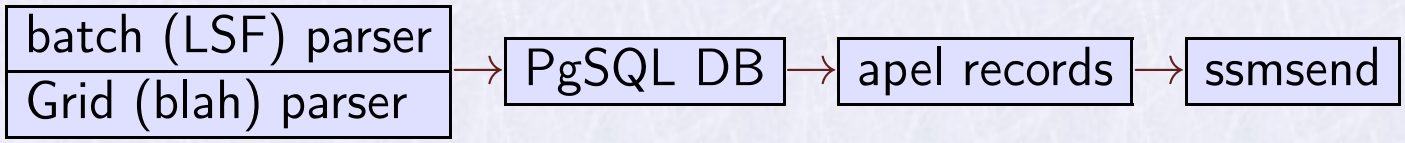
Si abilita editando `05-ce-view.conf`.



Accounting

Al CNAF usiamo da fine 2013 una soluzione custom. Abbiamo cercato di mantenerla col passaggio a HTCondor e HTCondor-CE

Accounting con LSF

- 

- Rispetto a APEL, raccogliamo piú dati per uso interno: job exit status, nome del WN (poi mappato sugli HS06 del nodo), risorse richieste, ...
- Se possiamo raccogliere gli stessi dati dal HTC-CE, possiamo mantenere tutti gli altri componenti.

Accounting con HTC-CE

Primo tentativo: HTCondor python bindings, come alternativa a `condor_history`. Funziona, ma ci sono stati a volte timeout. Definire `PER_JOB_HISTORY_DIR` sembra la cosa giusta da fare (ci ispiriamo al principio KISCS).

- `PER_JOB_HISTORY_DIR=/var/lib/gratia/data/`
- HTCondor crea un text file per job, di nome `history.<jobid>` contenente coppie <key> = <value>, una per linea.
- Ogni file è completo: contiene sia i dati grid che quelli batch. Non servono record blah, non serve cercare corrispondenze tra insiemi di record grid e batch.
- Con python: `jobfile2dict(fn)` legge un file e rende un dict. Le coppie k,v di interesse sono usate per creare la query per l'INSERT della tupla nel DB di accounting. Raccogliamo lo stesso set di chiavi (Job ClassAds in slang HTCondor) che raccoglie il parser HTCondor di apel, piú le extra di nostro interesse.
- Dopo parsing e insert, i files vengono spostati in una backup directory (previene double counting).

Apel records come 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
```

Accesso GPUs via Grid

Client side:

Nel condor submit file:

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

HTC-CE side:

Si configura JOB_ROUTER_ENTRIES nel HTCondor-CE

```
[name = ``condor_pool_atlas";
 TargetUniverse = 5;
 Requirements = target.x509UserProxyVOName =?= ``atlas";
 copy_requirements = ``original_requirements";
 set_requirements = original_requirements;
...]
```

Accounting per uso GPU

Nel job history file si trovano queste voci:

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

Aggiungendo GPUsProvisioned tra le extra keys raccolte

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

e otteniamo:

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

Cluster Management

- Il modello cambia da **centralizzato** (LSF) a **distribuito** (HTC). Ogni nodo del cluster ha **i suoi** files di configurazione.
- Ogni modifica che impatti un set di nodi va distribuita sui file di conf dei nodi interessati.
- Soluzioni popolari per distribuire le configurazioni: puppet, git, shared fs.

Dalle versioni piú recenti HTCondor accetta direttive di configurazione come output di script. (es: `/mysharedfs/configure_wn.py|`)

Questo consente di ottenere una modalità "emancipata" da puppet. Permette di modificare velocemente molte configurazioni; quelle che si rivelano "stabili" vengono poi fissate in puppet.

Fairshare con LSF

$$U_{\mathrm{dp}} = \frac{U_{\mathrm{share}}}{\varepsilon\,\mathrm{CPT} + \alpha\,\mathrm{WCT} + \beta\,(1+\mathrm{RUN}) + \gamma\,\mathrm{ADJ}}$$

LSF invia ai WN piú job di utenti con U_{dp} **maggiore**. Gli utenti accumulano CPT, WCT e job running (RUN) e U_{dp} cala. L'attività di competing users si assesta su quote proporzionali ai rispettivi share. ADJ può essere controllato dall'admin.

Fairshare con HTC

Da HTCondor manual, 3.6

$$\begin{aligned} \pi_r(u,t) &= \beta\,\pi_r(u,t-\Delta t) + (1-\beta)\,\rho(u,t) \\ \pi_e(u,t) &= \pi_r(u,t)\,f(u,t) \\ \beta &= (0.5)^{\Delta t/\tau} \end{aligned}$$

$\tau \equiv$ `PRIORITY_HALFLIFE`, $\rho \equiv \mathrm{RUN}$, $\pi_r \approxeq U_{\mathrm{share}} =$ RUP, $\pi_e \approxeq U_{\mathrm{prio}} =$ EUP

HTC invia ai WN piú job per utenti con π_r **minore**. I job sono inviati in proporzione inversa alle rispettive EUP. L'attività di competing users si assesta su quote proporzionali ai rispettivi share. Gli share sono assegnati per *user group* g_i, e $\sum g_i = 1$.

GROUP_NAMES = group_atlas, group_cms
GROUP_QUOTA_DYNAMIC_group_atlas = 0.42
GROUP_QUOTA_DYNAMIC_group_cms = 0.58

Work in progress

Fairshare. Serve uno script che converta pledge $\longleftrightarrow$ quote HS06.

Command line utils. Stiamo lavorando a una serie di alias/wrapper con output "ispirato" ai comandi di LSF (Kudos: F. Fornari)

Cluster Management. Estendere le funzionalità di `configure_wn.py` (ancora rudimentale).

Supporto Utenti. Abbiamo tradotto alcuni JDL (es. VIRGO) a submit file per HTCondor-CE, e alcuni di esempio per uso GPU. (Kudos: F. Fornari)

Cluster Monitoring and reports. Accedere a grafana alle info sullo stato del cluster attualmente disponibili per LSF (TODO).

HTCondor-CE su Slurm

- Un HTC-CE è in fase di setup per accesso a risorse HPC al CINECA (30Mh su Intel KNL) da parte degli esperimenti LHC (progetto: LHC@HPC, PI: T. Boccali)
- Difficoltà supplementari (organizzazioni differenti, tempi di compresenza piuttosto radi) ma superabili grazie ad un buon livello di collaboratività

HTCondor Task Force

- ml italiana https://lists.infn.it/sympa/info/htcondor-support (chi è interessato si iscriva!)
- Wiki: http://wiki.infn.it/progetti/htcondor-tf/
- Membri: P. Andreetto, S. Dal Pra, S. Fantinel, F. Fanzago, A. Italiano, F. Prelz, D. Rebatto;
- Best Effort mode ($0 \leqslant \sigma < \varepsilon$ dedicated resources)
- Confermo l'utilità, essendone stato primo fruitore!

Considerazioni finali

- HTC-CE funziona quasi "out of the box" con pilot jobs. Attualmente ancora alcuni "gaps": serve piú documentazione non OSG.
- JobRouting è il meccanismo piú importante per gestire un HTC-CE in produzione. Serve una certa pratica per prendere confidenza con le sue possibili configurazioni.
- Sufficiente corrispondenza con le funzionalità di CREAM-CE.
- GPU aware, accounting facilmente adattabile.
- Avvicendamento LSF $\rightarrow$ HTC in corso. Compresenza dei due sistemi per alcuni mesi.
- Supporto: con LSF supporto a "runtime". Ora: ml internazionale (altri siti, sviluppatori, altri) o "Task force locale". Risposte e contributi utili, ma soggette a "Best Effort syndrome" e Response Time $w \in \mathcal{E}(\lambda) \cup \{+\infty\},\ \lambda \sim O(\mathrm{days})$.
- $\sum_i \vec{u}_i \cdot \mathrm{Risorsa}_i \cdot \Delta T_i = \mathrm{Percorso}$