Centre de Calcul de l'Institut National de Physique Nucléaire et de Physique des Particules

Computing facility strategy

V. Hamar

CCIN2P3

cnrs

Overview

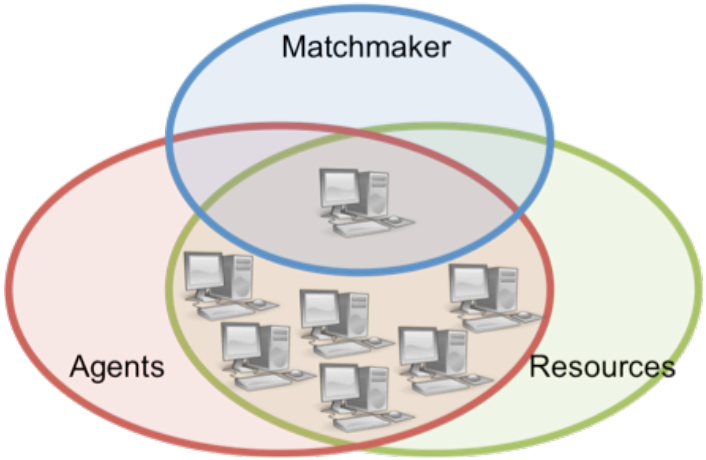
- HTCondor-CE
  - Deployment
  - Configuration
  - External grid nodes configuration
  - Monitoring
- HTCondor
  - Configuration at CC
- Future Work
- Conclusions

Centre de Calcul de l'Institut National de Physique Nucléaire et de Physique des Particules

HTCondor-CE

Architecture

Our choice



http://condorpy.readthedocs.io/en/latest/htcondor.html

Deployment

- HTCondor_CE cernops module
  - Customized to fit our needs
    - BDII deployment added
    - A few parameters changed
    - Deploy a CE is easy and reproducibly.

https://github.com/cernops/puppet-htcondor_ce

- Used in conjunction with HTCondor HEP-Puppet module
  - Very useful deployment tool for HTCondor

https://github.com/HEP-Puppet/htcondor

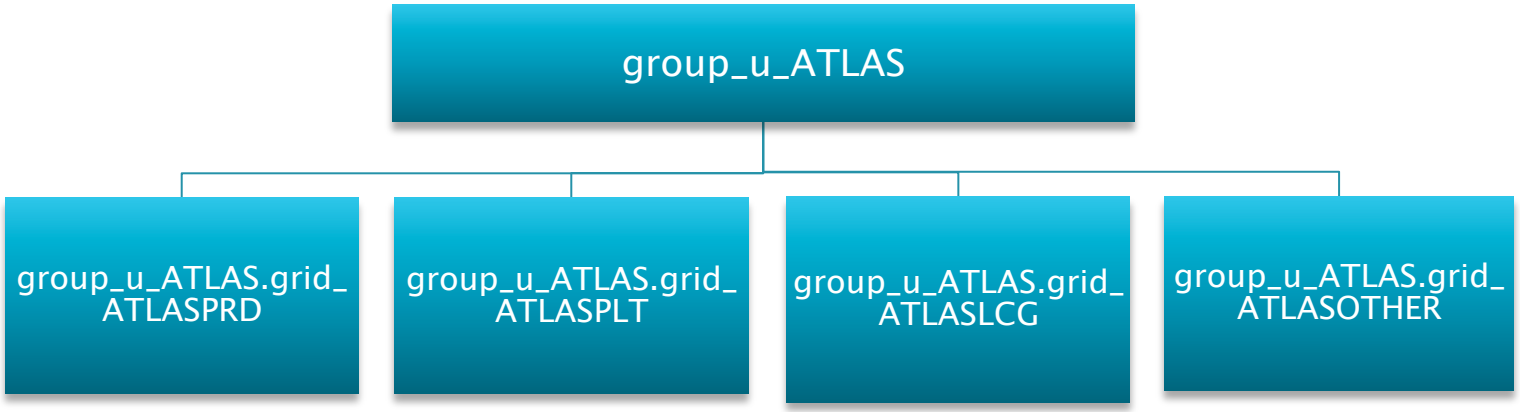
Deployment - Hieradata

```
htcondor_ce::pool_collectors:
  - 'cccondorcm01.in2p3.fr'
htcondor_ce::condor_view_hosts: []
htcondor_ce::ce_version: '3.3.0-1.el7'
htcondor_ce::lrms_version: '8.6.12-0.445603.el7'
htcondor_ce::uid_domain: 'in2p3.fr'
htcondor_ce::gsi_regex: '^\/O\=GRID-FR\/\C\=FR\/O\=CNRS\/OU\=CC-IN2P3\/CN\=([A-Za-z0-9.\-]*)$'
htcondor_ce::gsi_backend: 'argus'
htcondor_ce::argus_server: 'cctbargus01.in2p3.fr'
htcondor_ce::argus_port: 8154
htcondor_ce::argus_resourceid: 'http://cc.in2p3.fr/ce'
htcondor_ce::use_static_shadow: false
htcondor_ce::job_router_entries: >-
  [ \
   eval_set_environment = debug(strcat("HOME=/tmp CONDORCE_COLLECTOR_HOST=", CondorCECollectorHost, " ", \
TargetUniverse = 5; \
    name = "Local_Condor"; \
    set_VOName = ifThenElse(isUndefined(X509UserProxyVOName),"LOCAL",X509UserProxyVOName); \
    set_AcctSubGroup = toUpper(\
   …..
  eval_set_RequestMemory = ifThenElse(WantWholeNode is true, !isUndefined(TotalMemory) ? TotalMemory*95/100 : JobMemory, OriginalMemory); \
  ]

site_htcondor_ce::prerelease_repo_enabled: false
# BDII
htcondor_ce::install_bdii: true
htcondor_ce::supported_vos:
 - atlas
 - …..
htcondor_ce::goc_site_name: 'IN2P3-CC'
htcondor_ce::benchmark_result: '10.26-HEP-SPEC06'
htcondor_ce::execution_env_cores: 40
```

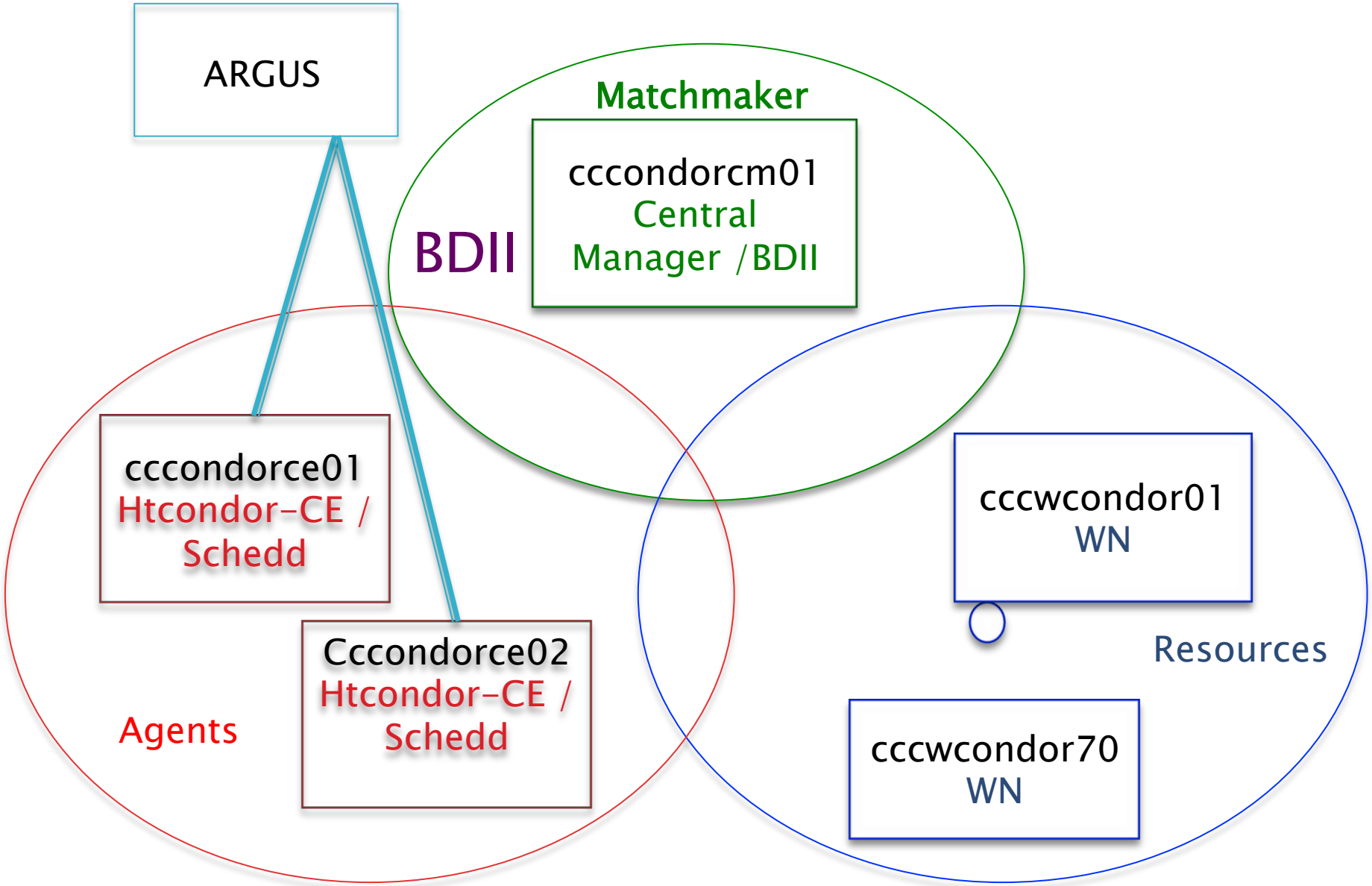
Configuration – Accounting Groups

- Job routers in local htcondor and htcondor-CE
- Hierarchy tree is based on VO names and user proxy role.
  - group_u_VONAME
    - group_u_VONAME.grid_{VONAME}{ROLE}



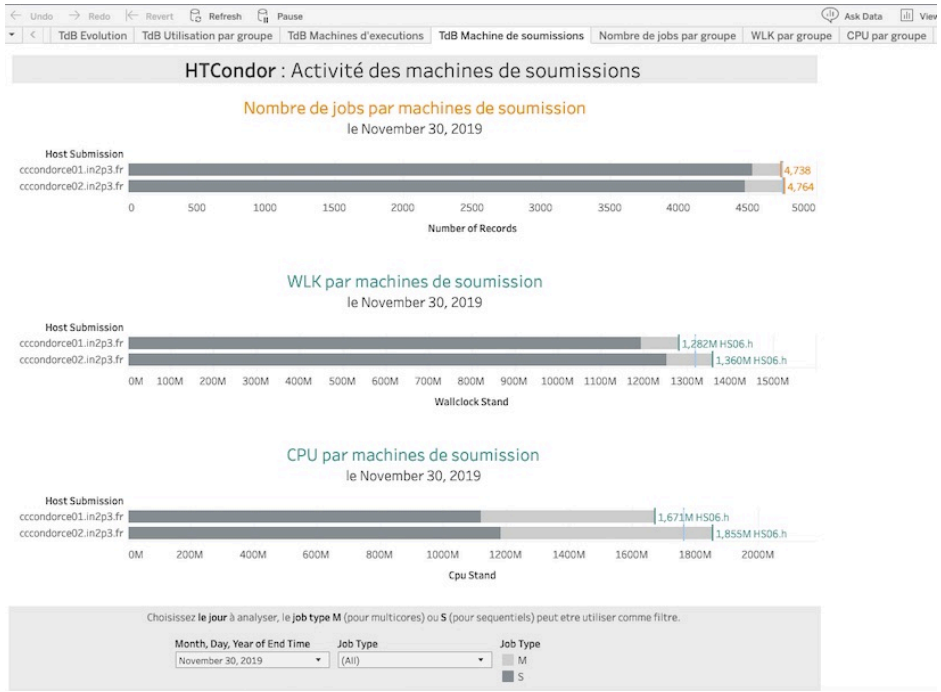
- Assuming that each proxy role represents a different activity, but a VO can use the same role for different activities.

Until now …


ARGUS
Matchmaker
cccondorcm01
Central
Manager /BDII
BDII
cccondorce01
Htcondor-CE /
Schedd
Cccondorce02
Htcondor-CE /
Schedd
Agents
cccwcondor01
WN
Resources
cccwcondor70
WN

Accounting

- Using condor_history command to generate json files by hour and by schedd and to save into a database.
  - Testing Apache Spark
  - Tableau – Data visualization tool
- Adapting our local scripts to generate APEL accounting files.



Jobs by HTCondor-CE

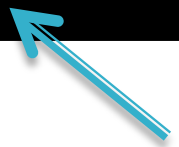
Monitoring

- Fifemon looked like the way to go
  - But CC-IN2P3 doesn't have graphite in its monitoring stack
  - Adapting it to Collectd + Elastic Search was time consuming and quite unsuccessful.

- Collectd + dedicated plugin
  - Currently writing a Python plugin for HTCondor
  - Using Python Bindings to gather classads
    - Though the Collector
      - Negotiator
      - Collector
      - Defrag
      - Schedd
      - Startd
    - Through the Schedd
      - Schedd ads (jobads of all jobs, regardless of their state)

- Collectd + plugin for HTCondor-CE as well
  - Query the HTCondor-CE collector with a simple trick

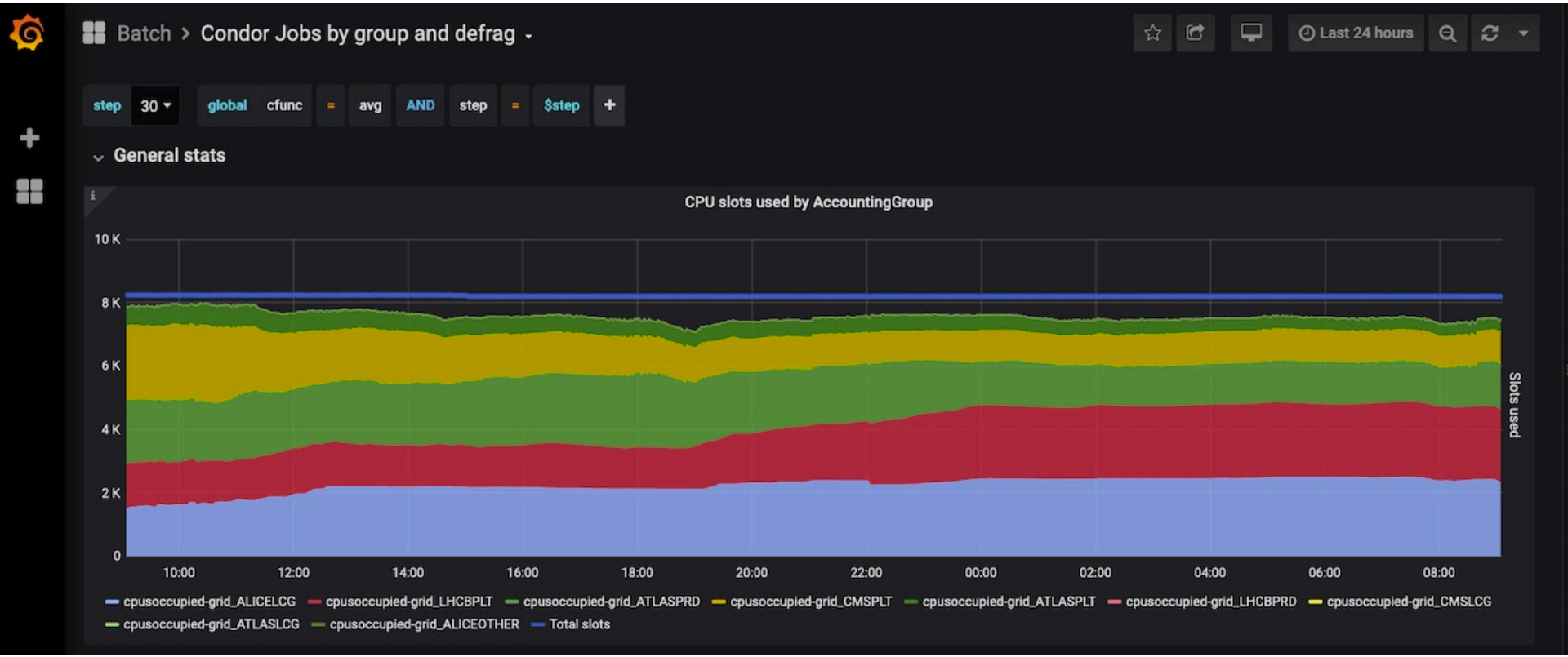
```
Import htcondor
ce_coll = htcondor.Collector('localhost:9619')
ce_coll.query()
```



Changing the port you get connected to CE

  - We gather
    - Job general stats
    - Schedd daemon metrics
  - Currently putting efforts in writing a proper reusable code for public release

- Grafana to plot the metrics



HTCondor Configuration

Configuration – Jobs Resource Requests

- Job Resource Requests
  - CPU's
  - Memory
  - Wallclock
  - Disk (N/A)

- Max limits
  - +xcount: Max 8 slots.
  - +maxMemory: Max 4GB by core.
  - +maxWallTime: Max 96 hours.

- The default values are set according to the VO demands and internal capacity. (CPUs vs Memory consumption)

Configuration – Accounting Groups

- Accounting is based groups depending on x509 user proxy role.
- There is not separation between single core and multicores.
- Let the VOs to used the “allocated” resources as they wish.
  - Local capacity plan will be enforce (respect the allocation of slots ~100%).
  - (e.g. the the maximum number of high memory jobs >= 4Gbytes per core)

Configuration – Fairshare Policy

- Dynamic Quota and “Surplus“ for all VOs
  - Share tree policy with groups and subgroup for optimum utilization of resources.
- Group Rebased Policy
  - in order all VO to be “delighted” from over-pledges resources.
- Trying to avoid practices which disturbs or constrain the Fairshare Scheduling
  - Reshape the multicore jobs priorities in order to facilitate the starting of the jobs.
- It could be special priorities for special users
  - SAM tests
- Try to use “breath-in” ramp-up mode of the machines
  - Uniform job distribution across all the machines

Resources fragmentation

- Single core (1) and multicore jobs (8-cores) should running simultaneously on a machine.
  - The problem of fragmentation of resources is that over time the machine resources may become partitioned into slots suitable only for running single core jobs.
  - If eight (8) single slots do not happen to become idle at the same time on a machine, then a multicore jobs with 8 cores will not be able to start that machine. Even if the multicore job has a better priority than the single jobs

- A lot of things to do!! Just starting
  - A short term:
    - Migrate all grid resources before February 2020.
    - Publish Collectd python plugin for HTCondor-CE and HTCondor ASAP.
    - Continue improving our configuration.

Conclusions

- Is easy to deploy HTCondor / HTCondor-CE using puppet modules.
- Defrag daemon is not aware about running condition on the farm therefore
  - We need an external script to feed the defrag parameter according to the runtime conditions on the farm (e.g. start or stop the defrag process)
- Needs to understand better the “wallclock” time distribution of single core jobs vs the multicore jobs.

A big thank you !!!

- To HTCondor developers, specially to:
  - Miron Livny
  - Todd Tannenbaum
  - John (TJ) Knoeller
  - Brian Lin
  - Zachary Miller

- CC-IN2P3 colleagues
  - Christelle Eloto
  - Cécile Evesque
  - Nicolas Fournials
  - Ghita Rahal
  - Emmanouil Vamvakopoulos