Deployment of HTCondor at CC-IN2P3

Guillaume Cochard, Vanessa Hamar, Bertrand Rigaud and Emmanouil Vamvakopoulos

HTCondor workshop autumn 2020

Computing Centre of IN2P3

Computing Centre of IN2P3 (or CC-IN2P3) is a CNRS support and research unit attached to IN2P3/CNRS institute which pursues and coordinates research on particle physics, nuclear physics and astroparticle physics

The main mission of the CC-IN2P3 is to provide these researchers with the computing power that they need and also the related information technology services:

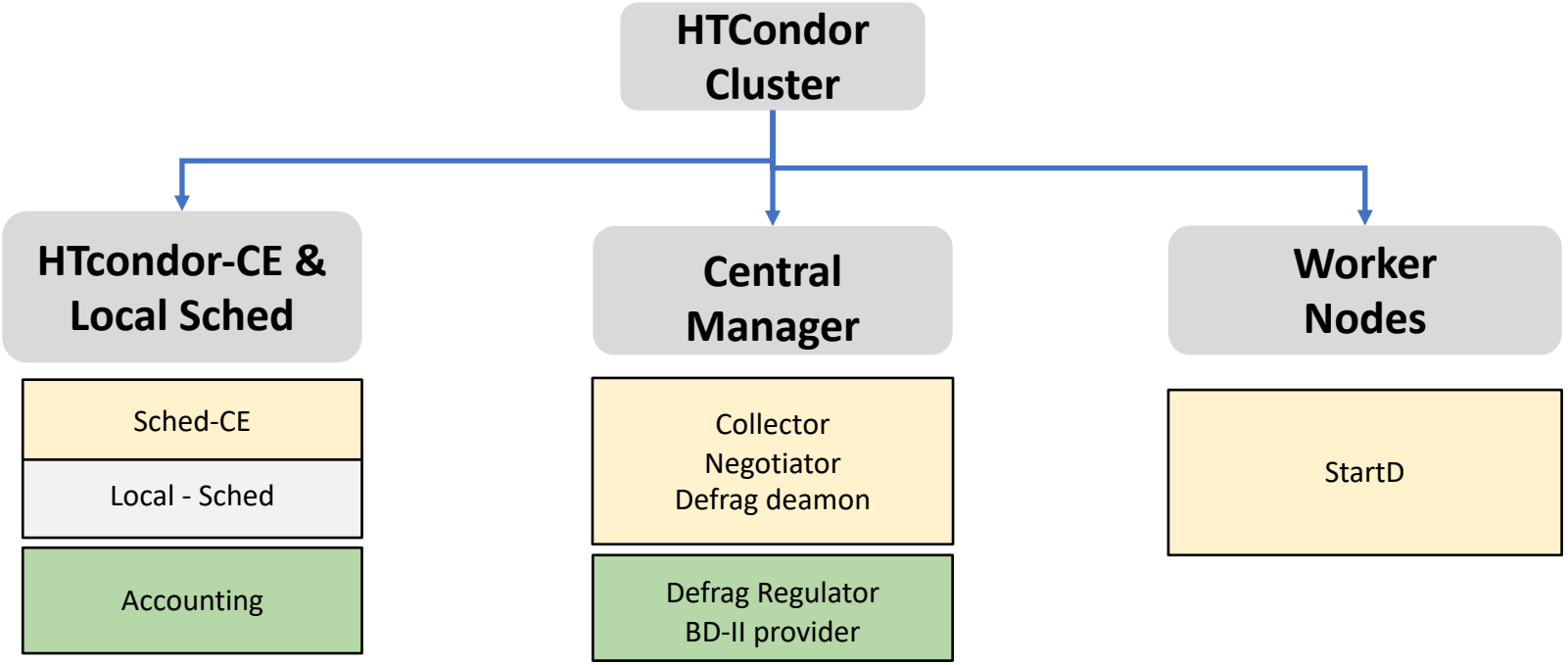
- ✓ A mass storage system and mass data processing resources
- ✓ Developing and managing tools for assisting scientific production
- ✓ Hosting services

Two Computer rooms ~850m^2 each , ~3000 servers ~15PB on disk ,~34PB on tapes.

CC-IN2P3 acts as Tier 1 for WLCG international project and support various VOs under EGI Umbrella

Service structure

A typical layout



Condor-ce version 4.1.0
Condor version 8.8.9

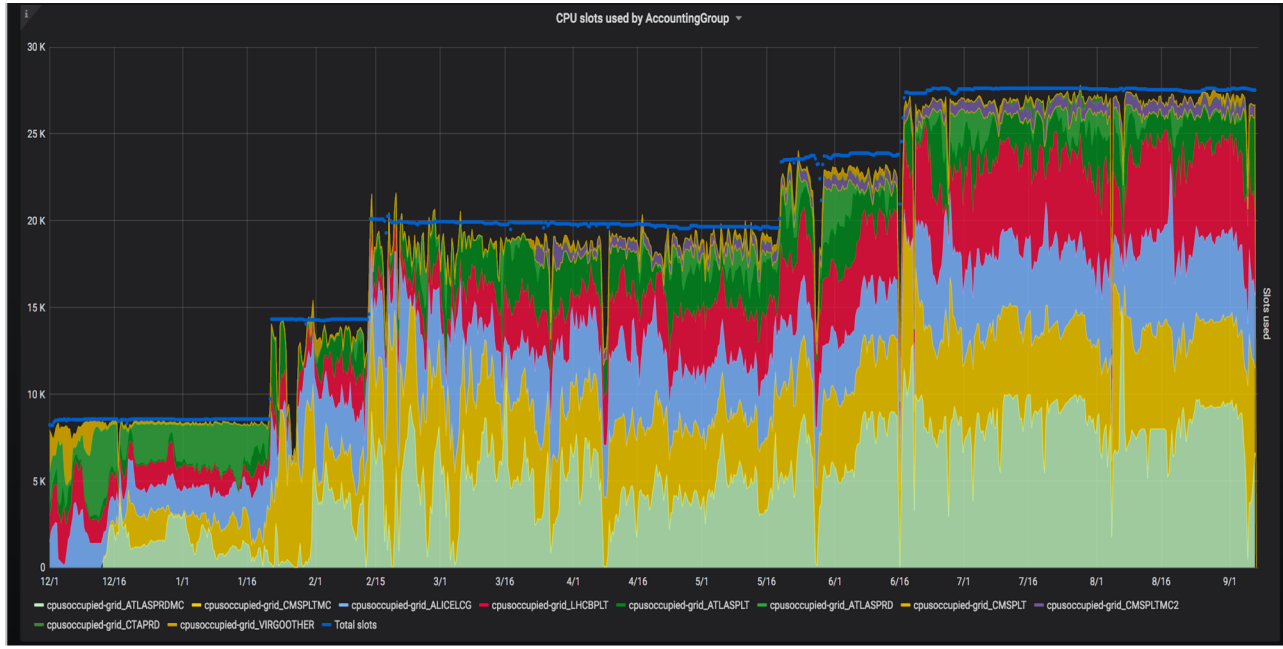
Migration to HTCondor

In the last 9 months, we migrated worker nodes from UNIVA Grid Engine (C) cluster to HTcondor in four (4) steps.

After June 2020, 100% of WLCG/EGI computing pledges (up to ~28K slots) are offered by our HTCondor cluster, exclusively.

HTCondor is the only workload management system for the Grid environment at CC-IN2P3

UNIVA Grid Engine serves local users and other particular research activities.



HighLigth Configuration

Acounting Group sets: Three (3) levels VO.Activity.Owner

- AccountingGroup = group_u_VOname.grid_SubGroup.OWNER
- VOname : A string which describes the NickName which corresponds to VO name after a mapping.
- Subgroup: String of VOname, activity and number of requested core for a job

(ag. ATLASPRD, ATLASPLT, ATLASPRDMC , ALICELCG, CTAUSR, …, etc)

ONWER : A string describing the user who submitted a job.

Job Router custom configuration

```
JOB_ROUTER_ENTRIES = \
  [ \
   name = "Local_Condor";\
   TargetUniverse = 5;\
   set_LocalTag = ifThenElse(isTagJob isnt undefined,isTagJob,False);\
   set_dLocalDisk = (25000000*localcpus);\
   set_RequestDisk = dLocalDisk ;\
   set_VOname=userMap("vomap",X509UserProxyVOName?:"local");\
   eval_set_VOGroup = VOname ;\
   set_dLocalMemory=int(userMap("vomem",VOname))*localcpus;\
   set_localcpus = xcount ?: (orig_RequestCpus?:1 > 1) ? orig_RequestCpus : (default_xcount?:1);\
   set_AcctSubGroup=toUpper(strcat(VONAME,userMap("vorole",X509UserProxyFirstFQAN)));\
   set_AcctSubGroupExt = ifThenElse(regexp("ATLAS|CMS|DTEAM",toUpper(VOname)),ifThenElse((localcpus >
1),strcat(AcctSubGroup,"MC"),AcctSubGroup),AcctSubGroup);\
   eval_set_AcctGroup = strcat("group_u_",VOname,".","grid_",AcctSubGroupExt);\
   eval_set_AccountingGroup = strcat("group_u_",VOname,".","grid_",AcctSubGroupExt,".",OWNER);\
   eval_set_ConcurrencyLimits = strcat(toUpper(VOname),",",AcctSubGroupExt,":",MY.RequestCpus);\
   eval_set_OriginalMemory = ifThenElse(maxMemory isnt undefined,maxMemory,dLocalMemory);\
  ]
```

vomap file:

```
...
* /^virgo/ virgo
* /^vo.france-grilles.fr/ fgrilles
* /^vo.formation.idgrilles.fr/ training
* /^vo.cta.in2p3.fr/ cta
```

vorole file:

```
...
* /production/ prd
* /pilot/ plt
* /lcgadmin/ lcg
* /(.*)/ other
```

Submitters

Condor_status -submitters

```
                                                RunningJobs   IdleJobs   HeldJobs

    group_u_alice.grid_ALICELCG.aligrid@in2p3.fr       4555        100          0
 group_u_alice.grid_ALICEOTHER.alice097@in2p3.fr          0          0          0
   group_u_atlas.grid_ATLASLCG.atlagrid@in2p3.fr          0          0          0
   group_u_atlas.grid_ATLASPLT.atlas099@in2p3.fr       1840          0          0
 group_u_atlas.grid_ATLASPLTMC.atlas099@in2p3.fr          0          0          0
   group_u_atlas.grid_ATLASPRD.atlas100@in2p3.fr        272          0          0
 group_u_atlas.grid_ATLASPRDMC.atlas100@in2p3.fr       1100        489          0
        group_u_cms.grid_CMSLCG.cmsgrid@in2p3.fr          0          1          0
       group_u_cms.grid_CMSPLTMC.cms271@in2p3.fr        630        594          0
      group_u_cms.grid_CMSPLTMC2.cms271@in2p3.fr         75         38          0
         group_u_cta.grid_CTAPRD.cta050@in2p3.fr          0          0          0
   group_u_dteam.grid_DTEAMOTHER.ops011@in2p3.fr          0          1          0
group_u_escap.grid_ESCAPOTHER.escape030@in2p3.fr          0          0          0
       group_u_ilc.grid_ILCOTHER.ilc036@in2p3.fr          0          0          0
      group_u_juno.grid_JUNOPLT.juno099@in2p3.fr        272        385          0
      group_u_lbno.grid_LBNOPLT.dune100@in2p3.fr          0          0          0
    group_u_lhcb.grid_LHCBOTHER.lhcb080@in2p3.fr          0          6          0
      group_u_lhcb.grid_LHCBPLT.lhcb049@in2p3.fr       5468        451          0
      group_u_lsst.grid_LSSTPLT.lsst001@in2p3.fr          0          0          0
         group_u_t2k.grid_T2KPLT.t2k099@in2p3.fr          0          0          0
 group_u_virgo.grid_VIRGOOTHER.virgo100@in2p3.fr          8        176          0

                                           Total      14220       2241          0
```

Negotiator and Startd Policy

- **Negotiator**
  - FairShare policy : Group Hierarchy + Surplus
  - NEGOTIATOR_PRE_JOB_RANK = - Memory (Best fit for memory)
  - NEGOTIATOR_POST_JOB_RANK = + 100*Cpus (First for Cpus in case of ties)
  - CLAIM_PARTITIONABLE_LEFTOVERS = False (everthing pass from negotiator)
  - Finite GROUP_QUOTA_ROUND_ROBIN_RATE =512
- **Special Start Condition**
  - Usages of Partitionable Slots
  - (Total Used Memory+RequestMemory) / (Total Used Cpus+Request Cpus) <= 3.5GB
  - Single core job up to ~60% of Total Cpu Slots of the worker
  - Permit match of MCORE jobs ONLY(for 900 sec) after a successful partial draining period
  - A tag condition in order to reserve a node for a specific job and/or specfic owner
  - Publish Puppet facts as htcondor machine classads (e.g. HS06 factor, machine type, … etc)

Startd custom knobs

```
htcondor::custom_knobs:
  CLAIM_PARTITIONABLE_LEFTOVERS: 'False'
  SLOT_TYPE_1: 'cpus=100%,mem=100%,disk=100%,auto'
  MODIFY_REQUEST_EXPR_REQUESTMEMORY: 'quantize(RequestMemory, {128})'
  JOB_RENICE_INCREMENT: 0
  MAXJOBRETIREMENTTIME: '$(HOUR) * 24 * 4'
  OnlyMulticoreInterval: '900'
  IsMulticore: 'RequestCpus >= IfThenElse(Cpus<8,1,8)'
  IsntUnmatchedPSlot: 'PartitionableSlot=!=true || State=="Matched »'

  OnlyMulticoreJobsAfterDrain : '$(IsntUnmatchedPSlot) || $(IsMulticore) ||
( $(StateTimer) > $(OnlyMulticoreInterval) && $(ISOK) && $(ISPOP))'

  ISPOP : 'ifThenElse(TotalCpus>40,NumDynamicSlots<= 36,NumDynamicSlots <= 24)'

  ISOK : '(( TotalMemory - Memory + Target.RequestMemory+1) / (TotalCpus - Cpus + Target.RequestCpus)) <
3500'
  isLocalTag: 'Target.LocalTag =?= False'
  START: '($(START)) && ($(OnlyMulticoreJobsAfterDrain)) && ($(isLocalTag))'
```

De-fragmentation of Resources

- Single core (1) and multicore jobs (8-cores) should run simultaneously on worker. 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.
- We need to defrag the machine (drain) up to the moment that we get back eight (8) slots free (partial drain).
- **When should the defrag process start/end ?**
  - We start the defrag process when the # running of MCore jobs is below of the «target»
  - We stop the defrag process when attain the target
- **Which machines are more desirable to defrag ?**
  - We sort the machines according to the number of SC running jobs
- **How many machines are defraged at once ?**
  - All the machines in order to fill the GAP as soon as posible
  - **GAP** = (MC_Target - MC_running)/8
  - DEFRAG_MAX_WHOLE_MACHINES=**GAP**
  - DEFRAG_MAX_CONCURRENT_DRAINING = **GAP**
  - DEFRAG_DRAINING_MACHINES_PER_HOUR=4*GAP
- We perform partial drain up to 8 slots, WHOLE_MACHINE_EXP = (Cpus >= 8 && PartitionableSlot)

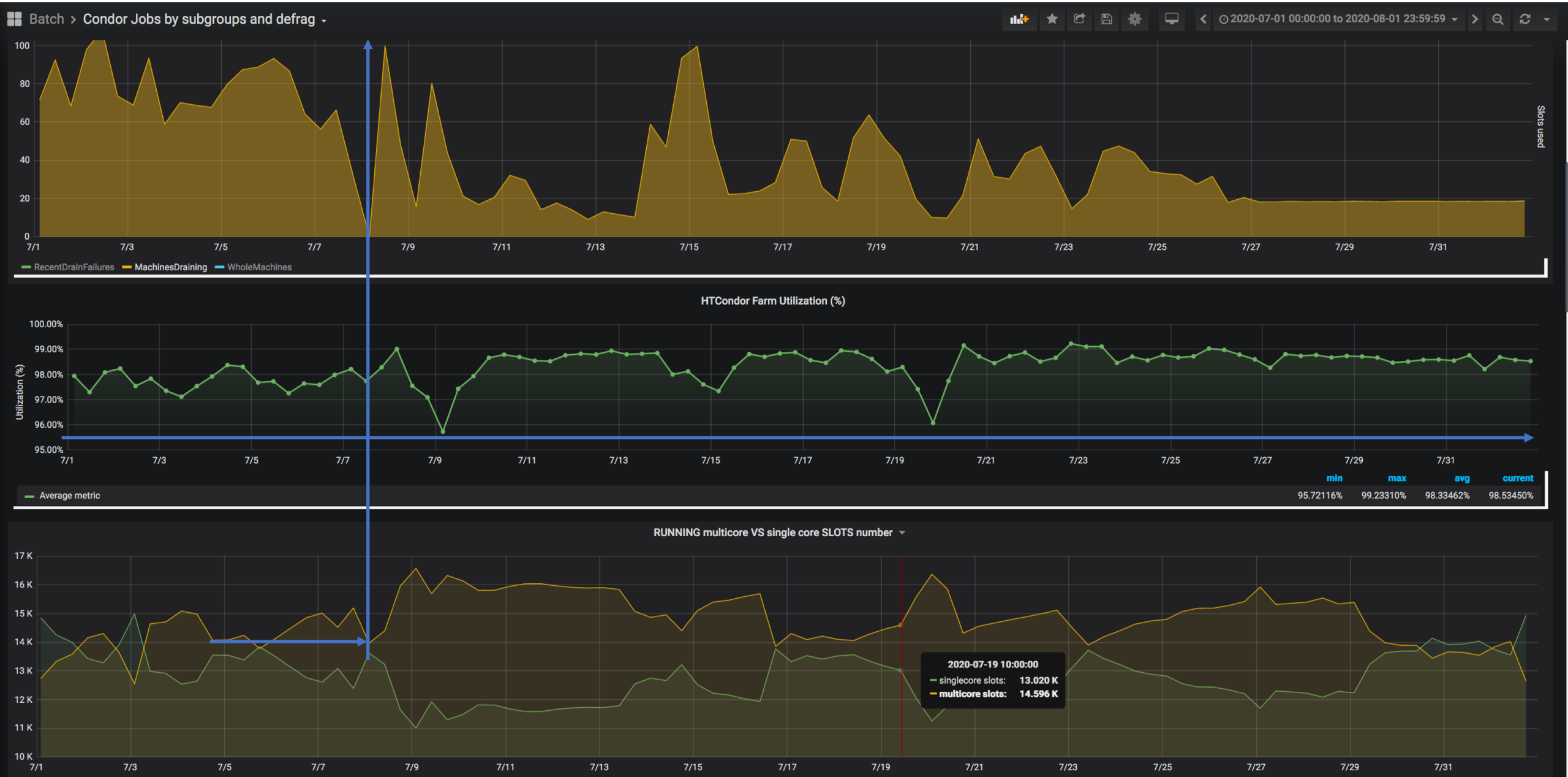
We follow the work « Examination of dynamic partitioning for multi-core jobs in the Tokyo Tier-2 center » T. Kishimoto et al. , ISGC 2017.

We define the MC_Target as the sum of the static demand for mutlicore jobs (CMS) plus the dynamic one (atlas).

$$\text{MC\_Target} = \text{CMS}_{\text{quota}} + \frac{\text{\# of MCore Jobs in all state}}{\text{(\# Total Atlas Jobs in all state)}} \text{ATLAS}_{\text{quota}}$$

- We are taking into account the fact that ATLAS VO uses «early binding» for pilot job-submition. Therefore, everything that is on the «queue» should run at some point. From the ratio of the atlas MCores jobs (in all states) over the number of total atlas jobs (in the queue), we can estimate the desirable dynamic share for atlas MC jobs and define a target level.
- We run periodicaly (~1h) an external script which queries the htcondor scheds and makes the appropriate calculations for parametrizing the defrag daemon on-the-fly.

Monitoring Defrag process



Common issues and problems

- Ticket **#7602**: quota incorrectly calculated for jobs that request more than 1 cpu afffects:
  - Our v8.8.6 installation, the problem solve with an update to v8.8.9.
  - We had diffuculties to respect the fairshare with group quotas + surplus
- Issue with LHCb proxy expiration of completed job which stay on spool(?) (complete jobs became on hold state ~10-20k), we upgrade from 3.6.5 condor-ce to 4.1.0 (skip the proxy expiration condition on SYSTEM_PERIODIC_HOLD)
- BDII-glue2, github issue **#376**
  - Due to GGUS ticket **#142898 (EGI)** and **#146122 (ILC)**
  - The GLUE2ExecutionEnvironmentLogicalCPUs set to total logical cpus
  - GLUE2ExecutionEnvironmentMainMemorySize to memory per slots
- As I understood from the comment/code the number of # surplus slots are analogous to the number of slots in pending state for one accountig subgroup. Therefore the accounting subgroup with bigger "pressure" will take bigger surplus: late binding vs early binding ?

Further Steps

- Finalize the defrag policy
- Update to the new version > 8.9.x
- SciTokens vs X509 proxy certificate
- Evaluate Htcondor as an option to replace Univa Grid Engine for Local Users
  - Submission, auth and authz (with Kerberos & Tokens)
  - Accounting groups
  - Resources and Limits
- Further improve the memory capacity plan for high memory jobs
- Match of special jobs (e.g. GPU)

Many thanks to HTCondor Team

for recommendations and fruitfull discussions

BACKUP

Outline

Service Structure and Volume

- How many different roles we have?
- Which is the volume of components?

Deployment

- Which version we use

Accounting Groups, FairShare and Limits

- How could we define different accounting Groups with different share?
- How could we configure limits ?

Defragmentation Policy

- What is the defrag process? When do we start the defrag? When do we stop it?
- and how many machines should we reserve for this operation?

Problems and bugs

Further Steps

Deployment

For the HTCondor and HTCondor-ce deployment, we use the puppet modules from the community.

- https://github.com/cernops/puppet-htcondor_ce
- https://github.com/HEP-Puppet/htcondor

In conjunction with local modules that install and manage local specific resources:

- BDII provider
- Machine job features
- Accounting classadds (e.g HS06 factor, machine type)
- Nagios checking probes
- Collectd module for monitoring
- Other local resources (external scripts for accounting)

Deployment of Condor-ce

Particular for the htcondor-ce 4.1.0 deployment, we made changes on puppet-htcondor_ce module

We deploy a template of condormap compatible with used cordormap file of v4.1.0

```
GSI (.*) GSS_ASSIST_GRIDMAP
GSI "(/CN=[-.A-Za-z0-9/= ]+)" \1@unmapped.opensciencegrid.org
CLAIMTOBE .* anonymous@claimtobe
FS "^(root|condor)$" \1@daemon.htcondor.org
FS (.*) \1
```

We remove all recursive configuration of «ce-site-security.conf.erb » like:

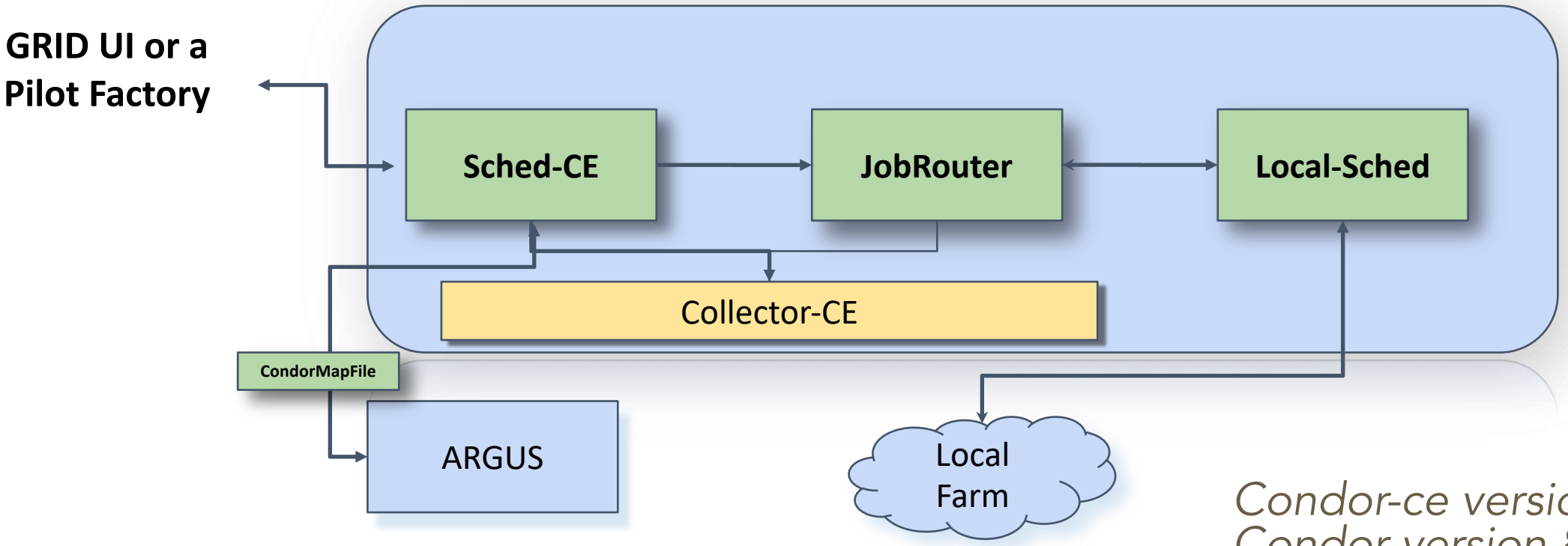
```
…
ALLOW_DAEMON = $(ALLOW_DAEMON), $(FRIENDLY_DAEMONS)
SCHEDD.ALLOW_NEGOTIATOR = $(SCHEDD.ALLOW_NEGOTIATOR), $(FULL_HOSTNAME)@$(UID_DOMAIN)/$(FULL_HOSTNAME), *@$(UID_DOMAIN)
…
```

And use valid defaults security configuration from rpm package which are installed at */usr/share/condor-ce at /config*

Capacity of the Service

- **646 x worker of 3 types ~ 28000 slots , hypertheading is enabled**
  - **96** PowerEdge C6220 II (Intel E5-2680 @2.80GHz), 40 slots 3G/Slot RAM, 1.6TB scratch,
  - **454** PowerEdge C6420 (Intel Silver 4114@2.20GHz), 40 slots 3G/Slot RAM, 1.6TB scratch
  - **96** PowerEdge C6525 (AMD EPYC 7302@3.3GHz), 64 slots 3G/Slot RAM, 1.6TB scratch
- **2 x HTCondor-CE for T1**
  - PowerEdge R440 (IntelSilver 4112@2.60GHz), 32GB RAM, 2x1TB (ssd)
- **1x HTCondor-CE for T2**
  - VM, 8 vcpu, 8GB ram, 50GB disk template
- **2x Central Manager**
  - PowerEdge R440, (IntelSilver 4112@2.60GHz), 32GB RAM, 2x1TB (ssd)

HTCONDOR-CE



Condor-ce version 4.1.0
Condor version 8.8.9

HighLigth Configuration II

ConcurrencyLimits = VOname, SubGroup:RequestCpus

- VOname : A string which describe the NIS Group Name which corresponds to VO name after a mapping.
- Subgroup: Concatenated string of VOname, activity abbrevation and number of requested core for a job (e.g. ATLASPRDMC , ALICELCG, CTAUSR, … etc)
- RequestCpus: The number of CPUs requested for a job.

Wallclock time per VO

[167]:

Cpus	x509UserProxyVOName	count	mean	std	min	5%	25%	50%	75%	95%	max
1	alice	19380.0	126754.0	30866.0	1.0	102811.0	110224.0	124572.0	150641.0	165874.0	173087.0
	atlas	86001.0	19780.0	27019.0	18.0	161.0	2518.0	5754.0	31437.0	67498.0	325273.0
	cms	3288.0	131.0	286.0	31.0	64.0	75.0	89.0	118.0	251.0	5325.0
	escape	77.0	117.0	238.0	1.0	2.0	4.0	21.0	86.0	797.0	1000.0
	juno	865.0	10492.0	40731.0	1.0	26.0	31.0	126.0	143.0	104577.0	261403.0
	lhcb	38407.0	50526.0	38173.0	63.0	6219.0	20606.0	38188.0	71954.0	119138.0	345213.0
	lsst	2.0	200.0	16.0	189.0	190.0	195.0	200.0	206.0	211.0	212.0
	ops	144.0	2.0	1.0	2.0	2.0	2.0	2.0	3.0	4.0	7.0
	vo.cta.in2p3.fr	122.0	12946.0	16918.0	105.0	123.0	184.0	1226.0	21374.0	51025.0	77973.0
2	virgo	3980.0	10286.0	22923.0	750.0	2529.0	2608.0	2696.0	2812.0	68528.0	164817.0
4	dune	395.0	7968.0	12255.0	704.0	1300.0	1446.0	1641.0	10840.0	29004.0	97644.0
8	atlas	34457.0	14741.0	9520.0	2.0	170.0	8483.0	13407.0	20405.0	28749.0	230201.0
	cms	3576.0	117292.0	64591.0	745.0	3629.0	45705.0	158025.0	165169.0	171658.0	173458.0