Condor at the RACF

Migration to 7.4,
Group Quotas, and More

William Strecker-Kellogg
Brookhaven National Lab

RHIC/ATLAS Computing Facility Overview

- Physics Dept. at Brookhaven National Lab—provides computing and storage to active RHIC experiments
- Serves as a Teir-1 for ATLAS computing
- Uses Condor to manage RHIC and ATLAS compute clusters
- 14.4k cores running SL5.3 currently
- With new hyper-threaded 12-core Westmere systems, to grow to over 20k cores

RHIC/ATLAS Computing Facility Overview

- One instance for ATLAS
  - 5100 slots
  - 2 submit nodes manage all production/analysis jobs
  - Other smaller queues managed on 3 other submit nodes
- Instance each for STAR and PHENIX experiments
  - 4300, 4500 slots resp.
  - 20 submit nodes each
  - "General Queue"—flocking between RHIC pools
- Smaller experiments grouped into another instance

New Since Last Year

- New condor administrator
- Migration to 7.4.2
  - Up from 6.8.9—long overdue
- Move ATLAS to Group Quotas
  - Easier configuration—from 16 configuration files to 5—90% of slots use just 1)
  - Management via web-interface
  - Some problems we've had to solve…more later

Upgrade to 7.4.2

- Get rid of suspension model
  - Undesirable to have slots =!= real cores
  - Simplify START expression
- Better negotiator performance, results later
- Bugfixes all around

Example START Expression

```
Start = ((((RealExperiment == "atlas") && (VirtualMachineID >= 7) && ((TARGET.RACF_Group =?= "short" ||
        TARGET.RACF_Group =?= "dial" || Owner =?= "usatlas2" || (stringListMember("acas0201",
"acas0200,acas0201,acas0202,acas0203,acas0204") && TARGET.RACF_Group =?= "lcg-ops") || (stringListMember
 ("acas0201", "acas0200,acas0201,acas0202,acas0203,acas0204") && TARGET.RACF_Group =?= "lcg-dteam")) &&
      (RemoteWallClockTime < 5400))) || ((RealExperiment == "atlas") && ((VirtualMachineID < 7) &&
(VirtualMachineID >= 5)) && ((TARGET.RACF_Group =?= "usatlas" || TARGET.RACF_Group =?= "usatlas-grid" ||
 (stringListMember("acas0201", "acas0200,acas0201,acas0202,acas0203,acas0204") && TARGET.RACF_Group =?=
 "lcg-atlas") || TARGET.RACF_Group =?= "bnl-local") && ((((vm7_Activity =?= "Busy") + (vm7_Activity =?=
 "Retiring") + (vm8_Activity =?= "Retiring") + (vm8_Activity =?= "Busy"))) < 2))) || ((RealExperiment ==
  "atlas") && ((VirtualMachineID >= 3) && (VirtualMachineID < 5)) && ((TARGET.RACF_Group =?= "grid" ||
       (stringListMember("acas0201", "acas0200,acas0201,acas0202,acas0203,acas0204") =?= FALSE &&
    TARGET.RACF_Group =?= "lcg")) && ((((vm7_Activity =?= "Busy") + (vm7_Activity =?= "Retiring") +
(vm8_Activity =?= "Retiring") + (vm8_Activity =?= "Busy")) + ((vm5_Activity =?= "Busy") + (vm5_Activity
        =?= "Retiring") + (vm6_Activity =?= "Retiring") + (vm6_Activity =?= "Busy"))) < 2))) ||
   (((RealExperiment == "atlas") || (RealExperiment =!= "atlas" && FALSE == FALSE && TRUE == FALSE &&
    LoadAvg < 1.400000 && TotalVirtualMemory > 200000 && ((Memory * 1024) - ImageSize) > 100000)) &&
     ((VirtualMachineID >= 1) && (VirtualMachineID < 3)) && ((TARGET.RACF_Group =?= "gridgr01" ||
TARGET.RACF_Group =?= "gridgr02" || TARGET.RACF_Group =?= "gridgr03" || TARGET.RACF_Group =?= "gridgr04"
    || TARGET.RACF_Group =?= "gridgr05" || TARGET.RACF_Group =?= "gridgr06" || TARGET.RACF_Group =?=
 "gridgrXX" || TARGET.RACF_Group =?= "gridgr08" || TARGET.RACF_Group =?= "gridgr09" || TARGET.RACF_Group
=?= "gridgr10" || TARGET.RealExperiment =!= "atlas") && ((((vm7_Activity =?= "Busy") + (vm7_Activity =?=
 "Retiring") + (vm8_Activity =?= "Retiring") + (vm8_Activity =?= "Busy")) + ((vm5_Activity =?= "Busy") +
      (vm5_Activity =?= "Retiring") + (vm6_Activity =?= "Retiring") + (vm6_Activity =?= "Busy")) +
      ((vm3_Activity =?= "Busy") + (vm3_Activity =?= "Retiring") + (vm4_Activity =?= "Retiring") +
(vm4_Activity =?= "Busy"))) < 2)))) && (Owner =!= "jalex" && Owner =!= "grau" && Owner =!= "smithj4") &&
                                        FALSE == FALSE)
```

From this

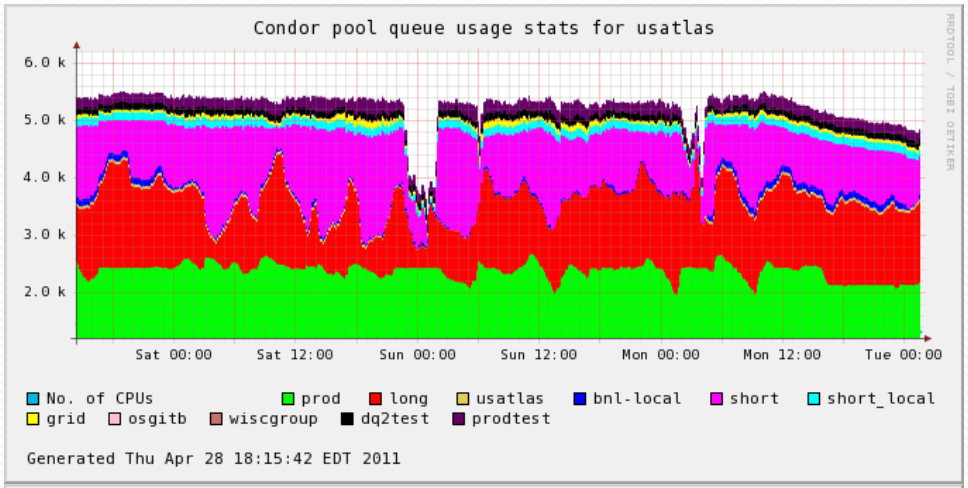

```
File  Edit  Tools  Syntax  Buffers  Window  Help
((TARGET.RealExperiment =?= "atlas" && (TARGET.RACF_Group =?= "prod" ||
TARGET.RACF_Group =?= "distr_analysis" || TARGET.RACF_Group =?= "short" ||
TARGET.RACF_Group =?= "prodtest")) || (TARGET.RealExperiment =!= "atlas" &&
FALSE == FALSE && FALSE == FALSE && LoadAvg < 1.400000 && TotalVirtualMemory >
200000 && ((Memory * 1024) - ImageSize) > 100000)) && TARGET.AccountingGroup
=!= UNDEFINED && (TARGET.Owner =!= "jalex" && TARGET.Owner =!= "XXX")
~
~
~
~
~
~
~
~
                                                          6,70          All
```

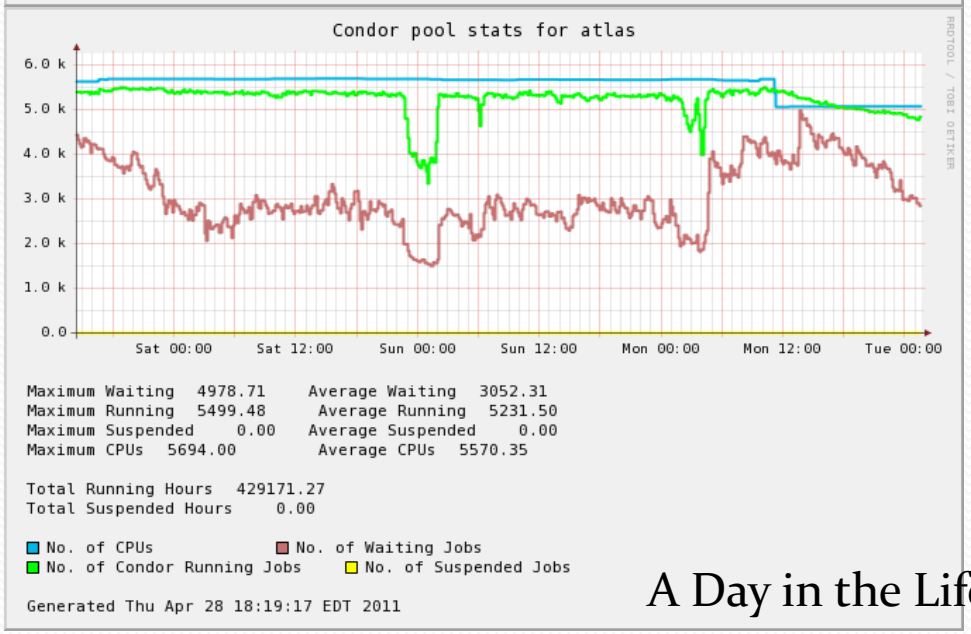
To this

Group Quotas

- ATLAS only, for now
  - PHENIX to follow suit in a few months
  - No plans for STAR
- What groups buy us:
  - Manage ATLAS production/analysis jobs separately from many other smaller queues
  - Unify configuration files—one config for vast majority of ATLAS nodes

ATLAS Group Quotas





Reallocation of resources between queues managed via web interface

File Edit View History Bookmarks Tools Help

bnl.gov https://webdocs.racf.bnl.gov/Facility/LinuxFarm/cgi-bin/grou

Fedora Project Farm Webpages RACF WebDocs Public RACF BNL Homepage

ATLAS Group Quotas

ATLAS Condor Group Quotas:

Caution: User *willsk* is authorized to add/remove groups!

With this page, you can edit the number of slots in each group. The number of slots is called that group's "Quota" and determines the maximum amount of resources that condor allocates to that group. As the sum of all the quotas must add up to the total number of slots available, this sum cannot change. **If you reallocate resources from one quota to another, make sure that this sum remains constant, or the change will not take effect.**

Group Name	Quota	Priority	Auto Regroup	Busy*
group_atlaslocal	48	100.0	TRUE	48
group_bnllocal	120	100.0	TRUE	21
group_distr_analysis	700	10.0	TRUE	694
group_grid	40	100.0	TRUE	64
group_prod	3125	1.0	FALSE	3107
group_prodtest	160	3.0	FALSE	0
group_short	512	4.0	TRUE	685
group_usatlas2	10	1.0	FALSE	0
group_wisc	368	100.0	FALSE	295

*Busy slot info updated every 5 minutes
*Last updated: Apr 28 06:15:01 PM

Total Slots=5083

Edit Group Quotas

Now: Foggy, 59° F

A Day in the Life of ATLAS

Issues Moving to Group Quotas

- Backwards compatibility with our accounting and monitoring systems
  - Solution: Retain previous job-type flag that used to hard-partition slots
- How does it interact with flocking?
- Fairness / Enforcement of group memberships
  - “We rely on societal enforcement”
- Not good enough…solution for ATLAS
  - ATLAS uses PANDA, we control local submission
  - Other users few enough to monitor individually

Issues Moving to Group Quotas

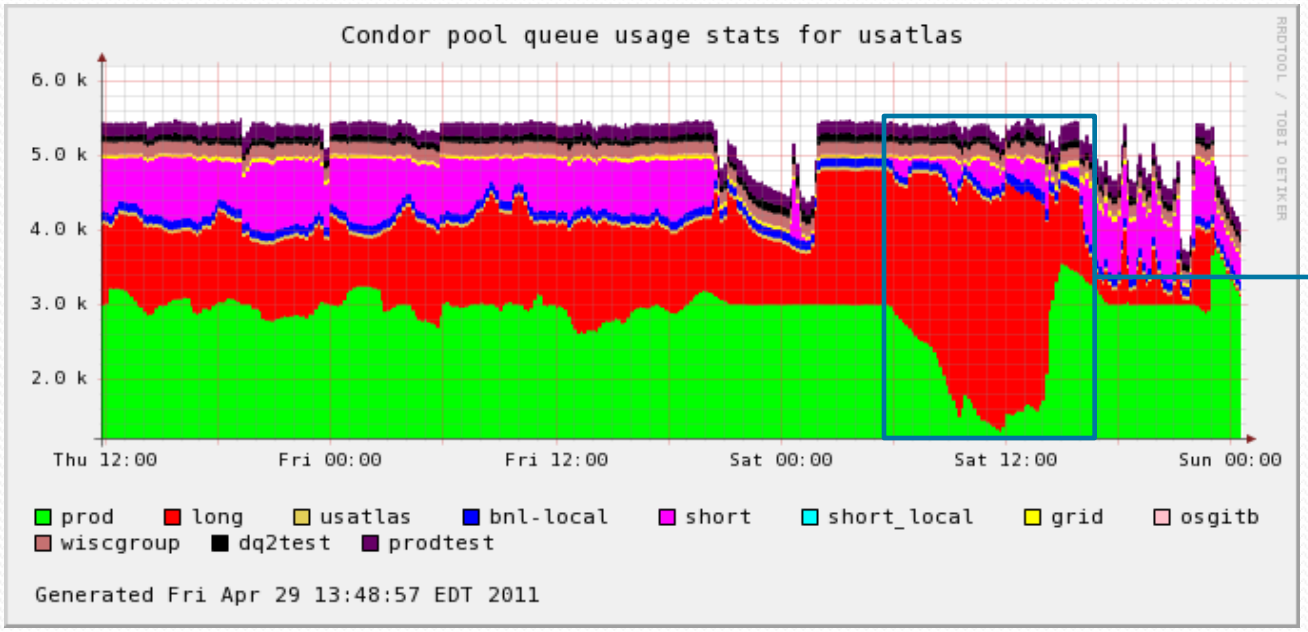
- PHENIX: two classes—user and special jobs
  - Special jobs submitted from few machines, separate users
  - User jobs from 20 submit nodes
- Two solutions
  - Submit node based partition: regex-match GlobalJobID ClassAd against list of valid sources in START expr.
  - Coexist with users: three configs, user nodes w/ no AccountingGroup flag, shared nodes that run anything but are ranked last by user and special jobs, and special nodes requiring AG flag

Group Priorities

- ATLAS setup: three main groups
  1) Production: highest prio., hard quota, no preemption
  2) Short analysis: medium prio., auto-regroup on, preemption enabled
  3) Long analysis: lowest prio., auto-regroup on, preemption enabled
- Idea—short and long analysis spill over *into each other* as needed and not be squeezed out by production
- Problem—sometimes short and long will "eat into" production even when they are over-quota and production is under its quota

ATLAS Priority Inversion

- Group-priority affects only order of negotiation
- When an analysis queue starts up after a quiet period, production starts to lose out. **Even though production is below its quota it loses slots to analysis jobs because they get negotiated for first.**
- Negotiation should stop for a queue that is over quota (w/ auto-regroup on) and there are other queues with waiting jobs below their quotas.

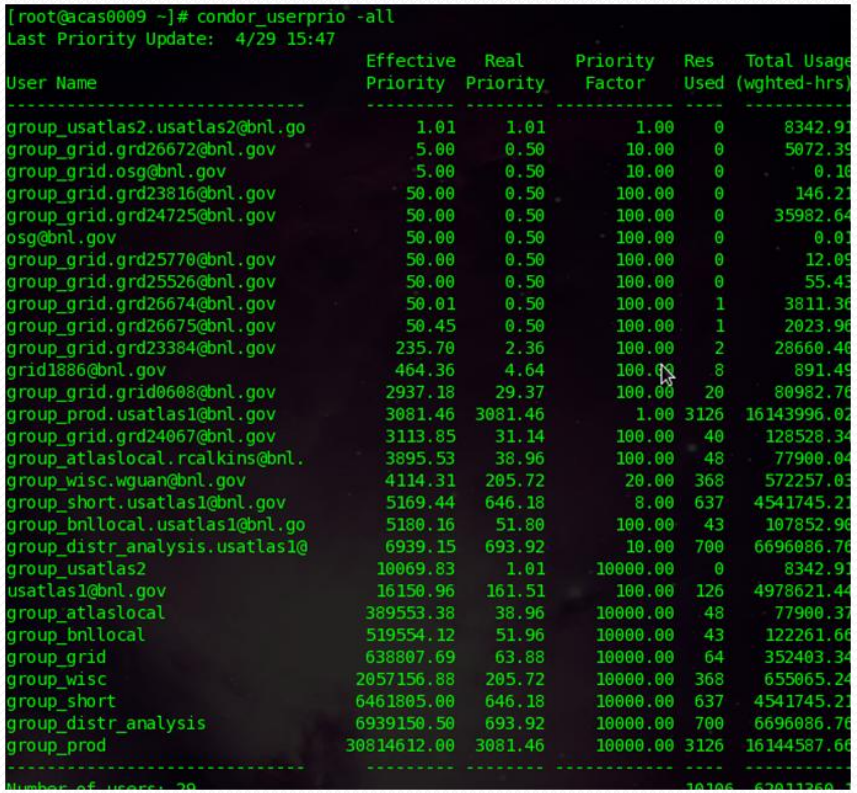


Problem area

ATLAS Priority Inversion

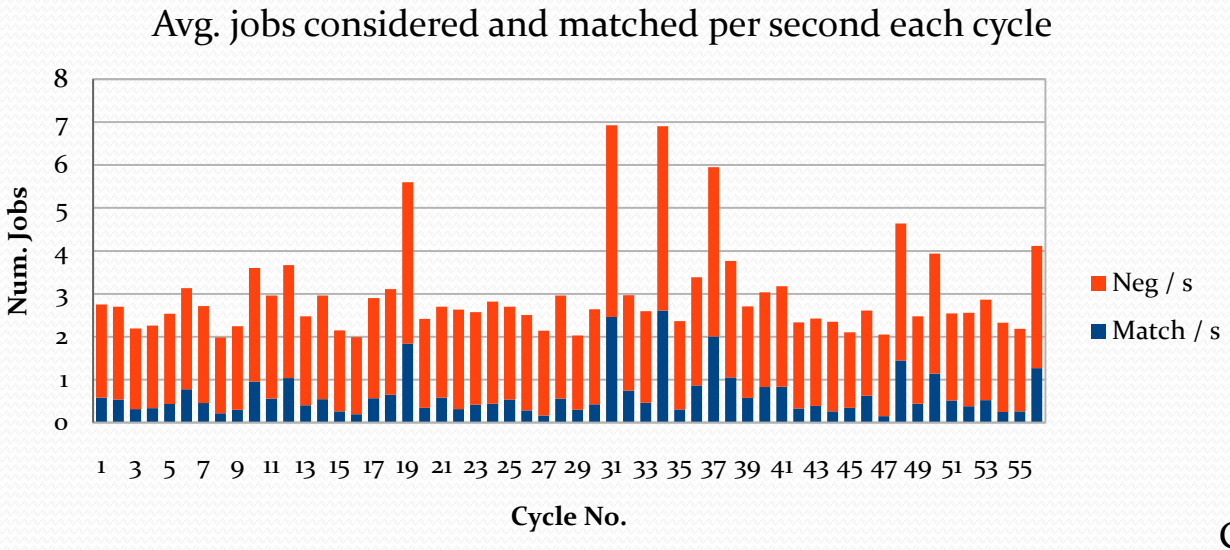
- Solution? *Increasing the spread of priority factors as more lots get added to production.* Required spread scales with size of the largest queue, and if another queue quiesces for long enough it will outrank production

- E.g. Production goes from 3k to 4k slots: usage increases 33% making its priority that much worse and an inversion that much more likely to occur…



```
[root@acas0009 ~]# condor_userprio -all
Last Priority Update:  4/29 15:47
                                    Effective   Real      Priority   Res   Total Usage
User Name                           Priority  Priority    Factor    Used (wghted-hrs)
------------------------------      --------- --------  ---------- ----  -----------
group_usatlas2.usatlas2@bnl.go          1.01     1.01        1.00    0       8342.91
group_grid.grd26672@bnl.gov             5.00     0.50       10.00    0       5072.39
group_grid.osg@bnl.gov                  5.00     0.50       10.00    0          0.10
group_grid.grd23816@bnl.gov            50.00     0.50      100.00    0        146.21
group_grid.grd24725@bnl.gov            50.00     0.50      100.00    0      35982.64
osg@bnl.gov                            50.00     0.50      100.00    0          0.01
group_grid.grd25770@bnl.gov            50.00     0.50      100.00    0         12.09
group_grid.grd25526@bnl.gov            50.00     0.50      100.00    0         55.43
group_grid.grd26674@bnl.gov            50.01     0.50      100.00    1       3811.36
group_grid.grd26675@bnl.gov            50.45     0.50      100.00    1       2023.90
group_grid.grd23384@bnl.gov           235.70     2.36      100.00    2      28660.40
grid1886@bnl.gov                      464.36     4.64      100.00    8        891.49
group_grid.grid0608@bnl.gov          2937.18    29.37      100.00   20      80982.76
group_prod.usatlas1@bnl.gov          3081.46  3081.46        1.00 3126   16143996.02
group_grid.grd24067@bnl.gov          3113.85    31.14      100.00   40     128528.34
group_atlaslocal.rcalkins@bnl.       3895.53    38.96      100.00   48      77900.04
group_wisc.wguan@bnl.gov             4114.31   205.72       20.00  368     572257.03
group_short.usatlas1@bnl.gov         5169.44   646.18        8.00  637    4541745.21
group_bnllocal.usatlas1@bnl.go       5180.16    51.80      100.00   43     107852.90
group_distr_analysis.usatlas1@       6939.15   693.92       10.00  700    6696086.76
group_usatlas2                      10069.83     1.01    10000.00    0       8342.91
usatlas1@bnl.gov                    16150.96   161.51      100.00  126    4978621.44
group_atlaslocal                   389553.38    38.96    10000.00   48      77900.37
group_bnllocal                     519554.12    51.96    10000.00   43     122261.66
group_grid                         638807.69    63.88    10000.00   64     352403.34
group_wisc                        2057156.88   205.72    10000.00  368     655065.24
group_short                       6461805.00   646.18    10000.00  637    4541745.21
group_distr_analysis              6939150.50   693.92    10000.00  700    6696086.76
group_prod                       30814612.00  3081.46    10000.00 3126   16144587.66
------------------------------      --------- --------  ---------- ----  -----------
Number of users: 29                                                10106  62011260
```

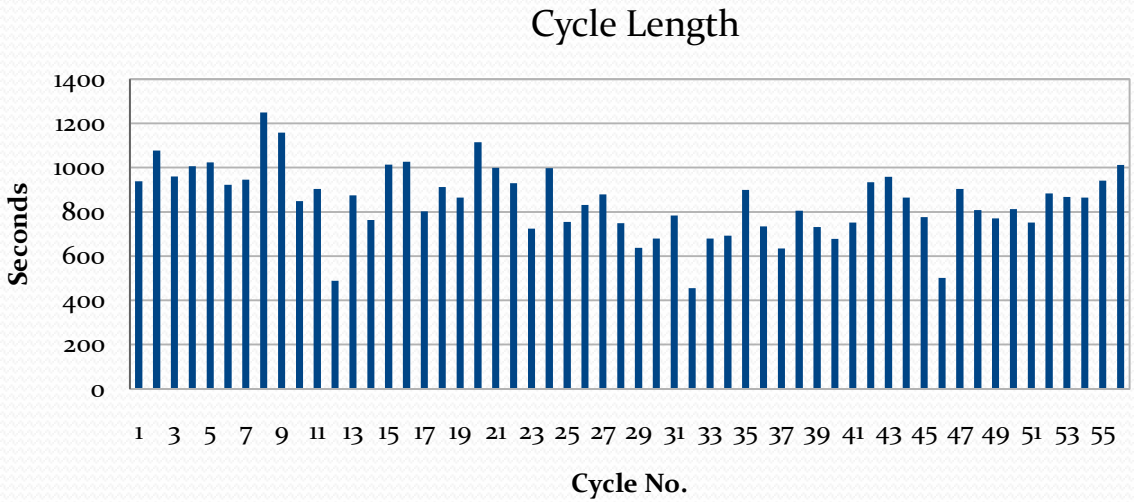
Negotiator Performance



Performance from one day of the STAR negotiator

Cycle occasionally blocks waiting for a schedd (many successes near the default 30s timeout)

In case where many submitters are on many machines each, much wasted time



Issues with Scheduler/Negotiator

- User frequently polling large queue
  - Schedd would fork a child which would use 5-10s of CPU time to answer query (1.6Gb Size!)
- Auto-clustering sometimes doesn't skip similar jobs
- Globally-scoped ClassAds would be nice, e.g. for the usage of a shared scratch NFS filesystem

Puppet-ize Configuration Files

- New Puppet-based centralized configuration management system of general-purpose servers
- Will templatize condor configuration

```
class ldap {
  # First make sure nscd is included:
  include nscd
  case $net_domain {
    ATLAS: {
      $ldap_servers = [ 'aldap01.usatlas.bnl.gov', 'aldap02.usatlas.bnl.gov', 'aldap03.usatlas.bnl.gov' ]
      $ldap_home_dir = 'homeDirectoryAtlas'
    }
    RHIC: {
      $ldap_servers = [ 'rldap01.rcf.bnl.gov', 'rldap02.rcf.bnl.gov', 'rldap03.rcf.bnl.gov', 'rldap04.rcf.bnl.gov',
                        'rldap05.rcf.bnl.gov', 'rldap06.rcf.bnl.gov', 'rldap07.rcf.bnl.gov' ]
    }
    default: {
      $ldap_servers = [ ]
    }
  }
  case $operatingsystem {
    RedHat: {
      case $operatingsystemrelease {
        /^5/: {
          $package_names = [ 'nss_ldap' ]
          $ldap_template = "ldap/ldap.conf-rhel5.erb"
          $ldap_conf_path = '/etc/ldap.conf'
          $nsswitch_file = "puppet:///modules/ldap/nsswitch.conf-rhel5"
          $service_names = [ ]
        }
        /^6/: {
          # RHEL6: can use either nss-pam-ldapd OR sssd!
          $package_names = [ 'nss-pam-ldapd' ]  # sssd
          $ldap_template = 'ldap/nslcd.conf-rhel6.erb'
          $ldap_conf_path = '/etc/nslcd.conf'  # /etc/sssd/sssd.conf
          $nsswitch_file = "puppet:///modules/ldap/nsswitch.conf-rhel6"  # Replace ldap with sss in this file!
          $service_names = [ 'nslcd' ]  # sssd
        }
        default: {
          $package_names = ''
          $ldap_template = ''
          $ldap_conf_path = ''
          $nsswitch_file = ''
          $service_names = [ ]
        }
      }
    }
```

- Configuration done using a Ruby-based object-oriented templating language
- Suitably scary at first…but worth the effort

http://www.puppetlabs.com

Motivation to use Puppet

- Configuration is similar in structure between experiments
  - Memory Limits for regular and flocked jobs
  - Preemption/Retirement-time on a per-job-type basis
  - Policy expressions (RANK/START/etc…)
  - List of currently blocked users
- Recent blocking/unblocking of users took editing 6 different files and a reconfig everywhere
- Using Puppet would separate each logical entity, making it easy to change things on a *per-entity basis*, and would automate pushing of changes and reconfiguration. All changes versioned in git—accountability and reliability

Thanks!

Questions? Comments?

CRS Job System

- Written in Python, submits to condor
- Asynchronous IO for staging data from tape
- Stages in and out are done outside of condor
  - Previously done with extra slots, not good aesthetically and otherwise
  - Can combine stage requests for jobs intelligently
- Abstraction layer for IO, similar to plugins
- Own basic workflow tools—DAGs not suitable

ATLAS Shared Pool

- Allow smaller ATLAS queues and OSG grid jobs to run in their own queue that can utilize a small shared pool of resources
- Implemented with Group Quotas
- Jobs “compete” for extra slots made available by ATLAS
- Necessitates adding AG flag by users (small enough it works)