AGLT2 Site Report

Wenjing Wu / UM
Dan Hayden, Philippe Laurens, MSU
Shawn Mckee, UM

Nov 4, HEPiX Fall 2024

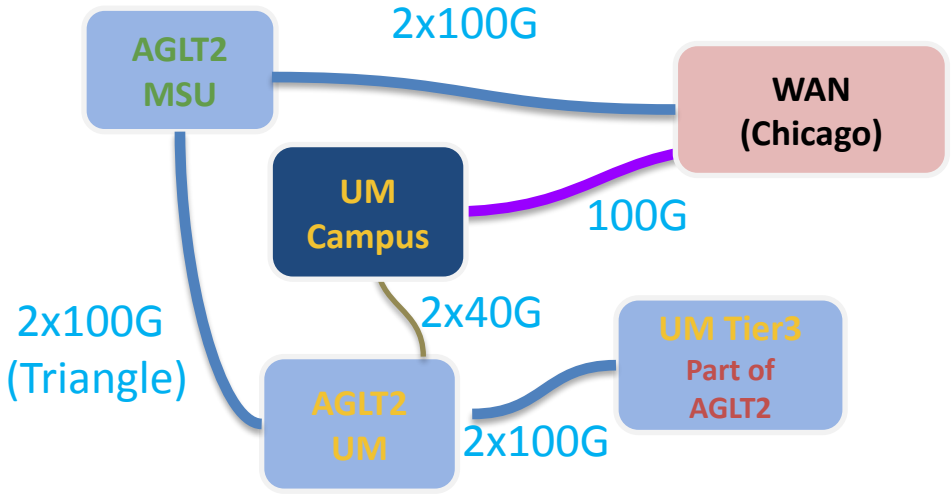
ATLAS Great Lakes Tier 2
UNIVERSITY OF MICHIGAN AGLT2 MICHIGAN STATE UNIVERSITY

Outline

- Site overview
- Transition from CentOS 7 to RHEL 9
- Transition from CFEngine to Ansible
- BOINC on RHEL 9
- Improve job slots occupancy
- Operation issues
- Summary

AGLT2 Overview

- AGLT2 (ATLAS Great Lake Tier-2) is an LHC Tier-2 Computing Center for ATLAS, located at our UM site (University of Michigan) and MSU site (Michigan State University).
  - What VO(s) we serve
    - **ATLAS** Tier2/Tier3
    - **OSG** (ligo, uscms, glow etc.)
  - Resource overview as of the end of 2024
    - 18.5K cores/253 kHS CPU, 19 PB dCache Storage
  - Resource Usage:
    - Over 92% are constantly used by ATLAS Tier-2 jobs
    - the rest is shared with UM ATLAS Tier3 and other VOs
  - Resilient **100G** path-diverse between the 2 sites and to Chicago (ESnet)



Software versions at AGLT2

- We do firmware and kernel updates every quarter which requires rebooting nodes
- dCache and HTCondor are also updated quarterly.
- Software version
  - dCache 9.2.27
  - HTCondor
    - UM, on RHEL9, mixed 23.10.0 and 23.0.12, 23.10.0 is used to test the new cgroup2 to work with BOINC jobs
    - MSU, on EL7, 10.0.9
  - Provision: Satellite server/capsule server 6.15
  - Configuration: Ansible 7.7

Transition to RHEL 9

- Choice of OS after CentOS 7-> RHEL 9
  - Both **MSU** and **UM** have licenses for RHEL through the University and satellite servers are hosted by IT
  - RHEL 9+ gives a modern kernel, compilers, and improvements for data transfers and long lifetime, quicker security fix.
- Challenges from transitioning from cobbler to the UM Satellite Server (version 6.15 on EL8)
  - The AGLT2 network is not routed to access the College/University Satellite server
  - The College/University Satellite server is not set up support PXE booting
- Solution
  - We deployed a capsule server (~Foreman SmartProxy) as a proxy between the AGLT2 network and the College Satellite server
  - Settled on using UEFI boot as the bootstrap mechanism (we were using Legacy BIOS because cobbler's lack of support for secure boot)
    - 90% nodes support UEFI (Newer ones support UEFI HTTP(s), older ones only UEFI PXE)
    - 10% very old nodes, only support legacy BIOS.
    - Enable LACP fallback for bonded NICs on the switch end, the NIC on the lowest switch port will get the DHCP IP assigned to (if both ports have the same priority). A bit tricky, because one needs to put the MAC address of the NIC which gets the IP as the MAC of the bond interface in the host profile.

Progress on RHEL 9 transition

- Progress
  - We finished rebuilding all the Tier2 nodes (dcache and HTCondor and interactive) at the UM site by the end of June (EOL for CentOS 7), but there are still other servers, like AFS, Lustre, DNS, waiting to be upgraded to the RHEL 9 version.
  - Lustre server version does not have anything ready for EL9 yet, only clients support, we tried to build from source, success building lustre kernel, but miss zfs support.
  - MSU site still working on making its RH Capsule node operational — delayed start due to initial lack of enthusiasm from IT; but fully supportive after decision made. Then faced installation and implementation hurdles:
    - Followed nice guide from RH, but expected subscription for infrastructure software was not visible from Capsule. Ticket with RH. Resolution not totally clear. Installation started with Satellite at V6.12, later updated to 6.13 and instructions are different. Further difficulties when configuring software, with certificates. Currently still working through provisioning first node.

Transition from CFEngine to Ansible (1)

AGLT2 has been using CFEngine since 2007, 17 years of legacy code

- We need to rewrite everything in Ansible
- Timeline
  - Starting from early March, takes about 1 month to learn about Ansible and its available resources.
  - started to write Ansible playbooks as we started to rebuild EL9 work nodes.
  - By the end of June, we have finished all the Ansible playbooks to configure all Tier2 nodes, including work nodes/login nodes/dcache nodes.
- Design
  - Server: Ansible 7.7 on RHEL 9.4
  - Use the ansible pull architecture (better scalability), use subversion as the repository
  - Use nmap plugs in to build inventory (get inventory of all hosts on the defined subnet and put them in host groups)

Transition from CFEngine to Ansible (2)

- Design (cont.)
  - organize playbooks and configurations by functionalities, i.e, configure SSH, AFS, Condor, dCache etc
  - For each function, there is one playbook and one configuration file directory, and it can have different rules for different host groups
  - A node could be in several different host groups and this is defined in the inventory files.
  - can run ansible pull on specific functions, i.e , run the sshd.yaml to configure ssh on any nodes, different rules will be applied based on the host groups the node belongs to.

```
[root@umansb ~]# ls /root/ansible/inventory_public/
01-nmap-um.yaml  02-nmap-msu.yaml  03-agl.yaml  04-service.yaml
[root@umansb ~]# ls /root/ansible/playbooks/
actmon.yaml    check.yaml    dcache.yaml    generalinfo.yaml  libs            networkinfo.yaml   passwd_group.yaml  test.yaml
afs.yaml       checkmk.yaml  dell.yaml      group_vars        login.yaml      nftables.yaml      sample.yaml        test2.yaml
ansible.yaml   chrony.yaml   dnf.yaml       hostcert.yaml     lustre.yaml     nftables.yaml.bak  sshd.yaml          yumrepo.yaml
autofs.yaml    condor.yaml   fail2ban.yaml  krb5.yaml         main.yaml       osgwn.yaml         sshkey.yaml
boinc.yaml     cvmfs.yaml    fstab.yaml     levlab.yaml       networkconf.yaml  package.yaml       sudoer.yaml
[root@umansb ~]# ls /root/ansible/configs/
actmon  ansible  boinc  checkmk  cvmfs  dell      hostcert  login   networkconf  osgwn         ssh
afs     autofs   check  condor   dcache  fail2ban  krb5      lustre  osg ce       passwd_group  yumrepo
```

BOINC on RHEL 9

- We have been running BOINC/ATLAS@home jobs on our Condor cluster in backfilling mode since March 2019, it has helped to improve the overall CPU Utilization of the cluster. (reported in 2019 spring HEPiX)
- Some efforts were needed to make BOINC to continue to run on RHEL 9
  - Non trivial work to recompile the BOINC client on the RHEL 9 environment.
  - A lot of control/monitoring scripts had deprecated python 2 syntax, need to "translate" them into python 3 syntax.
  - Need to upgrade condor to condor 23.10.X which supports one writer cgroup tree, to reduce the impacts to the Condor jobs's CPU efficiency by BOINC jobs. Future work is still necessary to measure and reduce the impacts to the CPU Efficiency of condor jobs. On EL7, we were able to manage the CPU Efficiency loss under 5%.

Tuning to improve job slots occupancy (1)

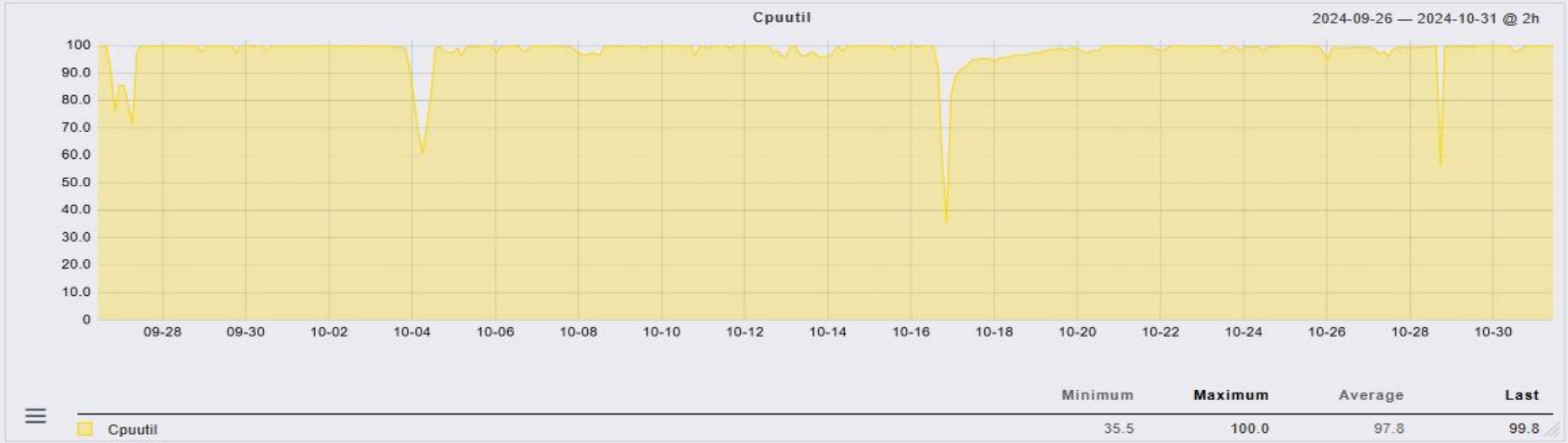
- Why
  - Avg. job slots occupancy had been 99.5% for the past 5 years after tuning
  - job slots Occupancy = total_claime_js/total_online_js
  - From the beginning of 2024, AGLT2 became a test site to allow an unified ATLAS PanDA queue to submit jobs with dynamic memory requests (both single and multi core jobs request memory/core ranging from 1GB~6GB, queue has avg memory/core 2.3GB set while avg. RAM/core is 3.3 GB for the AGLT2 cluster)
  - dynamic memory jobs causes several problems:
    - job slots with memory starvation (high memory jobs concentrate on the same work nodes, and used all the available memory, half of the CPUs are not claimable without available memory)
    - the dynamical ratio changing between single core and multi core jobs often cause low job slots occupancy, as a significant sudden drop of single core jobs would cause the fragmented job slots (1<=js_cores<8) to stay idle until there are enough to run a mcore jobs (js_cores=8).

Tuning to improve job slots occupancy (2)

- to address defragmentation of job slots, our approach is to reduce the number of work nodes that could be fragmented.
  - set one set of work nodes (30% of the total job slots) , define a tag “allow_score” = True (Boolean value) on job slots
  - have a script to dynamically change the job router rules of Condor-CE, with requirements like *Target.allow_score==True* for single core jobs when there is a low volume of score jobs in queue, and remove this condition when score jobs ramps up and mcore jobs ramps down.
  - When the rule *Target.allow_score==True* is applied, score jobs can only run the subset of the work nodes which have allow_score=True defined.
  - When the rule *Target.allow_score==True* is removed, score jobs can run on any work nodes, we need it because there are times 90% of the jobs are score jobs, without the rule removed, the cluster would be draining.

Improvement of occupancy after tuning

- Avg. job slots occupancy is 97.8% (Including a few Dips/draining caused by site problems) , still need improvement.
- fragmentation still exists, with tuning, we reduce its impacts(drop of occupancy) from over 10% to 3%.



Tuning to reduce starved job slots

- to address memory starvation for job slots, the goal is to control the number of hmem jobs on work nodes depending on the hardware configuration (RAM/core ratio)
  - on work nodes, define resource type LMEMTask, similar to the the amount of CPU/Memory resources, it can be defined/detected and claimed upon request
  - On the job router, RequestLMEMTask is set as the same number of RequestCPUs if job's memory/core >= 6GB/core (a threshold value we define)

    `RequestLMEMTask=8` for mcore jobs requesting 6GB/core

    `RequestLMEMTask=1` for score jobs requesting 6GB/core
  - Work nodes allow different number of LMEMTask depending on the RAM/core ratio from hardware configuration. High RAM/core ratio work nodes allow more.

```
MACHINE_RESOURCE_NAMES = LMEMTask
MACHINE_RESOURCE_LMEMTask = int($(DETECTED_CORES))
JOB_DEFAULT_REQUESTLMEMTask = 0
SLOT_TYPE_3_CONSUMPTION_LMEMTask = ifThenElse(target.RequestLMEMTask =!=
undefined, target.RequestLMEMTask, 0)
SLOT_TYPE_3 = cpus=$(DETECTED_CORES),LMEMTask=30%
```

The number 30% is calc. based on RAM/core ratio for each WN

this work node only allows 30% cores to run high memory jobs

Large Mem Task limit on the WN

```
[root@c7-[redacted]3 ~]# condor_status -af name detectedmemory/detectedcpus memory/2/cpus detectedcpus detectedlmemtask cpus lmemtask -constr 'state=?="Claimed"'|grep c7-[redacted]3
slot1_1@c7-[redacted]3.aglt2.org 4015 1000 64 64 8 0
slot1_2@c7-[redacted]3.aglt2.org 4015 6016 64 64 1 1
slot1_3@c7-[redacted]3.aglt2.org 4015 1984 64 64 1 0
slot1_4@c7-[redacted]3.aglt2.org 4015 6016 64 64 1 1
slot1_5@c7-[redacted]3.aglt2.org 4015 1984 64 64 1 0
slot1_6@c7-[redacted]3.aglt2.org 4015 6016 64 64 1 1
slot1_7@c7-[redacted]3.aglt2.org 4015 1984 64 64 1 0
slot1_8@c7-[redacted]3.aglt2.org 4015 1000 64 64 8 0
slot1_9@c7-[redacted]3.aglt2.org 4015 2000 64 64 8 0
slot1_10@c7-[redacted]3.aglt2.org 4015 1984 64 64 1 0
slot1_11@c7-[redacted]3.aglt2.org 4015 1984 64 64 1 0
slot1_12@c7-[redacted]3.aglt2.org 4015 6016 64 64 1 1
slot1_13@c7-[redacted]3.aglt2.org 4015 1984 64 64 1 0
slot1_14@c7-[redacted]3.aglt2.org 4015 6016 64 64 1 1
slot1_15@c7-[redacted]3.aglt2.org 4015 2048 64 64 1 0
slot1_16@c7-[redacted]3.aglt2.org 4015 6016 64 64 1 1
slot1_17@c7-[redacted]3.aglt2.org 4015 6016 64 64 1 1
slot1_18@c7-[redacted]3.aglt2.org 4015 1984 64 64 1 0
slot1_19@c7-[redacted]3.aglt2.org 4015 6016 64 64 1 1
slot1_20@c7-[redacted]3.aglt2.org 4015 1984 64 64 1 0
slot1_21@c7-[redacted]3.aglt2.org 4015 6016 64 64 1 1
slot1_23@c7-[redacted]3.aglt2.org 4015 1984 64 64 1 0
slot1_24@c7-[redacted]3.aglt2.org 4015 1984 64 64 1 0
slot1_25@c7-[redacted]3.aglt2.org 4015 6016 64 64 1 1
slot1_26@c7-[redacted]3.aglt2.org 4015 6016 64 64 1 1
slot1_27@c7-[redacted]3.aglt2.org 4015 6016 64 64 1 1
slot1_28@c7-[redacted]3.aglt2.org 4015 2048 64 64 1 0
slot1_29@c7-[redacted]3.aglt2.org 4015 2048 64 64 1 0
slot1_31@c7-[redacted]3.aglt2.org 4015 6016 64 64 1 1
slot1_32@c7-[redacted]3.aglt2.org 4015 2048 64 64 1 0
slot1_33@c7-[redacted]3.aglt2.org 4015 1984 64 64 1 0
slot1_34@c7-[redacted]3.aglt2.org 4015 6016 64 64 1 1
slot1_35@c7-[redacted]3.aglt2.org 4015 6016 64 64 1 1
slot1_36@c7-[redacted]3.aglt2.org 4015 6016 64 64 1 1
slot1_37@c7-[redacted]3.aglt2.org 4015 1984 64 64 1 0
slot1_38@c7-[redacted]3.aglt2.org 4015 6016 64 64 1 1
slot1_39@c7-[redacted]3.aglt2.org 4015 2048 64 64 1 0
slot1_40@c7-[redacted]3.aglt2.org 4015 1984 64 64 1 0
slot1_41@c7-[redacted]3.aglt2.org 4015 6016 64 64 1 1
slot1_42@c7-[redacted]3.aglt2.org 4015 1984 64 64 1 0
slot1_43@c7-[redacted]3.aglt2.org 4015 6016 64 64 1 1
[root@c[redacted] ~]# condor_status -af name detectedmemory/detectedcpus memory/2/cpus detectedcpus detectedlmemtask cpus lmemtask -constr 'state=?="Claimed"'|grep c[redacted]3|awk 'BEGIN{a=0}{a+=
19
[root@[redacted] ~]# echo 19/64|bc -l
.29687500000000000000
[root@c7-[redacted]3 ~]# grep "LMEMTask=" /etc/condor/config.d/* -h
SLOT_TYPE_3 = cpus=$(DETECTED_CORES), AnalyTask=100%, LMEMTask=30%
```

for jobs requesting 6016 MB/core, LMEMTask=CPUs (score job, lmemtask=cpus=1; mcore jobs, lmemtask=cpus=8), otherwise, LMEMTask's value is 0

19 CPU cores are running LMEMTaks, that is about 29.6% of the total CPU cores

WN configuration allows 30% CPU cores to run LMEMTask

Improvement after tuning (1)

We only implemented LMEMTask on the EL9 nodes,and EL9 nodes do not see starved job slots while there is a spike of LMEM jobs in the queue.

Improvement after tuning (2)

- very significant drop of starved job slots, from 225 to single digits
- starved job slots: job slots do not have any memory available.

Operation issues

- CVMFS problems
  - always have occasional I/O errors on different repos, always hand problematic nodes to cvmfs experts to debug
  - It seems the recent cvmfs version 2.11.5 has more errors
  - did tunings, increase the cache size (from 26GB to 50+GB), increased autofs mount timeout (to 5 min)
  - Debug conclusion(nothing definite): likely to be caused by network or cache server glitches.
  - Implemented scripts to do automatic recovery via cvmfs reload/wipecache/killall etc.

Summary

- Updates of OS, software, firmware and security patches are applied in a timely way to keep AGLT2 updated
- Finished most of the migration work to RHEL 9 and Ansible
- Tuning on Condor and Condor-CE to improve occupancy
- Got BOINC/ATLAS@home jobs on EL9.
- **FUTURE:**
  - Finish upgrades to RHEL 9 , Choices about VMWARE
  - keep tuning Condor/Condor-CE to improve job slots/ memory/ disk occupancy
  - Participate in mini Data Challenge testing ahead of DC26/27
  - Add alerting to existing SOC/Zeek installations at both sites

Questions ?