EPJ Web of Conferences **214**, 08020 (2019) https://doi.org/10.1051/epjconf/201921408020
CHEP 2018

Establishment of new WLCG Tier Center using HTCondor-CE on UMD middleware

Geonmo Ryu[1,*] and *Seo-Young* Noh[1]

[1]Korea Institute of Science and Technology Information, 245 Daehak-ro, Yuseong-gu, Daejeon, 34141, Korea

Abstract. CREAM is a CE program used by many sites based on UMD middleware to handle grid computing jobs. CREAM can be combined with various local batch programs, but it did not work with the HTCondor batch program. Since KISTI has been using HTCondor for many years, we searched another CE program which can work with HTCondor to process grid computing tasks. We proposed HTCondor-CE as the CE program, but it was not smoothly compatible with the UMD middleware environment because the OSG developed the HTCondor-CE software. In 2017, we manually configured HTCondor-CE and deployed a new CMS Tier-2 Center using the HTCondor-CE and a local HTCondor cluster in the UMD middleware environment. The center has passed a SAM test suite that confirms the CMS computing performance of the center.

1 Introduction

A CE (Computing Element) program is a grid job manager program. It collects information about the computing resources from the LRMS (Local Resource Management System) like the torque job manager. It also receives grid job requests from the central job management server and translates the grid job requests into local job requests according to LRMS. The CREAM [1] was a popular CE, and it belonged to UMD middleware[2]. CREAM can combine with various local batch programs. However, it did not support the HTCondor [3] as LRMS CREAM. We have been accustomed to HTCondor because we have used HTCondor clusters for many years. So we tried to use HTCondor as the LRMS for the new CMS Tier-2 Center.

There are two methods to use HTCondor LRMS in UMD middleware environment. One method is to use ARC-CE [4] and the second method is to use HTCondor-CE [5]. RAL Tier-1 center is a computing center to use ARC-CE. RAL Tier-1 center was a first center to support HTCondor LRMS in UMD middleware. They reported using the ARC-CE at pre-GDB on Batch Systems workshop in 2014 [6]. On the other hand, CERN and PIC Tier-1 centers are representative centers using HTCondor-CE. CERN announced the success stories using the HTCondor-CE and HTCondor LRMS at HTCondor/ARC CE Workshop in 2016 [7]. CERN also provided puppet codes so that other sites could easily configure HTCondor-CE [8]. PIC Tier-1 center also built the HTCondor-CE server for the LHCb experiment in September 2016 and registered with BDII [9]. According to GOCDB information [9], a total of 7 centers have HTCondor-CE servers. (PIC, CERN, KISTI, CIEMAT, TIFR, Liverpool, and Czech)

*e-mail: geonmo@kisti.re.kr

2 Overview of KISTI CMS Tier-2 center

As shown in Figure 1, the KISTI Tier-2 Center has an HTCondor-CE server managing the HTCondor LRMS. HTCondor-CE changes the request received from the remote to fit the LRMS. Then, it submits the converted request to HTCondor LRMS. It also collects status information about computing resources and sends the information to the WLCG central management server. The Argus system is a program that authenticates the requested grid user and maps it to the appropriate local account. We also installed and configured the CVMFS server to deploy the CMS software. The dCache [10] system provides a storage pool that can store large amounts of data.

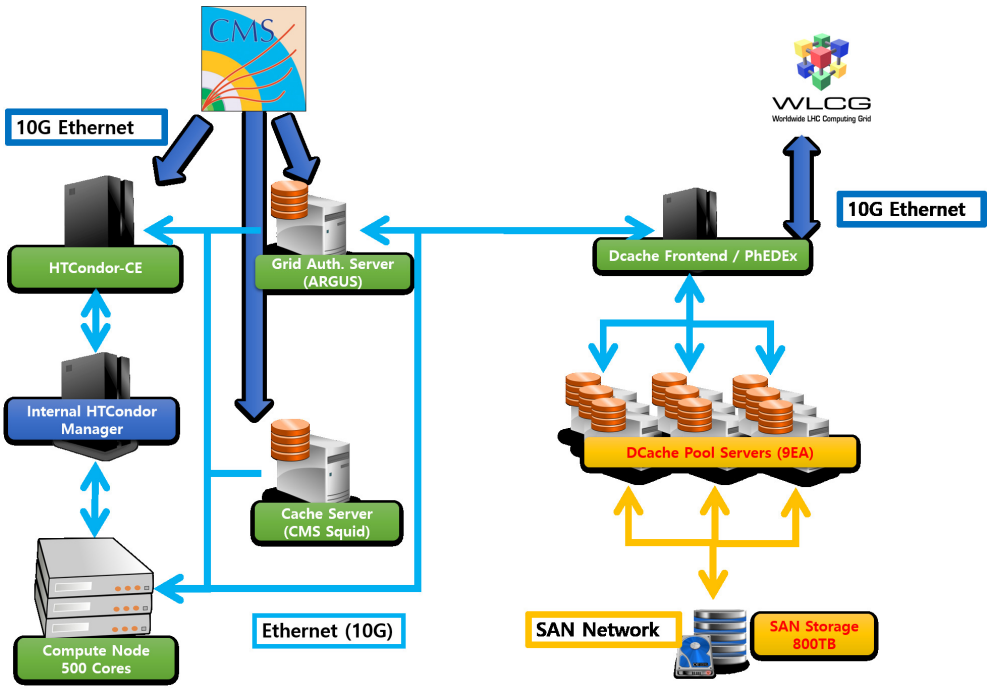

Figure 1. KISTI CMS The Tier-2 center consist of HTCondor-CE and dCache SE system.

3 System Setup

3.1 HTCondor-CE and HTCondor LRMS

The HTCondor-CE has been managed and developed by the OSG community. Therefore, UMD middleware does not provide an installation package of HTCondor-CE. We obtained the HTCondor-CE Source Redhat Package Management (SRPM) from the development site of HTCondor-CE [11]. To resolve the package dependency problem of the HTCondor-CE package, we needed to modify HTCondor-CE's spec file in SRPM. In particular, the name of the same package was different between OSG and UMD middleware, so we changed the requirements of SRPM for the certificate package (ca-policy-lcg in UMD vs. certificate in OSG).

HTCondor-CE settings were more complex than the installation steps. If you installed HTCondor-CE on the OSG middleware, you can configure HTCondor-CE settings via the *osg-configure* program. However, the osg-configure does not work on UMD middleware. So we had to manually modify the configuration file in the /etc/condorce/config.d directory. As CERN IT suggested in [7], they shared the configuration template of almost all environment variables related to HTCondor-CE as a puppet file format. We configured HTCondor-CE using the share information without the osg-configure program.

However, some settings needed to be changed depending on the environment of the center. We set the values as shown below.

- **/etc/condor-ce/condor_mapfile:** This file is used to authenticate valid grid users and convert them to local accounts. This file is related to the "Unified Mapfile for authentication" chapter in the HTCondor manual documentation [12]. In most cases, the grid users are mapped to the local accounts by "GSI (.*) GSS_ASSIST_GRIDMAP". When you are configuring the authentication system, you can temporarily move "CLAIMTOBE" entry upwards or add a dummy GSI entry to pass the authentication unconditionally if necessary. Our center used Argus to authentication. Therefore, we also modified the */etc/sysconfig/condor-ce* file and added the PEP-callout which is a subsystem of the Argus to provide the GSI mapping.

- **/var/lib/osg/osg-local-job-environment.conf:** There are environment variables that must be specified to run the CMS analysis code. For example, if *VO_CMS_SW_DIR* and *PATH* are not set correctly, the program did not work correctly and failed the site validation process, such as SAM testing. The environment variables written in this file are set when the job is submitted from HTCondor-CE to HTCondor LRMS. Administrators can set up files in the same format as the configuration files in the general environment variable configuration file /etc/profile.d. We have specified the /cvmfs directory as VO_CMS_SW_DIR and added the directory to the PATH.

- **/etc/condor-ce/config.d/61-job-routes.conf:** There are various experiments such as ATLAS, CMS, ALICE and LHCb only in the WLCG experiment. In other words, HTCondor-CE may be requested to work from various teams. In this case, HTCondor-CE's JobRouter manages the priority of each experiment and the number of work requests. You can modify the file to specify the total job quota and the maximum number of queues to keep. If the job request is larger than the maximum, the HTCondor-CE ignores the extra requests. We have set up a policy for ops, dteam, and cms group for our center.

- **/var/lib/bdii/gip/provider/condor_ce_bdii_generate_glue2.py:** HTCondor-CE collects resource information from HTCondor LRMS and delivers it to the site-BDII server. To this end, HTCondor-CE uses the LDAP daemon to publish the collected information and site-BDII to collect the published content. This file is a script that generates information to be passed to site-BDII. The htcondor-ce-bdii package contained these files. BDII information varies from site to site, so an administrator must verify the results of that file. When we built KISTI Tier-2, there was a minor problem. So we had to modify the file. We have used the "glue-validator" program to ensure that the content generated by the script is correct.

The HTCondor cluster, which is the LRMS of the KISTI CMS Tier-2 Center, was installed in the same way as a typical HTCondor cluster. We added the HTCondor YUM repo from HTCondor homepage and installed the HTCondor v8.6 as the production version. A CMS grid job requests 8 cores. Therefore, we have added machines as a dynamic slot type to handle the jobs. Each machine has a singularity program [13] for isolation testing and a gLExec program [14] for account mapping. In particular, gLExec needed a separate setup

to use Argus. We referenced the related webpage [15] for its configuration. If this setting is correct, the gLExec provides the local account for the remote user.

3.2 APEL

APEL is a program that sends accounting information to the WLCG. This information includes details such as the running time of the job and CPU usage. The results are included in the WLCG monitoring report and serve as a basis for analyzing how much resources are provided compared to the resources that each center pledges to provide. When we were building the CMS Tier-2 Center, the official APEL software did not have a feature to handle the HTCondor-CE. However, already PIC Tier-1 has modified the APEL to collect account information for the HTCondor LRMS [16]. Since APEL v1.6.0, the APEL team added the "HTCondor Parser" and "htcondor_acc.sh" script to acquire the job information using "condor_history". It filled the "EventRecords" in APEL database. PIC Tier-1 center added a new "HTCondorCE Parser" to update *BlahpRecords*. We have added some additional features based on the PIC code. The PIC Tier-1 APEL obtains a list of jobs from the LRMS via the "condor_history" command, while KISTI APEL uses the "condor_history" and "condor_ce_history" to retrieve the job lists from LRMS and HTCondor-CE. We guessed that it has attempted to solve duplicate accounting problems caused by the local jobs. However, in most cases, this difference can be ignored.

We knew that the Liverpool Tier-2 Center has been working to update APEL recently. They said that the PIC method had a problem about maintenance because it uses a different routine than using another LRMS [17]. They are being developed to generate the accounting files in the form of blahp file format and to fill the BlahpRecords using "blahp Parser" instead of the HTCondorCE parser. Although the complete code has not been submitted, we hope to be able to use the official program written in Liverpool Tier-2 in future APEL v1.8.0.

3.3 Argus

HTCondor allows a file-based authentication method called GRIDMAP. In the past, the site maintained the latest account information by updating the GRIDMAP file from the VO management server. Argus [18] is a program designed to support policies that are difficult to support by GRIDMAP authentification. Administrators can use the Argus PEP daemon to manage service usage rights for specific users. KISTI CMS Tier-2 sets the Argus to provide local account information to HTCondor-CE and dCache SE. Argus parses the information of the proxy certificate which was submitted by remote grid users. We installed the argus using UMD middleware repository. We also configured the default settings using the *YAIM* configuration tool. However, YAIM tool did not fully support UMD4 on CentOS7 system anymore [19]. So we had to set policy rules manually. We have added policy conditions for our system by referring to the example provided on the Argus home page [20].

4 Testing

We have registered the HTCondor-CE server as APEL and org.opensciencegrid.htcondorce service in GOCDB. Regrettably, EGI does not provide testing procedures for HTCondor-CE services. So we created a simple HTCondor-CE test script. This script was written to submit five jobs per day for testing. Test results indicate that HTCondor-CE has successfully transitioned to a local condor work. We built the test environment using the condor_ce_ping, condor_ce_run, or condor_ce_trace provided in the HTCondor-CE package [22]. These commands are very similar to those submitted in the purest form using the condor_ce_submit

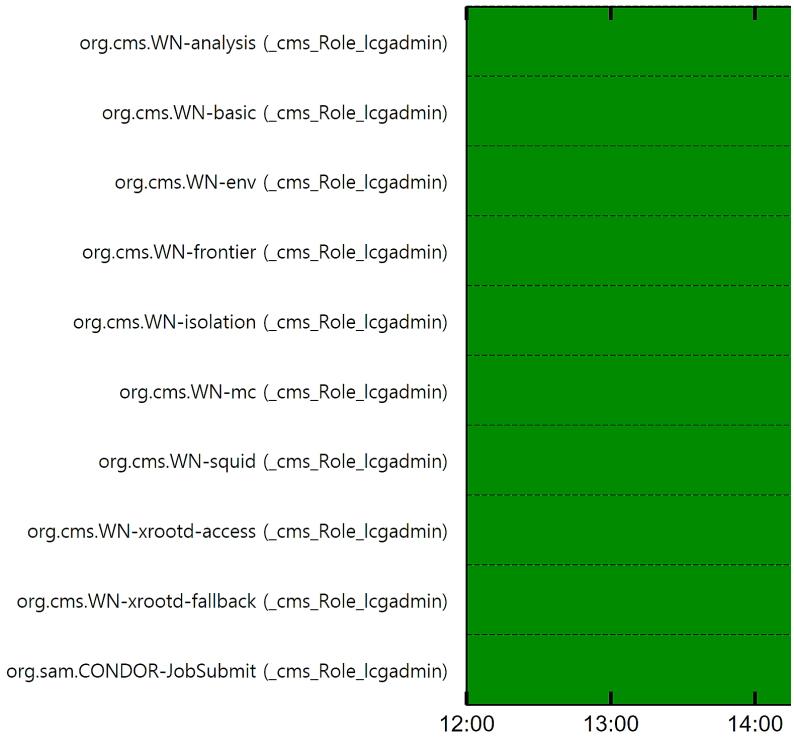

Figure 2. CMS SAM testing is testing many of the features provided by the CMS. This means that the services must be available on the calculation servers. Important services include remote access file protocols such as XRootD [21], container-based isolation program such as Singularity [13].

command. These commands make it easy to see that the job conversion between HTCondor-CE and HTCondor LRMS is smooth. Also, it is easy to check even when network problems occurred.

The HTCondor-CE server must pass the CMS SAM test [23] to process CMS operations as a Tier-2 center. The SAM test is used to test that the functions required to run CMS tasks are working correctly. Figure 2 shows the SAM test results for the HTCondor-CE server in the KISTI CMS Tier-2 Center. More information is available on the CMS TWiki page [24]. The tests used in Figure 2 are:

- **org.cms.WN-env:** This test verifies environment variables and storage status of WN. For example, if the working directory is too small to run the CMS software, this test must be failed.
- **org.cms.WN-basic:** This test is performed to check the basic state of the WN. Simple environment variables and the location of required files, and whether the GIT server is accessible.
- **org.cms.WN-mc:** This test verifies that the Monte Carlo job is executable on the WN. By default, the test is performed to make sure that the storage space is accessible.
- **org.cms.WN-squid and frontier:** These two tests are the same test. The squid test verifies that it works as a cache server regardless of the CMS software, and the frontier server verifies that it is acting as a frontier cache server in the CMS software.

- **org.cms.WN-xrootd-access and fallback:** These two tests check that the WN can access the xrootd server. The access test verifies that the data in your Tier Center is accessible and fallback verifies that the data at the other site is accessible. The site should join to CMS XRootD federation (CMS AAA) [25] to pass this test.

5 Conclusions and Future Work

We were able to successfully build a CMS Tier-2 center using HTCondor-CE in a UMD-based middleware environment. Although the automated tools were not present, HTCondor-CE could be utilized in a way that is not difficult. However, there is no clear documentation on how to install and configure HTCondor-CE in UMD middleware. There is no YUM repository to install HTCondor-CE conveniently. Similarly, APEL does not officially support HTCondor-CE. We will try to solve these problems.

References

[1] P. Andreetto, S. Borgia, A. Dorigo, A. Gianelle, M. Marzolla, M. Mordacchini, M. Sgaravatto, F. Dvorák, D. Kouril, A. Krenek et al., *CREAM: a simple, grid-accessible, job management system for local computational resources*, CHEP 2006, Mumbay, India (2006)

[2] The European Grid Initiative, *"Unified Middleware Distribution (UMD)" [software]*, version 4.1.3-1, 2018., Available from http://repository.egi.eu/category/umd_releases/distribution/umd-4/ [accessed 2018-10-10]

[3] M. Litzkow, M. Livny, M. Mutka, *Condor - A Hunter of Idle Workstations*, in *Proceedings of the 8th ICDCS* (1988)

[4] NORDUGRID project, *"ARC-CE" [software]*, version 1.0.7, 2018., Available from http://www.nordugrid.org/documents/arc-server-install.html [accessed 2018-10-15]

[5] OSG community, *"HTCondor-CE" [software]*, version 2.1.5-1, 2018., Available from http://koji-hub.batlab.org/koji/buildinfo?buildID=10202 [accessed 2018-10-10]

[6] A.D. Lahiff, *HTCondor* (2014), Available from https://indico.cern.ch/event/272785/contributions/1612799/ [accessed 2018-10-15]

[7] I.B. Steers, *HTCondor-CE and CERN* (2016), Available from https://indico.cern.ch/event/467075/contributions/1143807/ [accessed 2018-10-15]

[8] CERN, *Puppet configuration files which were provides by CERN* (2016), Available from https://github.com/cernops/puppet-htcondor_ce [accessed 2019-02-11]

[9] EGI, *HTCondor-CE Service in GOCDB*, Available from https://goc.egi.eu/portal/index.php?Page_Type=Services&serviceType=org.opensciencegrid.htcondorce&selectItemserviceType=org.opensciencegrid.htcondorce&ngi=&searchTerm=&production=TRUE&monitored=&certStatus=&scopeMatch=all&servKeyNames=&servKeyValue= [accessed 2019-02-11]

[10] P. Fuhrmann, *dCache, the Overview*, White paper (2004)

[11] OSG Community, *Information for build htcondor-ce* (2016), Available from https://indico.cern.ch/event/467075/contributions/1143807/ [accessed 2018-10-15]

[12] HTCondor Community, *HTCondor Manual* (2019), Available from http://research.cs.wisc.edu/htcondor/manual/v8.8/condor-V8_8_0-Manual.pdf [accessed 2019-02-13]

[13] Sylabs, *"Singularity" [software]*, version 2.6.1-1.1, 2019., Available from https://www.sylabs.io/ [accessed 2019-02-11]

[14] EGI, *"gLexec-wn" [software]* (2015), version 1.3.0, 2016., Available from http://repository.egi.eu/2016/11/23/glexec-wn-1-3-0-2/ [accessed 2019-02-14]

[15] Nikhef Wiki, *GLExec Argus Quick Installation Guide* (2013), Available from https://wiki.nikhef.nl/grid/GLExec_Argus_Quick_Installation_Guide#.2Fetc.2Flcmaps.2Flcmaps-glexec.db_for_use_with_the_Argus_Authorization_Service [accessed 2019-02-14]

[16] Steve Jones, *HTCondor-CE Accounting via APEL client software*, Available from https://twiki.cern.ch/twiki/bin/view/LCG/HtCondorCeAccounting#Set_up_some_scripts_crons_etc [accessed 2019-02-14]

[17] PIC Tier-1 Center, *"APEL with HTCondor-CE" [software]*, version 1.6.0, 2018., Available from https://github.com/jcasals/apel/blob/master/README.md [accessed 2019-02-14]

[18] Argus project, *"Argus" [software]*, version 1.7.0, 2018., Available from https://argus-documentation.readthedocs.io/en/stable/ [accessed 2018-10-23]

[19] The European Grid Initiative, *Middleware*, Available from https://wiki.egi.eu/wiki/Middleware [accessed 2018-10-23]

[20] Argus project, *Example of Authorization Requests and Policies*, Available from https://argus-documentation.readthedocs.io/en/stable/misc/examples.html [accessed 2019-02-14]

[21] XRootD project, *"XRootD" [software]*, version 4.8.4-1, 2018., Available from http://xrootd.org/ [accessed 2019-02-11]

[22] B. Bockelman, T. Cartwright, J. Frey, E. Fajardo, B. Lin, M. Selmeci, T. Tannenbaum, M. Zvada, *Commissioning the htcondor-ce for the open science grid*, in *JPCS* (IOP Publishing, 2015), Vol. **664**, p. 062003

[23] SAM project, *"SAM Monitoring" [software]*, version v1.7.01.wlcgmon6, 2018., Available from http://wlcg-sam-cms.cern.ch/templates/ember/#/historicalsmry/heatMap?profile=CMS_CRITICAL_FULL&site=T2_KR_KISTI&view=Test%20History [accessed 2018-10-23]

[24] D.W. Dykstra, *CMS SAM tests*, Available from https://twiki.cern.ch/twiki/bin/view/CMSPublic/CompOpsSAMTests [accessed 2019-02-14]

[25] Pradeep Jasal, *Operations Guide of AAA* (2018), Available from https://twiki.cern.ch/twiki/bin/view/CMSPublic/CompOpsAAAOperationsGuide [accessed 2019-02-14]