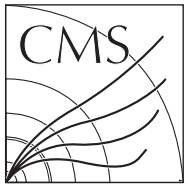
Available on CMS information server

CMS CR -2013/367



The Compact Muon Solenoid Experiment

Conference Report

Mailing address: CMS CERN, CH-1211 GENEVA 23, Switzerland

28 October 2013

Using ssh as portal - The CMS CRAB over glideinWMS experience

Stefano Belforte , Igor Sfiligoi , James Letts , Federica Fanzago , M D Saiz Santos , T Martin

Abstract

The User Analysis of the CMS experiment is performed in distributed way usingboth Grid and dedicated resources. In order to insulate the users from the details of computing fabric, CMS relies on the CRAB (CMS Remote Analysis Builder) package as an abstraction layer. CMS has recently switched from a client-server version of CRAB to a purely client-based solution, with ssh being used to interface with HTCondor-based glideinWMS batch system. This switch has resulted in significant improvement of user satisfaction, as well as in significant simplification of the CRAB code base and of the operation support. This paper presents the architecture of the ssh-based CRAB package, the rationale behind it, as well as the operational experience running both the client-server and the ssh-based versions in parallel forseveral months.

Presented at *CHEP2013 Computing in High Energy Physics 2013*

Using ssh as portal - The CMS CRAB over glideinWMS experience

S Belforte[1], I Sfiligoi[2], J Letts[2], F Fanzago[3], M D Saiz Santos[2], T Martin[2]

[1]INFN Sezione di Trieste, Via Valerio 2, 34127 Trieste, Italy
[2]University of California San Diego, 9500 Gilman Dr, La Jolla, CA 92093, USA
[3]INFN Sezione di Padova, Via Marzolo 8, 35131 Padova, Italy

E-mail: stefano.belforte@ts.infn.it

Abstract. The User Analysis of the CMS experiment is performed in distributed way using both Grid and dedicated resources. In order to insulate the users from the details of computing fabric, CMS relies on the CRAB (CMS Remote Analysis Builder) package as an abstraction layer. CMS has recently switched from a client-server version of CRAB to a purely client-based solution, with ssh being used to interface with HTCondor-based glideinWMS batch system. This switch has resulted in significant improvement of user satisfaction, as well as in significant simplification of the CRAB code base and of the operation support. This paper presents the architecture of the ssh-based CRAB package, the rationale behind it, as well as the operational experience running both the client-server and the ssh-based versions in parallel for several months.

1. Introduction

The Compact Muon Solenoid experiment (CMS) [1] is a general purpose detector built and operated at the CERN's Large Hadron Collider (LHC) by a community of over four thousand physicists to investigate fundamental particle physics in proton-proton collisions at the highest energy available in laboratory. LHC accelerates two counter-circulating proton beams at 7 TeV each and makes them intersect. When protons from the two beam undergo a nuclear interaction, their energy in converted into hundreds of elementary particles of various mass and life time. This process is called an event. Events happen inside CMS at a rate of about 500MHz and are captured by the CMS detector in a digital representation. A small fraction of those events, about 1KHz, is stored for offline reprocessing and analysis. Adding the original (raw) and the reprocessed data and the data from simulated events, overall CMS physicists have almost 50 PetaBytes of data available for studying the physics of proton-proton interaction.

Analysis of those data proceeds via hierarchical selection. Data are naturally grouped from the start in so called datasets. A dataset is a container of events collected (or simulated) under similar conditions and expected to have similar physics content. Typical datasets have a size of tens of TB's. Users run an initial step of event analysis and selection using a distributed computing system, such as the Open Science Grid (OSG)[2] and the European Grid Initiative (EGI)[3], to process one dataset at a time and derive smaller samples in the O(1-1000)GB range, which are eventually retrieved by the users (i.e. extracted from the CMS data management system and not further tracked by common tools) for iterated detailed analysis at local facilities.

A typical use case for physics analysis may require for the user to go through a few datasets from real data for event selections, a few others for efficiency and background studies, and up to tens of simulated datasets to compare with various theoretical models. Those data processing passes are called "workflows" and are usually performed at the CMS Tier2 sites on the WLCG, there are currently 56 such sites world-wide.

Within CMS Computing project, the Analysis Operations responsibility is part of the Physics Support office and has its focus on making it easy (ideally transparent) for users to run their workflows on the WLCG. Since 2004 CMS users have interfaced with Grid using a tool called CRAB (CMS Remote Analysis Builder) [4] which decomposes a workflow into a set of individual jobs and allows users to submit them for execution, track their status, and finally retrieve locally the output. Within CRAB the set of jobs used in one workflow is called a "task"

2. CRAB

CRAB was initially born as a pure client interface combining four main functionalities:

1) Look up CMS central databases for dataset details and location in order to do job splitting
2) Sandbox the user's executable and environment in order to replay it on remote CMSSW installations
3) Offer a convenience wrapper and an abstraction layer around Grid client
4) Keep track of which workflows (CRAB tasks) the user has ran.

CRAB was interfaced to Grid submission via gLite and to local submission to a few batch systems, notably HTCondor (mostly for use at the FNAL LPC farm) and LSF (mostly for use at the CERN CMS CAF farm). Adoption of the CRAB tool in place of custom scripts for functionalities 1), 2) and 4) above was very good, but submission to Grid has been exposing users to all sort of site and middleware issues.

In order to overcame the "Grid problems" CMS introduced in 2004 a client-server tool for CRAB, moving 1) and 3) above to the server side. Commands are sent to CRAB server over a GSI-authenticated SOAP connection, while input and output files are moved via gsiftp.

The CRAB server gave to users fast submission, equal if not better to local batch system and relieved users from the need to resubmit jobs failed due to random problems. CRAB server featured a simple table allowing Analysis Operation team to decide the retry policy among any combination of now or later, same site or different site, maximum number of retries, based on the exit code of the user's job. Also CRAB server was retrying three times aborted jobs in gLite, i.e. jobs where the user's application could not terminate.

Those automated resubmissions were extremely important in the first years, but as Grid became mature and stable their usefulness to users decreased and other problems became the focus of users and support attention.

CRAB server had been designed to be very powerful, flexible and general. But in time the CRAB server implementation proved to be too complex and fragile and bookkeeping problems inside the CRAB server itself surpassed those due to Grid failures. Development focused on building a new server based on a different architecture, common with the CMS Production tool, and effort on fixing the CRAB server shortcomings stopped in 2005. Analysis Operations kept up supporting CMS using community with the exiting CRAB server adding also support for submission to the glideinWMS [5] system.

3. glideinWMS and CRAB

glideinWMS is a pilot based job submission framework that presents the Grid to users as a single HTCondor batch pool. There are many advantages to this approach, like moving completely out of user's worry list all problems due to interaction with the site CE's. On the other hand

it requires a persistent service (HTCondor schedd process) on the submitting machines, so it does not allow users to submit from a remote machine, disconnect, and check on jobs later, as they could do with gLite. For this reason we introduced glideinWMS using the CRAB server approach. CMS Analysis Operations group deployed two servers at CMS Tier2 center at University of California, San Diego and operated those for four years. The HTCondor schedd's were running local to the servers and glexec [6] was used to do submissions in multi-user mode inside CRAB sever code. glexec is then used by glideinWMS pilots to switch temporarily to the user's identity when running user payload, thus providing the same secure insulation among users and between users and submission infrastructure as gLite.

Once we had user jobs running on glideinWMS we got rid of the largest part of grid issues and aborted jobs, the need for automatic resubmission decreased.

As of 2012, CMS analysis users and Analysis Operation were sort of cornered. Most Grid infrastructure problems were solved thanks to mature Grid middleware and experienced administrators at remote sites and to glideinWMS plus HTCondor, but we had introduced new problems due to CRAB server internals. Users satisfaction was very low. Users were forced to choose between large job resubmission effort to deal with gLite problems or suffer through CRAB serve problems which often lead to whole tasks to be lost needing resubmission. Support was having so many problems with the tool around glideinWMS that could not look at anything else. We were ready for change.

4. Introducing ssh

When we heard of the RCondor [7] project, we realized that there could be a way out. So we decided to adapt CRAB client to make use of it and try. The concept is strikingly simple: execute condor commands via ssh on a remote machine where HTC schedd runs. Original RCondor is using sshfs to make the local user file available to remote schedd and viceversa, but during implementation we found a performance penalty, especially when many files are involved and RTT between client and schedd is large, so we switched to moving tarballs with scp.

Given that CRAB already was build as client-server and already was submitting to local condor, adapting to submission to HTCondor via ssh was very simple and went quick and smoothly, user interface remained completely unchanged. A schematic view of the ssh-enabled CRAB client is in figure 1.

Since in the end we need GSI authentication, we used gsissh[8] as communication protocol between CRAB client and remote submission node where HTCondor schedd runs, remote grid identities are mapped to local unix users via LCMAPS call backs to GUMS in the US and ARGUS in Europe.

5. Implementation details

In order to put this into operations for our community we had to address a few technical issues.

5.1. Performance

We aimed to give users the same feel as working with a local batch manager, and largely succeeded. It was an incremental process, we started very simple and added a few things for faster reaction time along the way. Relevant actions for this where:

a) Avoid massive use of condor_q. We obtained from HTCondor team a new option for condor_history where only the local condor log file for each user task was parsed, this made for very fast crab -status. So user can now run it as often as they like, without excessively loading the schedd host.

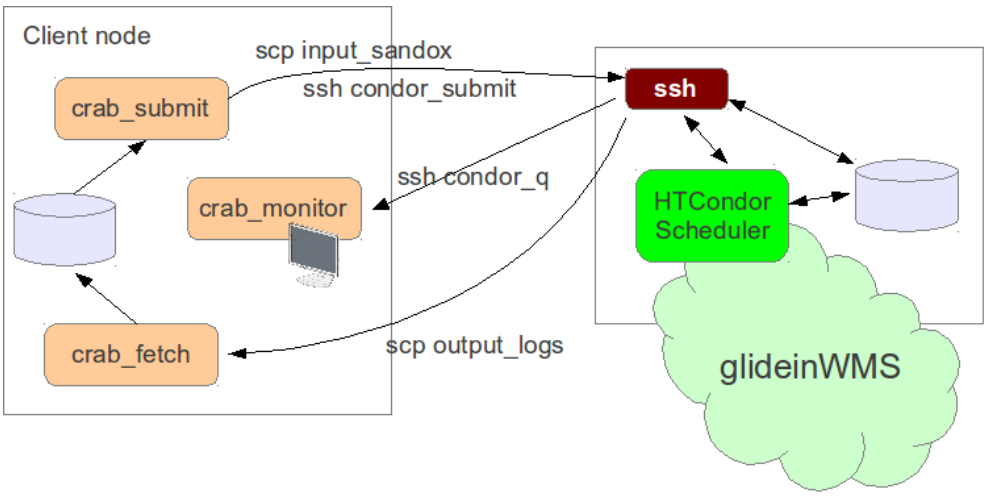


Figure 1. CRAB submission to glideinWMS using ssh

b) Avoid long delays due to gsi authentication. For this we reuse ssh sockets for command issued close in time by the same user, i.e. the ssh ControlPath feature. Proper handling of ControlPath including ControlPersist option is only available in OpenSsh 5.6 or later, not for the version distributed with gsissh in our grid clients. We emulated ControlPersist behaviour by submitting a remote sleep lasting 1hour and renewed when close to expire. This worked very well, initial connection can take up to ten seconds to establish e.g. between CERN and US West Coast, but after the first time all other crab commands are executed immediately. At times ssh connection terminates badly and leaves a corrupted socket (only fixed in OpenSsh 5.8 or later). This was cured by detecting the tell-tale error message in stdout, removing the socket and recreating the ControlPath automatically.

c) As a final performance boost, instead of using scp to copy back output files and logs from remote submission host to local UI one by one, we used rsynch over the existing ssh socket.

5.2. Security

Using ssh as a portal means that every CMS user can login interactively on the machines where HTCondor schedd's run. This is not an ideal situation but we decided to accept the associated risk and monitor how things go. The main concern was not hostile attack, there is nothing on the submission hosts that requires more security then any login node of the many multiuser interactive facilities CMS uses already world wide. Rather we feared well meaning users who could decide to log on the submission hosts to short cut the CRAB interface and directly submit/monitor or even look at output files, thus adding uncontrolled load on the machine. Here the work on the performance of the ssh connection payed off, users have been happy with the functionality provided. and we never saw any sign of abuse of the system.

As a minor note, ControlPath functionality in OpenSsh does not work if the user's .ssh directory is on AFS. Since this is the situation at CERN, i.e. for majority of our users, we had to created an ad-hoc directory in /tmp to host the socket and took special care in making sure the directory and the socket have the correct ownership and only owner can access them.

5.3. Scalability

CMS Analysis load is about 200k job submission daily from roughly 200 different users, with 40k jobs running at any time. To sustain this and provide some resiliency, we setup submission nodes (schedd's) at different locations worldwide (CMS Tier2's at UCSD and Nebraska and CERN). While those schedd's submit to the same condor pool, each job is only tracked by the schedd which submitted it, so CRAB client need to remember whom to ask for status and output. For this we re-used same mechanism that existed in CRAB already to deal with multiple servers:

a) A list of available submission hosts is maintained centrally by Analysis Operations and looked up via http by CRAB client at the time of the first submission in each task. A remote server is then chosen at random from that list.
b) Once a submission host has been picked like that, its name is stored in the local SqlLite database that CRAB client uses to keep track of the task progress and that name is then reused for any further CRAB command.

This is a very primitive form of load sharing, submission servers are chosen regardless of current load, hardware capability and size of the current task. Anyhow on average load is fairly well distributed given that we have servers of similar hardware strength. Main drawback of this approach is the fact that each user task is married to one particular server for life, so performing any disruptive or lengthy maintenance on a machine can only be done after a couple of weeks of draining.

5.4. Reliability

As expected gsissh is a very solid tool and we had almost no problem. One thing we never got around to solve is that at times $gsissh\ user@host\ "command"$ returns a non zero exit code even if the command actually succeeded. We worked around this by replacing $"command"$ with $"command;\ echo\ 'EXIT\ STATUS\ IS\ '\ \$?"$ and parsing stdout returned by gsissh. We also benefited from the simple trick of leaving users output files on the submission server even after successful copy to the local UI, so that the copy could be safely repeated if CRAB client fails later even for trivial reasons like full local disk. We assumed HTCondor to be fully reliable and took no particular care there, and have not found any reason to think otherwise.

Currently almost all problem reports from users about communication with the submission servers are due to actual lack of network connection.

6. Operational experience

Deployment in production of the new, gsissh based, submission method went very smoothly. It was presented to users as an additional option $scheduler = remoteglidein$ in the CRAB client submission file, everything else being unchanged. We let users free to choose if to use the new or the old methods, simply encouraging them to try the new one. Since CRAB with ssh was not providing any new functionality, only less problems and easier support, we took user's adoption as metric for success. Response was very good, as shown in figure 2. In a few months we found ourselves able to start decommissioning old CRAB servers and to stop active support for gLite scheduler.

From the operations point of view we had to ramp up our competence and expertise on HTCondor, but in the meanwhile we collected many benefits from the new submission method:

- Since submission is via gsissh, there is no need for full UI grid on client, no dependencies on swig wrapped C++ libraries, no problem of compatibility of environment with CMSSW and so on. CRAB for gissh works on SLC6 out of the box.

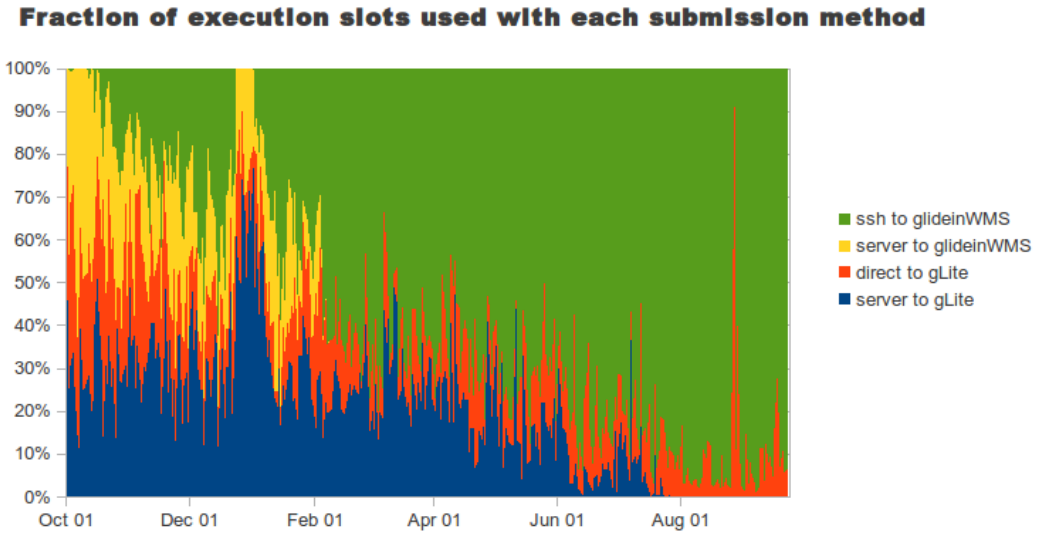


Figure 2. Steady shift to the use of the ssh-based CRAB over the period October 2012 to October 2013. It is clearly visible how ssh-based submission host at UCSD became unavailable during Christmas 2012, before we introduced multiple hosts in different regions

- No CMS software needed on submission server, only standard gsisshd and HTCondor. Simple installation, no problem ever with gsisshd. All problems with HTCondor routed to experts outside CMS core development team.
- The only work Analysis Operations needed to do on submission server was some simple crontab to cleanup old user files and basic system monitoring.

In practice operations load has been continuously decreasing over last year and it is now at an all time low, in spite of overall amount of analysis work having kept doubling every year, as can be seen in Figure 3.

Having moved users to the ssh portal gave us the possibility to work in the testing and commissioning of the prototypes of the new tool, called CRAB3, and to address long standing problems like user jobs running out of resources (time, memory, disk) on remote WN's, glitches in CMS software deployment at sites, problems with failing glexec. The better understanding that we have achieved of those problems and of the possible solutions will be directly incorporated in CRAB3 by the time it will be released for production.

We never had any major difficulty with the new system, occasional gsissh failures have been addressed and largely solved as indicated above. The fact that HTCondor automatically restarts any job that fails in case of a software or hardware glitch on the schedd nodes was also extremely useful. As of July 2013 we turned off the last CRAB server. In a way all its functionalities, but automatic resubmission of failed jobs, are reproduced in the current system.

7. Conclusions

Using (gsi)ssh as a portal has proven to be very effective in giving transparent, efficient access to remote submission servers. We could simplify enormously the software stack that we need to maintain both on the client and the server side and focus on more interesting problems like overall scalability, site's reliability and global policies. In a sentence: it is never too late for

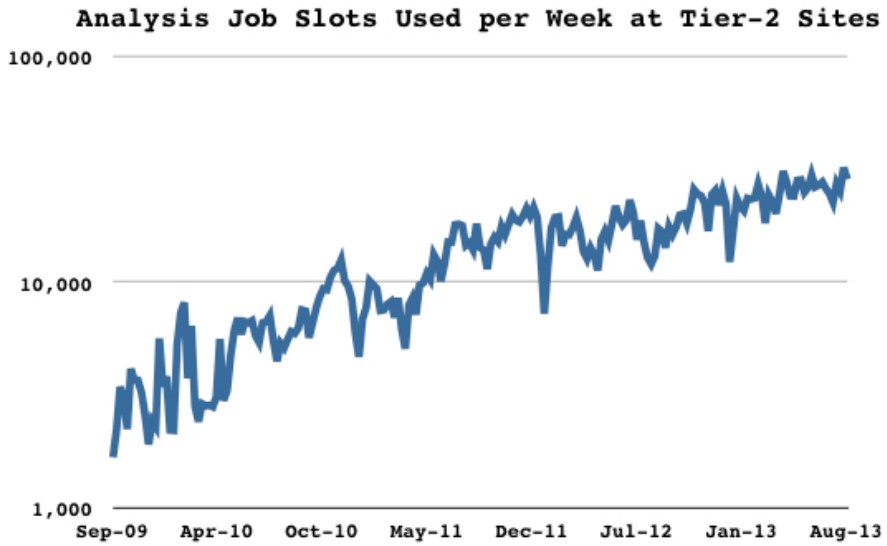


Figure 3. Amount of work for Analysis

throwing away software and simplifying the system.

Looking forward, we note two areas were improvement is still highly desirable, in spite of the success of the ssh portal approach. We hope that the next version of the CMS analysis submission tool will address those as well:

- Flexible automatic resubmission policies based on the failure kind (not all job failures benefit from the same resubmission policy).
- Current CRAB client sorts of "does too many things". It would be good to move some functionalities to the server side were it is possible to deploy changes faster and more frequently and thus better address the changing needs of our environment.

8. Acknowledgments

This work was partially sponsored by the US National Science Foundation under Grants No. PHY-1148698 and PHY-1120138.

References

[1] Chatrchyan S et al. 2008 *J. of Instr.* **3** S08004 doi:10.1088/1748-0221/3/08/S08004
[2] Pordes R et al. 2007 *J. Phys.: Conf. Ser.* **78** 012057 doi:10.1088/1742-6596/78/1/012057
[3] Kranzlmüller D, Marco de Lucas J and Öster P 2010 *Remote Instr. and Virt. Lab.* 61-66 doi:10.1007/978-1-4419-5597-5_6
[4] Spiga D et al. 2007 The CMS Remote Analysis Builder (CRAB) *Lect. Notes in Comp. Sci.* **4873** 580-586 doi:10.1007/978-3-540-77220-0_52
[5] Sfiligoi I, Bradley D C, Holzman B, Mhashilkar P, Padhi S and Würthwein F 2009 The pilot way to grid resources using glideinWMS *Comp. Sci. and Info. Eng., 2009 WRI World Cong. on* **2** 428-432 doi:10.1109/CSIE.2009.950
[6] Groep D, Koeroo O and Venekamp G 2008 gLExec: gluing grid computing to the Unix world *J. Phys.: Conf. Ser.* **119** 062032 doi:10.1088/1742-6596/119/6/062032
[7] Sfiligoi I and Dost J M TBD Using ssh and sshfs to virtualize Grid job submission with rcondor *Proc. CHEP 2013* (Amsterdam, NL)
[8] Butler R et al. 2000 A national-scale authentication infrastructure *Computer* **33** **12** 60-66 doi:10.1109/2.889094