CRAB: Introduction

CMS Tier3 Workshop
10 August 2011

Eric Vaandering
CMS / Fermilab

Outline

- Overview
- Client/Server
- Usage
- Support
- Intro to CRAB3
- Transition to CRAB3

CRAB:

CMS
Remote
Analysis
Builder

CRAB

- CRAB enables submission of CMSSW jobs to all CMS datasets within the data-location driven CMS computing structure
- The aim of CRAB is to hide as much of the complexity of the GRID as possible from the end user
- CRAB provides a user front-end to
  - Find data in and publish data to DBS
  - Split user jobs into manageable pieces
  - Transport user analysis code to the data location for execution (compiled on submitting node)
  - Execute user jobs, check status and retrieve output

Software Components

- CRAB interacts with many different pieces of CMS and Grid software on behalf of the user
  - DBS: What data exists? Publish the results.
  - Phedex: Where is the data?
  - BDII/SiteDB: What sites are available, who is the user?
  - Proxies/MyProxy: User authentication
  - Dashboard: Statistics and status of jobs
  - Grid middleware: Job submission, I/O

Grid UI in CRAB2

- CRAB (CMS Remote Analysis Builder) is the CMS user front end to the GRID
- The User Interface is the GRID specific software for authentication, job submission and all other GRID interactions
- CRAB works with UI's from EGEE and from OSG
- FNAL initialization: `source /uscmst1/prod/grid/gLite_SL5.(c)sh`
- CERN initialization: `source /afs/cern.ch/cms/LCG/LCG-2/UI/cms_ui_env.(c)sh`
- You won't have to use it directly, CRAB uses it for you

CRAB Server or Client

- Originally CRAB was only standalone. Ran from CLI and interacted with Grid jobs directly
- Most use now transitioned to CRAB Server
  - "crab" now a client interface to a server which submits and tracks jobs. Server is based on ProdAgent
  - Current recommended way to work if possible
- Standalone still exists. Mostly useful to interact with local schedulers (condor, PBS)

GRID certificates

- CMS uses GRID certificates and a dedicated Virtual Organization (VO) management to have better access control for specific tasks/groups
- Your GRID certificate is important, follow all rules, don’t let it expire!
- Can be a pain the first time, but then it's over

Underlying analysis model

User runs interactively on small samples in the local environment to develop the analysis code and test it

Once ready the user selects a large (whole) sample to submit the very same code to analyze many more events

The results are made available to the user to be analyzed interactively to produce final plots

Output handling

- CRAB has two ways of handling output:
  - The output sandbox
  - Copy files to a dedicated storage element (SE)
- Like the input sandbox, the output sandbox is limited in size (stand-alone only):
  - Input Sandbox: 10 MB
  - Output Sandbox: 50 MB
    - TRUNCATED if it exceeds 50 MB → corrupt files
- Rule of thumb:
  - If you would like to get event files back, please use a storage element

Data Publication

- CRAB allows you to publish the results of your work and share with others through DBS
- Requirements
  - SE that allows user copied data
    - Must be able to place into CMS's LFN structure
  - An analysis DBS server (several centrally maintained)
    - Need to know dbs_url
  - Must know at submission time that you will want to publish data
- Full instructions are at
  https://twiki.cern.ch/twiki/bin/view/CMS/SWGuideCrabForPublication

How to run CRAB

The basic CRAB workflow is organized into 4 steps:

- Job Creation
- Job Submission
- Check job status
- Output retrieval

Job Creation: **crab –create**

At this level CRAB interacts with the DBS system, organizes the jobs of the task according to the user's job splitting parameters, packs the users specific code(/lib /module /data), prepares the script to configure the remote environment, and (using BOSS) prepares the jdl file to communicate with the RB

It also creates the working directory which is organized in 4 subdirectories named:

/job	: CRAB specific stuff
/log	: CRAB log file location
/res	: default results destination
/share	: CRAB and scheduler specific

How to run CRAB (2)

Job submission: **crab –submit**

The submission uses the previously created CRAB project to submit the jobs.
Before the real submission, CRAB always checks for available resources preventing the submission of unmatched jobs. By default all created jobs are submitted to the server or directly to the Grid.

Job status: **crab –status**

This command checks the status of all jobs in the CRAB project.
For each job CRAB prints on the screen the job id, scheduler status, site hosting the jobs, cmssw exit code, & job exit code. The output gives also a summary with a list of job IDs sorted by status categories. By default the status of all jobs is checked.

Job output : **crab –getoutput**

This command retrieves the output of all jobs of a CRAB project which have status "Done". By default the retrieved output files are copied in the "res" sub-dir of the CRAB workingdir. Included are the standard output and error of the jobs (CMSSW stdout and stderr) and the output files specified in crab.cfg.

Even if your job fails, run **crab -getoutput**. Otherwise your output clogs up a server.

Installation

- A fully functional installation of CRAB requires the GLite middleware.
  - To support EEGE sites (European T1/T2)
  - Sometimes not easy, but it is not too bad
- OSG client only can work for certain stand-alone configs or for communication with the server
  - Not supported, not really recommended

Current Development

- CRAB must constantly keep up with changes to CMSSW and Grid middleware. Unless you have a very good reason, try to stay current with releases
- CRAB 2.x (current version) is now in maintenance mode. Only important bug fixes are being made
    - Freeze becomes deeper and deeper over time

How to get CRAB support

Best source for user support is the CRAB feedback hypernews:
https://hypernews.cern.ch/HyperNews/CMS/get/crabFeedback

All CRAB questions and suggestions can be posted to this forum,

Analysis operations tries to solve the problems and give solutions.

User suggestions can influence CRAB3 direction.

A troubleshooting guide is at https://twiki.cern.ch/twiki/bin/view/CMS/WorkBookGridJobDiagnosisTemplate

Questions not directly related to CRAB (GRID related problems, CMSSW specific problems, etc…) should be referred to other hypernews forums

Additional Documentation:
https://twiki.cern.ch/twiki/bin/viewauth/CMS/SWGuideCrab

CRAB3: Motivation

- What is CRAB3 and why are we doing it?
  - CRAB3 is a complete rewrite of CRAB. Only the name remains the same.
  - Same client/server model as CRABServer
  - Much thinner client, all work done on the server
- More stable development model
  - Based on WMAgent, the current CMS workflow software
    - Consolidated development, WMAgent designed for data
  - Modular structure allows us to add features we've said "later" to for years

Local Mode Plans

- One missing part of WMAgent is support for local schedulers
    - Scheduler plugins are easy, also need support for user switching with glexec because of server architechture
    - Realize this is likely something Tier2/3 sites will be very interested in
    - Good news is that this is a requirement for the FNAL LPC Tier3 as well
        - Most reliable & powerful analysis resource in CMS
        - Local Condor scheduling
- Bad news: Local mode will involve running a server

Other Tier3 Implications

- Input sandboxes are handled differently than in CRAB2.
    - Stored on a central server, job (HTTP) requests from WN
    - Will be advisable to set up caching squid proxy
- Remote stage out will be totally different
    - Most common failure mode with CRAB2
    - Output files stored locally in /store/temp
    - FTS initiated transfers will transfer files to destination
    - Doesn’t occupy WNs, # of simultaneous transfers limited

CRAB3: Current Status

- Under active development. Early “alpha” versions in the hands of integration
  - Underlying WMAgent is used for all Tier1 work, not for MC and RelVals yet
- CRAB3 should be useful for some real workflows by expert users later this year
  - Remaining blockers are ability to restrict jobs to specific lumi sections and publication of results
- FTS based asynchronous stageout is written but not integrated yet
- Very little effort yet on modifications for local submission mode

Transition to CRAB3

- 2012 will be a year of transition
- Confident CRAB3 will be more reliable than CRAB2
- Expect that as feature set of CRAB3 grows, more people will move over
    - Can take over some non-CRAB2 workflows as well. (e.g. FWLite workflows)
- Still expect that CRAB2 will be supported through most of 2012
    - CRAB2 on the FNAL LPC will be supported until a replacement is available

Takeaway

- Grid submission for CMS should become much more reliable going forward
- Little that Tier2/Tier3 sites have to do. These are, after all, Grid jobs
- Be prepared for FTS transfers like MC production
- Squid proxy to cache user sandboxes
- Running client from your Tier3 will be easy
- Setting up a server to directly access local resources will be more challenging