



# **User Experience in using CRAB and the LPC CAF**

---

Suvadeep Bose

TIFR/LPC

US CMS 2008 Run Plan Workshop

May 15, 2008



# Outline

- Learning to use CRAB
  - Learning to use LPC CAF (Condor)
  - Useful twiki pages
  - Possible mistakes with CRAB
- 
- Lessons learnt
  - Some comments

# How did I learn CRAB?

<https://twiki.cern.ch/twiki/bin/view/Main/CRAB>

You are here: [TWiki](#) > [Main Web](#) > [TWikiUsers](#) > [OliverGutsche](#) > CRAB

r3 - 08 Apr 2007 - 21:10:58 - OliverGutsche

## CRAB: CMS Remote Analysis Builder

### Contents:

- ↓ [CRAB: CMS Remote Analysis Builder](#)
- ↓ [Introduction](#)
- ↓ [Setup and usage instructions](#)
- ↓ [Help and support](#)
- ↓ [Information Resources](#)

### Introduction

**CRAB**, short for **CMS Remote Analysis Builder**, enables the user to process datasets and MC samples using the GRID. It hides the interaction with the GRID and provides the user with a simple and easy to use interface.

CRAB organizes the processing of data and MC samples in 4 steps:

1. Job creation
2. Job submission
3. Job status check
4. Job output retrieval

The following description introduces the setup and usage of CRAB following these [formatting conventions for shell commands and file contents](#).

Please follow the [tutorial navigation](#) to navigate from topic to topic using the navigation bars:

Previous:

Top:

Next:

or use one of the following topic links.

# Useful workshops/ tutorials

A tutorial by Oliver Gutsche (26 June, 2007)

<https://twiki.cern.ch/twiki/bin/view/Main/CMSSWatFNALTutorialJune2007>

US CMS First Physics Workshop (11 Oct – 13 Oct, 2007) by Eric Vaandering

<https://twiki.cern.ch/twiki/bin/view/Main/CRABatFNALTutorialOctober2007>

“CRAB Tutorial - analysis of starter kit, crab, condor, storage” in US CMS JTerm II (Jan, '08)

<https://twiki.cern.ch/twiki/bin/view/Main/EricVaanderingCRABTutorialJan2008>

# Using CRAB – step by step

## Setup and usage instructions

1. **Prerequisites / Installations**
  1. [Analysis code](#)
  2. [GRID credentials](#)
  3. [User interface](#)
    1. [Local user interface for sh family](#)
    2. [Local user interface for csh family](#)
  4. [CRAB installation](#)
2. **Setup / Initialization**
  1. [GRID authentication](#)
  2. [CMSSW project](#)
  3. [CRAB setup](#)
3. **CRAB configuration: crab.cfg**
  1. [Basics](#)
  2. [Input and Output handling](#)
    1. [Storage element interaction in general](#)
  3. [Dataset discovery and job configuration](#)
  4. [Example CRAB configuration file](#)
4. **CRAB basic usage**
  1. [Job creation](#)
  2. [Job submission](#)
  3. [Job status check](#)
  4. [Job output retrieval](#)
5. **Special CRAB usage options**
  1. [Kill jobs](#)
  2. [Job resubmission](#)
  3. [Job GRID id listing](#)
  4. [Check for compatible resources](#)
  5. [Post mortem analysis for aborted jobs](#)

Initial preparation

← This talk starts from here

## A typical script for CRAB

```
[CRAB]
jobtype          = cmssw
scheduler        = glitecoll ←

```

---

```
[CMSSW]
datasetpath      = /CSA07AllEvents/CMSSW_I_6_9-FastSim-I205437066/AODSIM
pset             = analysis.cfg
total_number_of_events = -1
events_per_job   = 500000
output_file      = rootfile.root

```

---

```
[USER]
return_data      = 1
use_central_bossDB = 0
use_boss_rt      = 0

```

```
[EDG]
rb               = CERN
proxy_server     = myproxy.cern.ch
virtual_organization = cms
retry_count      = 0
lcg_catalog_type = lfc
lfc_host         = lfc-cms-test.cern.ch
lfc_home        = /grid/cms

```

## If one wants to get a sure output rootfile better to use storage-path

```
[CRAB]
jobtype = cmssw
scheduler = condor_g
```

```
[CMSSW]
```

```
datasetpath = /CSA07AllEvents/CMSSW_I_6_9-FastSim-I205437066/AODSIM
pset= analysis.cfg
total_number_of_events=-1
events_per_job = 500000
output_file = rootfile.root, test.txt
```

```
[USER]
```

```
return_data = 0
copy_data = 1
storage_element = cmssrm.fnal.gov
storage_path = /srm/managerv1?SFN=/resilient/username/subdir
use_central_bossDB = 0
use_boss_rt = 1
```

```
[EDG]
```

```
lcg_version = 2
rb = CERN
proxy_server = myproxy.cern.ch
additional_jdl_parameters = AllowZippedISB = false;
se_white_list = cmssrm.fnal.gov
ce_white_list = cmsosgce2.fnal.gov
virtual_organization = cms
retry_count = 2
lcg_catalog_type = lfc
lfc_host = lfc-cms-test.cern.ch
lfc_home = /grid/cms
```

```
mkdir /pnfs/cms/WAX/resilient/username/subdir
chmod +775 /pnfs/cms/WAX/resilient/username/subdir
```

# Possible mistakes with CRAB

- The **dataset** is not available in the scheduler chosen.
- After declaring a storage path is one forgets to perform **chmod +775** on that directory prior to creating the crab jobs – then crab fails to write the output.
- The **output file name** in the crab-config file does not match with that in the analysis config file.
- If the **output file size** is too big and the user does not specify a storage path and tries to retrieve output rootfiles from the **output sandbox** (by performing **getoutput**) then due to big output file size the job gets **aborted**.
- If the farms chosen by the **scheduler** do not have that CMSSW version which the user is working on, then the job aborts.
- Sometimes runtime of the job exceeds the **grid-proxy time** resulting in job failure.

# Lessons learnt

- ❑ User **must** run 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. Submitting a crab job once and realising the mistake and hence killing jobs minimise the job's priority in the cluster.
- ❑ CRAB has (currently) two ways of handling output:
  - the output sandbox
  - Copy files to a dedicated storage element (SE)
- ❑ The input and output sandbox is limited in size:
  - 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 CMSSW ROOT files back, please use a storage element.  
Recommended for large outputs.  
(the standard output and error from CRAB still come through the sandbox)
- ❑ Even if your job fails, run **crab -getoutput**. Otherwise your output clogs up the server.

# Useful links

- Grid Analysis Job Diagnosis Template twiki page

<https://twiki.cern.ch/twiki/bin/view/CMS/WorkBookGridJobDiagnosisTemplate>

## CRAB error messages

| JOB EXIT CODE | ERROR TYPE  |
|---------------|---|
| 10016         | OSG: the working dir can not be created in the WN |
| 10017         | OSG: the working dir can not be removed from WN   |
| 10020         | cmsset_default.sh not found in VO_CMS_SW_DIR      |
| 10030         | middleware not identified                         |
| 10031         | CMS software dir not found in the WN              |
| 10032         | problem sourcing cmsset_default.sh                |
| 10034         | CMSSW version not found in the WN                 |
| 10099         | OSG: error with CE name                           |
| 50113         | too few argument for wrapper                      |
| 50110         | executable not foud                               |
| 60302         | output file not found                             |
| 60303         | output file already exists in the SE              |
| 60307         | general output copy to SE problem                 |
| 70000         | output file too big                               |

Can we have a simpler but more elaborate error table?

Help!

Best source for user support is the CRAB feedback hypernews:

<https://hypernews.cern.ch/HyperNews/CMS/get/crabFeedback>

## Using LPC CAF (Condor)

- Main source of information :  
<http://www.uscms.org/SoftwareComputing/UserComputing/BatchSystem.html>
- One needs to write scripts (shell-script/ python) that produces these condor executables. This is very much unlike CRAB where CRAB splits user jobs into manageable pieces and transport user analysis code to the data location for execution and execute user jobs.
- In Condor jobs user can not modify the config file even after submitting the condor jobs until he/she gets the output whereas in CRAB the whole things gets bundled up into a package and once crab job is submitted one may modify the config file.
- Issues with Condor that I faced:
  - *At times jobs stand Idle for long. Is there any way to put higher priority to it?*
  - *When jobs are NOT exited normally then one needs to look into the Output / Error . Mostly it is easily understood.*

# Comments - I

- ❑ CERN has options for submitting small jobs in the short queue. One can mention expected duration of the jobs at the time of submission. Presently in LPC CAF there is no such system. Though there exists a command called **LENGTH = "short"** for jobs that are expected to run for less than one hour. But this simply increases the job's priority *only if the nodes in the LPC CAF cluster are free*. But there is **no separate queue** for short job submission.  
Hence for running short jobs a user has to run interactively in one of the twelve cmslpc machines consuming the available CPU time of that particular node.
- ❑ For job submission in CONDOR one needs to **create different jobs manually** which CRAB does automatically. This is important for massive job running.
- ❑ LPC condor farm has *less number of working nodes* than the production farm. So a job, submitted by CRAB with **scheduler condor\_g** gets more nodes to run on than the same job submitted directly by **condor\_submit**. But the user competes with the production jobs. If mass production going on (eg. CSA exercises) then a common user gets less priority and jobs stand idle. One needs take these things into account before choosing the mode of submission.

## Comments - II

- ❑ If there is some dataset in remote site and the user needs to use that for analysis then **even to test his analysis file** he needs to submit crab job and wait for the output. Can a user request small samples through PhEDEx?
- ❑ At times once a crab job is aborted the error message is not easy to understand. Can we have a more comprehensible error message?
- ❑ At times in CRAB status I got status “waiting” which is not listed in the twiki page for crab which has three status messages - **Scheduled, Running and Done**.!! In such cases I killed the job and resubmitted. *But what should one ideally do?*
- ❑ While checking the status of a job the concept of **Done(success)** is not very clear. Many a times a job is shown to be Done(success) but performing a **getoutput** one sees no output !!!



## Conclusion

- In a nutshell it depends a lot on the user. If the user is careful about the job he/she is going to submit then most of the unwanted failures can be escaped.
- If the data is sitting here at Fermilab then using `condor_submit` is faster than using CRAB in most cases.
- The twiki pages for CRAB along with the tutorials are very useful and answers most of the questions that a user comes across.
- If you have problems running crab please ask people who have been using crab for long. Experience is the key thing here. It may be a good idea to start from a running config script if one wants to start running things quickly.