******start of Matthias's email!
Greetings All,
Here's a short rundown on how the tau analysis works (starting from BEANs).
Potential caveat: you might need dbs access for this... and you should run on earth and the NDCMS condor pool.
# Ntupling
1. Create a CMS release, e.g.,
   cmsrel CMSSW_5_3_11_patch3
   cd CMSSW_5_3_11_patch3/src
   cmsenv
2. install the BEAN packages as per instructions on [github](https://github.com/cms-ttH/BEAN/blob/master/README.md).
3. install grid-control:
      cd ~; svn co https://ekptrac.physik.uni-karlsruhe.de/public/grid-control/trunk/grid-control; cd -
4. install the TauAnalysis package:
      cd $CMSSW_BASE/src
      mkdir -p ttH
      git clone https://github.com/matz-e/TTHTauTau-Analysis.git ttH/TauAnalysis
      scram b -j32
5. fix paths (look for 'workdir' and 'se path' in ttH/TauAnalysis/test/grid_control.cfg and run the configuration:
      cd ttH/TauAnalysis/test
      vim grid_control.cfg
      voms-proxy-init -voms cms -valid 192:00 # this might not be needed?
      ~/grid-control/go.py -c grid_control.cfg
  - wait until grid-control finishes
6. Analysis
Perform the installation steps on [github](https://github.com/cms-ttH/ttH-TauRoast), then fix the paths and run the analysis:
   cd $LOCALRT/src/ttH/TauRoast/
   vim data/generic_ttl.yaml
   # look and fix items under paths to where ntuples are stored, and the 'root' under which output will be stored
   roaster -atfpvy data/generic_ttl.yaml
This will give you a yield table (how many events passed which cuts for each sample), and plots as specified in the configuration.
******end of Matthias's email!

```
#!/bin/tcsh

#May take X minutes to install and run
date

set CMSSW_BASE = (CMSSW_5_3_11_patch3)
#set CMSSW_BASE = (CMSSW_5_3_8_patch1)
set finalOutput = (/afs/crc.nd.edu/user/p/pivie/roast.out)
#set ntupleFolder = ("/hadoop/users/matze/ttH/v53/2012/ttl_{p}/*_ntuple.root") #Use this if the TauRoast gets updated again
set ntupleFolder = ("/hadoop/users/matze/ttH/v53/2012/ttl_{p}")

module load git

cmsrel $CMSSW_BASE
cd ~/$CMSSW_BASE
cd src
cmsenv

setenv CVSROOT :pserver:anonymous@cmssw.cvs.cern.ch:/local/reps/CMSSW
cvs login

cvs co -d PhysicsTools/NtupleUtils UserCode/Bicocca/PhysicsTools/NtupleUtils
cvs co -d SusyAnalysis -r V01-02-02 SusyAnalysis/EventSelector/interface/uncorrectionTypeMET.h
```

```
cvs co -r V07-00-01    TopQuarkAnalysis/Configuration
cvs co -r V06-07-11-01 TopQuarkAnalysis/TopTools
cvs co -r V06-05-06-07 DataFormats/PatCandidates
cvs co -r V00-02-14    DataFormats/StdDictionaries
cvs co -r V00-00-70    FWCore/GuiBrowsers
cvs co -r V08-09-52    PhysicsTools/PatAlgos
cvs co -r V03-09-23    PhysicsTools/PatUtils
cvs co -r V00-03-15    CommonTools/ParticleFlow
cvs co -r V00-00-12    CommonTools/RecoUtils
cvs co -r V04-06-09    JetMETCorrections/Type1MET
cvs co -r V01-08-00    RecoBTag/SecondaryVertex
cvs co -r V00-00-08    RecoMET/METAnalyzers
cvs co -r V00-00-07    RecoMET/METFilters
cvs co -r V15-01-11    RecoParticleFlow/PFProducer
cvs co -r V02-02-00    RecoVertex/AdaptiveVertexFinder
cvs co -d Muon/MuonAnalysisTools UserCode/sixie/Muon/MuonAnalysisTools
cvs co -r V00-00-13 -d EGamma/EGammaAnalysisTools UserCode/EGamma/EGammaAnalysisTools
cd EGamma/EGammaAnalysisTools/data
cat download.url | xargs wget
cd -
cd EGamma/EGammaAnalysisTools/interface/
wget -r http://nd.edu/~abrinke1/ElectronEffectiveArea.h -O ElectronEffectiveArea.h
cd -
cvs co -r V01-04-23 RecoTauTag/RecoTau
cvs co -r V01-04-10 RecoTauTag/Configuration
cvs co -r V00-04-00 CondFormats/EgammaObjects
cvs co -r V00-02-10 -d CMGTools/External UserCode/CMG/CMGTools/External

git clone https://github.com/cms-ttH/BEAN.git

scram b -j32

wget -O - http://pyyaml.org/download/pyyaml/PyYAML-3.10.tar.gz|tar xzf -
cd PyYAML-3.10/
python setup.py install --user
cd ..
rm -rf PyYAML-3.10

cd ~/$CMSSW_BASE
cd src

mkdir -p ttH
git clone https://github.com/cms-ttH/ttH-TauRoast.git ttH/TauRoast
scram b -j32
git clone https://github.com/cms-ttH/ttHMultileptonAnalysis.git

cd ttH
cd TauRoast
git checkout 351767b174f301eb1e13995a7023b3aad00dfdf5
mv data/generic_ttl.yaml data/generic_ttl.old.yaml
cat data/generic_ttl.old.yaml | sed "s,^\([ ]*root\: \).*,\1$finalOutput,g" | sed "s,^\([ ]*ntuples\: \).*,\1$ntupleFolder,g" >
data/generic_ttl.yaml
scram b clean; scram b

# look and fix items under paths to where ntuples are stored, and the 'root' under which output will be stored

echo "Installation complete:"
date

scripts/roaster -atfpvy data/generic_ttl.yaml
date
```

The script of the third solution

```
#!/bin/tcsh

#May take X minutes to install and run
echo "<1>the start of python building:"
date

set CMSSW_BASE = (CMSSW_5_3_11_patch3)
set finalOutput = (/afs/crc.nd.edu/user/p/pivie/roast.out)
set ntupleFolder = ("/hadoop/users/matze/ttH/v53/2012/ttl_{p}")

set PARROT_HELPER = '/afs/crc.nd.edu/user/h/hmeng/cctools/lib/$LIB/libparrot_helper.so'
setenv

cmsenv

cd ~/$CMSSW_BASE
cd src
wget -O - http://pyyaml.org/download/pyyaml/PyYAML-3.10.tar.gz|tar xzf -
cd PyYAML-3.10/
cmsenv
python setup.py install --user
cd -
rm -rf PyYAML-3.10

echo "<1>the end of python building:"
date

echo "<2>the start of analysis"
date
cd ttH
cd TauRoast
scripts/roaster -atfpvy data/generic_ttl.yaml
echo "<2>the end of analysis"
date




#declaration of environment variables
#clear data dependency
#clear software dependency
#software build and install command
#actual analysis program
```