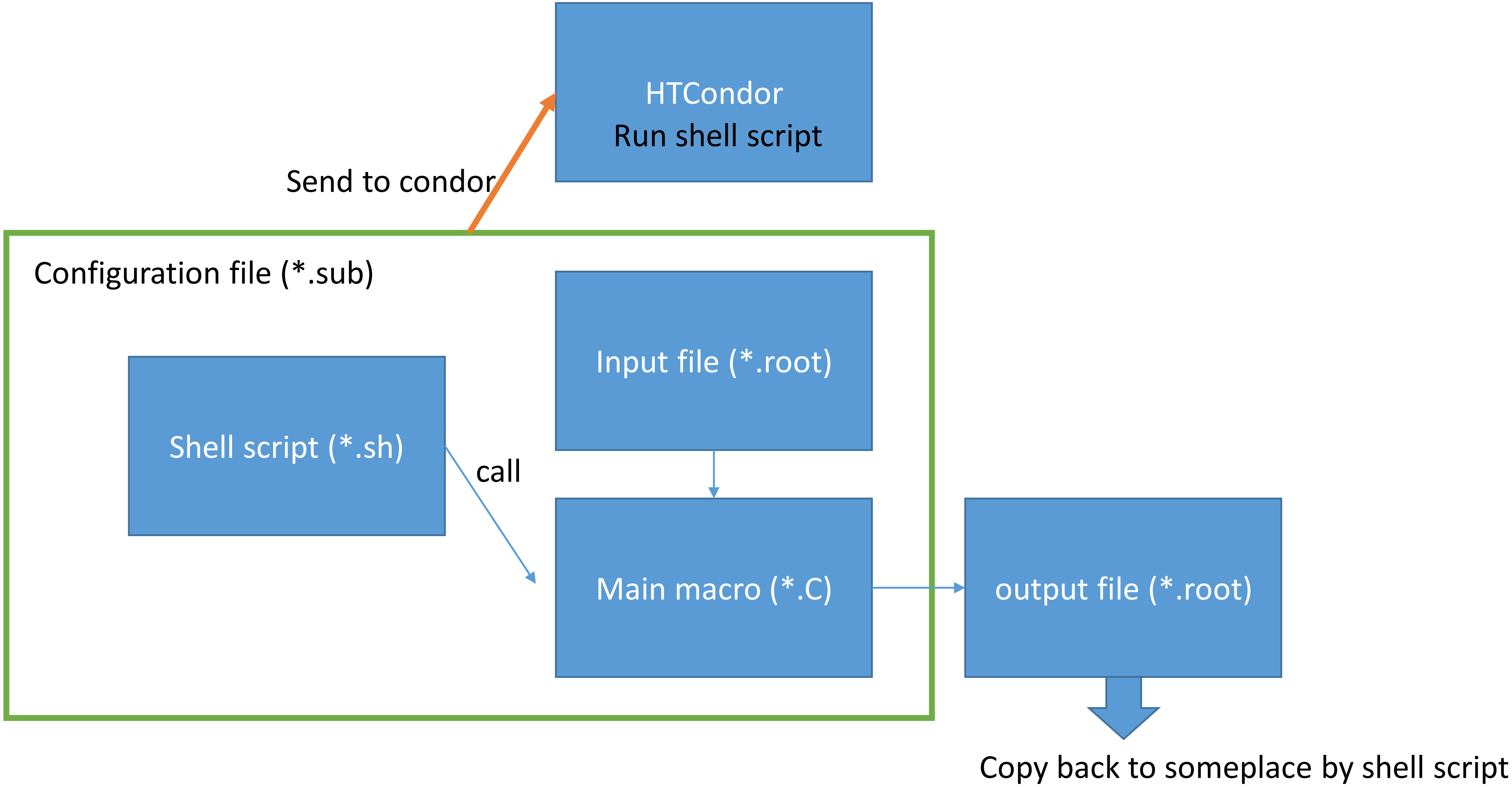
Condor job submit

Kong-Hsiang Chen

https://github.com/AllenChen1997/condor_submit

Structure


HTCondor
Run shell script
Send to condor
Configuration file (*.sub)
Input file (*.root)
Shell script (*.sh)
call
Main macro (*.C)
output file (*.root)
Copy back to someplace by shell script

Prepare the configuration file

example.sub

```
universe = vanilla
Proxy_filename = x509up
Proxy_path = /afs/cern.ch/user/k/kuchen/private/$(Proxy_filename)     1
request_memory = 4096          2
request_cpus = 4
+JobFlavour = "tomorrow"

executable = runAnalysis.sh               3
output = output/condor.$(Cluster).$(Process).out
error = error/condor.$(Cluster).$(Process).err
log = log/condor.$(Cluster).$(Process).log                      4
should_transfer_files = YES
transfer_input_files = runAnalysis.sh, readElement.C, dummy.txt, tmplist.txt
transfer_output_files = dummy.txt
on_exit_remove = (ExitBySignal == False) && (ExitCode == 0)
on_exit_hold = ( (ExitBySignal == True) || (ExitCode != 0) )
on_exit_hold_reason = strcat("Job held by ON_EXIT_HOLD due to ",ifThenElse((ExitBySignal == True), "exit by     5
signal",strcat("exit code ",ExitCode)), ".")
periodic_release =  (NumJobStarts < 5) && ((CurrentTime - EnteredCurrentStatus) > (60*60))

arguments = tmplist.txt  tmplist.root  $(Proxy_path)     3
queue
```

configuration file (1) setup proxy

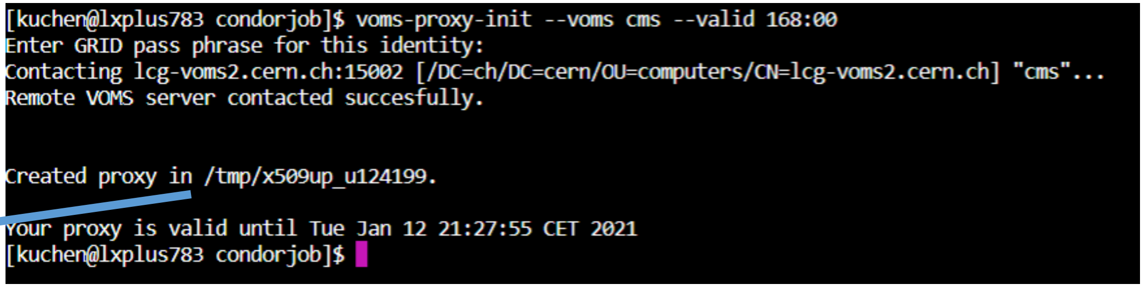
- If the job needs proxy to access files in other place…

e.g.
root://cmsxrootd.hep.wisc.edu///store/user/khurana/ExoPieElement/setup_2017_2016_v06/TTToSemiLeptonic_TuneCP5_PSweights_13TeV-powheg-pythia8/crab_TTToSemiLeptonic_TuneCP5_PSweights_13TeV-powheg-pythia8/200523_000250/0000/ExoPieElementTuples_1-1.root

Initial your proxy

```
voms-proxy-init --voms cms --valid 168:00
```

You may see



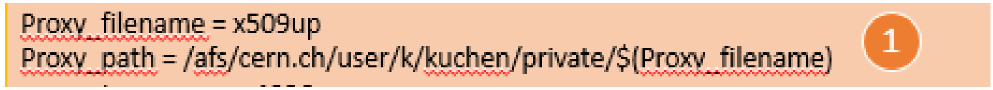
```
[kuchen@lxplus783 condorjob]$ voms-proxy-init --voms cms --valid 168:00
Enter GRID pass phrase for this identity:
Contacting lcg-voms2.cern.ch:15002 [/DC=ch/DC=cern/OU=computers/CN=lcg-voms2.cern.ch] "cms"...
Remote VOMS server contacted succesfully.


Created proxy in /tmp/x509up_u124199.

Your proxy is valid until Tue Jan 12 21:27:55 CET 2021
[kuchen@lxplus783 condorjob]$
```

Copy the tmp file to somewhere you want

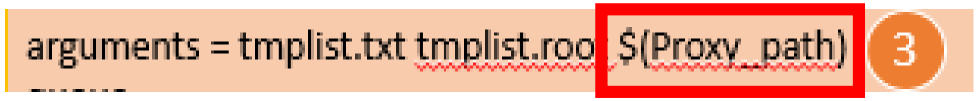
```
cp -v /tmp/x509up_u124199 /afs/cern.ch/user/k/kuchen/private/x509up
```

Setup in configuration file

this is not finished, please check (3-2)



```
Proxy_filename = x509up
Proxy_path = /afs/cern.ch/user/k/kuchen/private/$(Proxy_filename)
```



```
arguments = tmplist.txt tmplist.roo $(Proxy_path)
```

configuration file (2) job type

- In order to help scheduling or setup the requiring for the job, here are some options:
- request_memory = 4096
- request_cpus = 4
- +JobFlavour = "tomorrow"
  - This is the maximum job time. If the job is out of time, it will be terminated
  - Here is the table for Flavour

espresso	20min
microcentury	1h
longlunch	2h
workday	8h
tomorrow	1d
testmatch	3d
nextweek	1w

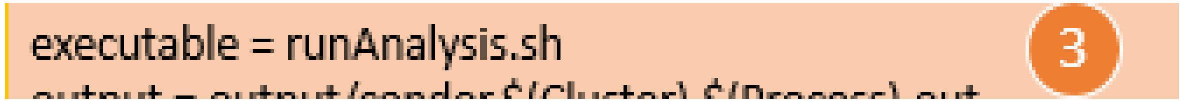
  - +MaxRuntime = Number of seconds also do the same thing

configuration file (3-1) executable code

- Write your own shell script to do the things
  - For example the bash script:

```
#!/bin/bash
root –b –q yourcode.C++\(\"var1\",\"var2\"\)
python yourcode.py
xrdcp <outputfile> <the_dir_you_want>
```

- Add these in configuration file



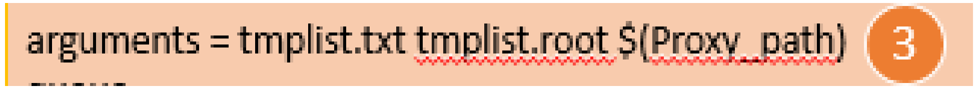
```
executable = runAnalysis.sh
output = output/condor.$(Cluster).$(Process).out
```

- Shell scripts can use some variable inputs when run as $n

  They are setup in arguments

  In this example $1 is tmplist.txt, $2 is tmplist.root, $3 is $(Proxy_path)

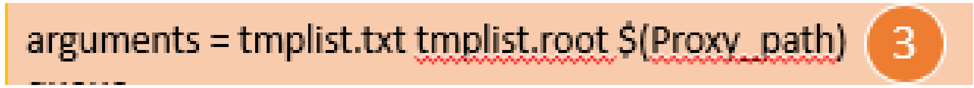
  For the usage example, you can check the next page



```
arguments = tmplist.txt tmplist.root $(Proxy_path)
```

configuration file (3-2) executable code

- If you want to setup the proxy in condor



```
arguments = tmplist.txt tmplist.root $(Proxy_path)
```

3

```
export X509_USER_PROXY=$3
voms-proxy-info -all
voms-proxy-info -all -file $3
```

In this example:

$1 is tmplist.txt

$2 is tmplist.root

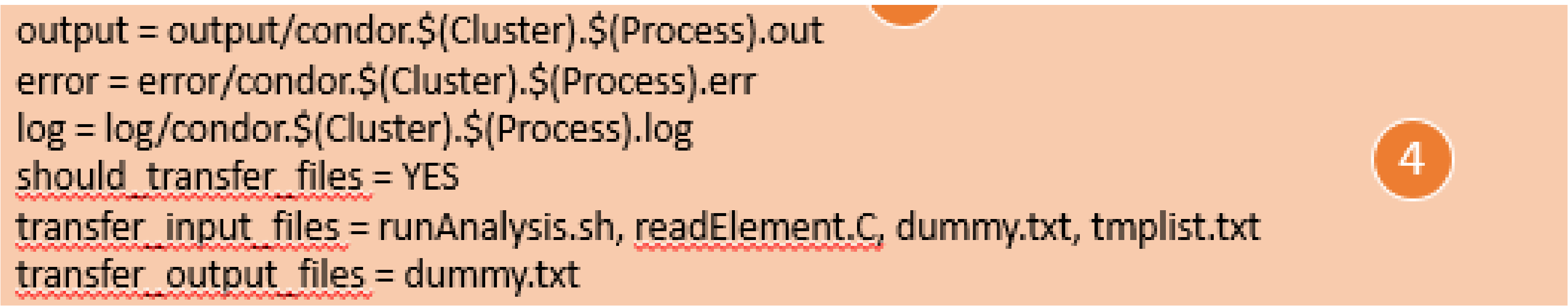
$3 is $(Proxy_path)

Combine previous page, you can run your root macro like this:

```
root –b –q yourcode.C++\(\”$1\”,\”$2\”\)
```

configuration file (4) input / output files

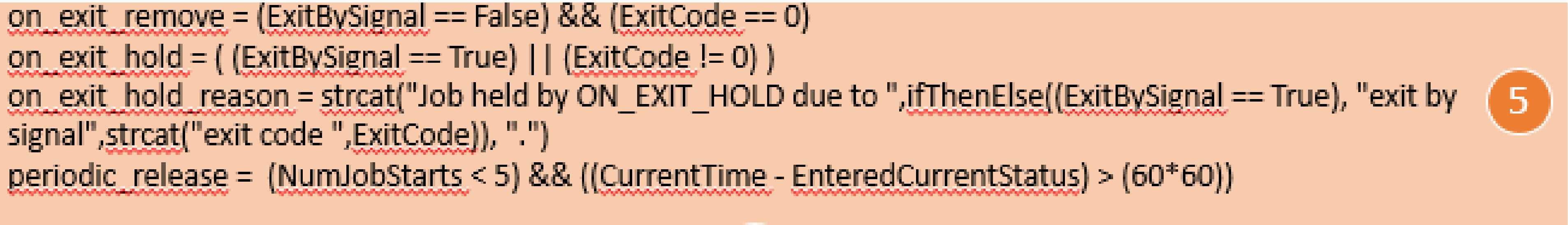
- There are some log files can help you to know the status about the job
- output will collect the contents when running your shell scripts
- error will collect the errors when running your shell scripts
- log will collect the job status along the time
- transfer_input/output_files declare the files to transfer to/from condor (make sure the file exist)



```
output = output/condor.$(Cluster).$(Process).out
error = error/condor.$(Cluster).$(Process).err
log = log/condor.$(Cluster).$(Process).log
should_transfer_files = YES
transfer_input_files = runAnalysis.sh, readElement.C, dummy.txt, tmplist.txt
transfer_output_files = dummy.txt
```

configuration file (5) job state control

- on_exit_remove: if the condition is true, it leave the job queue normally. If false, placed back into the Idle state
- on_exit_hold: if the condition is true, it place job into hold state. If false, nothing happened
- on_exit_hold_reason: show the description when on_exit_hold is true
- periodic_release: it the condition is true, the job will be released



```
on_exit_remove = (ExitBySignal == False) && (ExitCode == 0)
on_exit_hold = ( (ExitBySignal == True) || (ExitCode != 0) )
on_exit_hold_reason = strcat("Job held by ON_EXIT_HOLD due to ",ifThenElse((ExitBySignal == True), "exit by
signal",strcat("exit code ",ExitCode)), ".")
periodic_release =  (NumJobStarts < 5) && ((CurrentTime - EnteredCurrentStatus) > (60*60))
```

Useful commands

- condor_submit yourconfig.sub   submit the job
- condor_q   check the job state
- condor_q –analyze <jobId>   show more details

e.g. condor_q –analyze 7082186.0

- condor_tail <jobId>   if the state in run, show the contents in output now
- condor_rm <jobId>   remove the job
- condor_rm <your user name> remove all the job you submit

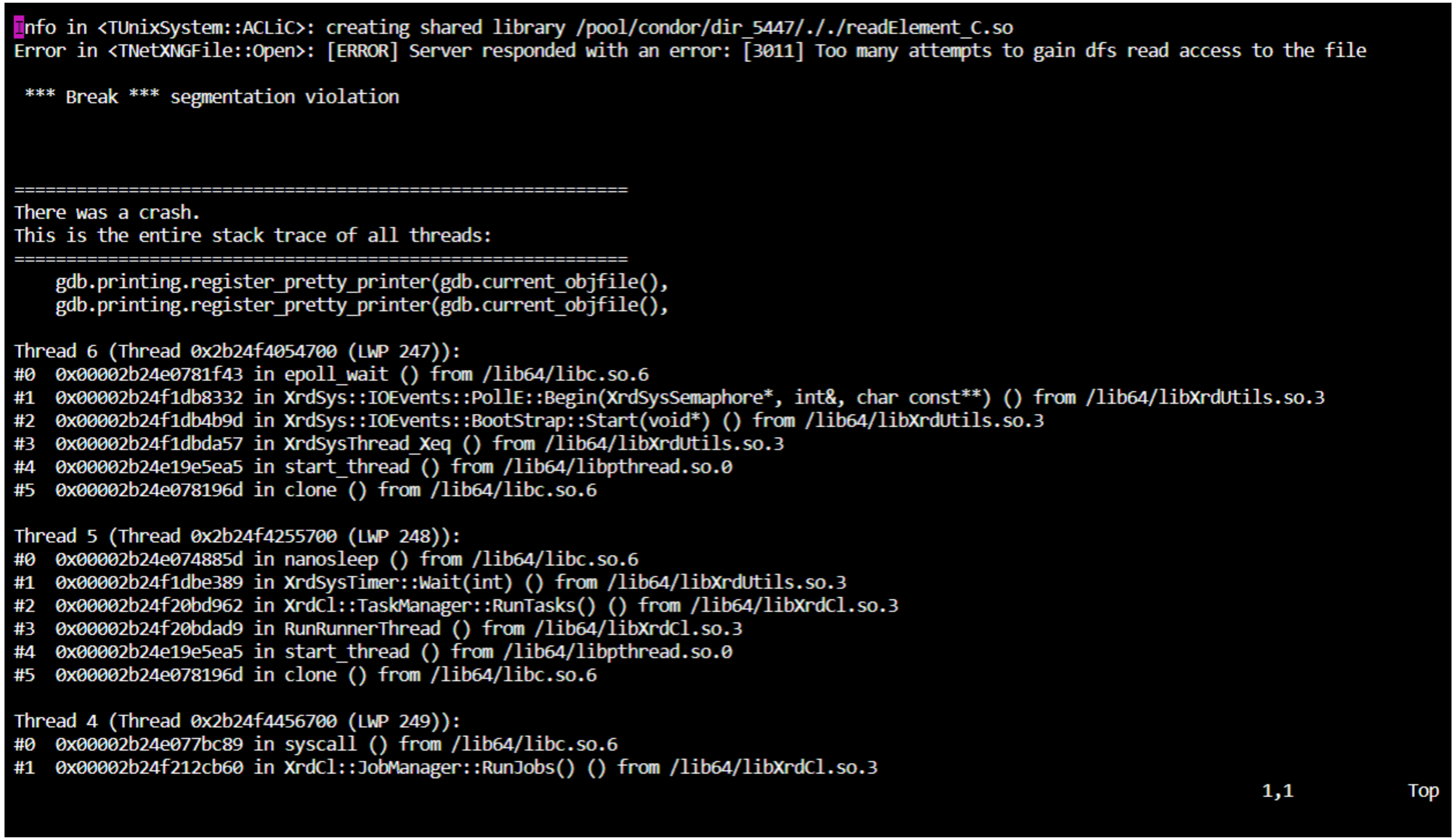
The error you may meet

Looks like very frequently open files in that sever cause this kind of error.

Sometimes resubmit will work fine, but still don’t know the exactly way to solve this problem

->solution:

separate into jobs per input root file by queue(see next page)



```
Info in <TUnixSystem::ACLiC>: creating shared library /pool/condor/dir_5447/././readElement_C.so
Error in <TNetXNGFile::Open>: [ERROR] Server responded with an error: [3011] Too many attempts to gain dfs read access to the file

 *** Break *** segmentation violation



===========================================================
There was a crash.
This is the entire stack trace of all threads:
===========================================================
    gdb.printing.register_pretty_printer(gdb.current_objfile(),
    gdb.printing.register_pretty_printer(gdb.current_objfile(),

Thread 6 (Thread 0x2b24f4054700 (LWP 247)):
#0  0x00002b24e0781f43 in epoll_wait () from /lib64/libc.so.6
#1  0x00002b24f1db8332 in XrdSys::IOEvents::PollE::Begin(XrdSysSemaphore*, int&, char const**) () from /lib64/libXrdUtils.so.3
#2  0x00002b24f1db4b9d in XrdSys::IOEvents::BootStrap::Start(void*) () from /lib64/libXrdUtils.so.3
#3  0x00002b24f1dbda57 in XrdSysThread_Xeq () from /lib64/libXrdUtils.so.3
#4  0x00002b24e19e5ea5 in start_thread () from /lib64/libpthread.so.0
#5  0x00002b24e078196d in clone () from /lib64/libc.so.6

Thread 5 (Thread 0x2b24f4255700 (LWP 248)):
#0  0x00002b24e074885d in nanosleep () from /lib64/libc.so.6
#1  0x00002b24f1dbe389 in XrdSysTimer::Wait(int) () from /lib64/libXrdUtils.so.3
#2  0x00002b24f20bd962 in XrdCl::TaskManager::RunTasks() () from /lib64/libXrdCl.so.3
#3  0x00002b24f20bdad9 in RunRunnerThread () from /lib64/libXrdCl.so.3
#4  0x00002b24e19e5ea5 in start_thread () from /lib64/libpthread.so.0
#5  0x00002b24e078196d in clone () from /lib64/libc.so.6

Thread 4 (Thread 0x2b24f4456700 (LWP 249)):
#0  0x00002b24e077bc89 in syscall () from /lib64/libc.so.6
#1  0x00002b24f212cb60 in XrdCl::JobManager::RunJobs() () from /lib64/libXrdCl.so.3
                                                                                              1,1           Top
```

HTCondor queue

You can see this configuration file https://github.com/AllenChen1997/condor_submit/blob/main/submit_multi_quever.sub

The main key point is in the last line

queue inputfile from $(listFile)

It will submit the condor jobs with the same cluster ID but different job ID(loop by different inputfile “value” here)

Other links

- Basic concept for Htcondor: https://indico.cern.ch/event/611296/contributions/2604376/attachments/1471164/2276521/TannenbaumT_UserTutorial.pdf

- Some introduction about content of configuration file: https://batchdocs.web.cern.ch/local/submit.html https://research.cs.wisc.edu/htcondor/manual/v8.7/Condorsubmit.html