Step 1: Install Condor

Download the latest version of HTCondor(native package) from the following URL.

http://research.cs.wisc.edu/htcondor/downloads/

Install Condor using the following commands:

```
$ sudo dpkg -i condor_8.6.11-440910-ubuntu14_amd64.deb

$ sudo apt-get update

$ sudo apt-get install -f

$ sudo apt-get install chkconfig

$ sudo chkconfig condor on

$ sudo service condor start
```

Step 2: Install Pegasus

Pegasus needs Java (1.6 or higher) and Python (2.4 or higher). Download the Pegasus(Ubuntu) frome the following URL.

https://pegasus.isi.edu/downloads/

To add the Pegasus repository for install and automatic updates, first download and install the repostiory key:

```
$ wget -O - http://download.pegasus.isi.edu/pegasus/gpg.txt | sudo apt-key add -
```

Create repository file, update and install Pegasus (currently available releases are trusty and xenial):

```
$ echo 'deb [arch=amd64] http://download.pegasus.isi.edu/pegasus/ubuntu xenial main' | sudo tee /etc/apt/sources.list.d/pegasus.list
$ sudo apt-get update

$ sudo apt-get install pegasus
```

Step 3: Install OpenCPU (at chapter 2 of OpenCPU PDF manual)

The OpenCPU PDF manual is available at:

https://opencpu.github.io/server-manual/opencpu-server.pdf

(1) Before installing the OpenCPU, it is necessary to ensure the system is the latest. Use the following Linux commands to update the system:

```
sudo apt-get update

sudo apt-get upgrade
```

(2) The first step in installing OpenCPU is to add necessary libraries for the system.

```
sudo add-apt-repository ppa:opencpu/opencpu-2.0 -y

sudo apt-get update
```

(3) Then install the OpenCPU server

```
sudo apt-get install opencpu-server
```

(4) Finally, to get install OpenCPU together with a lot of other potentially useful things:

```
sudo apt-get install opencpu-full
```

This package will install opencpu-server, texlive, rstudio-server, git and some more. Note that this takes is at least several GB of disk space on a fresh system.

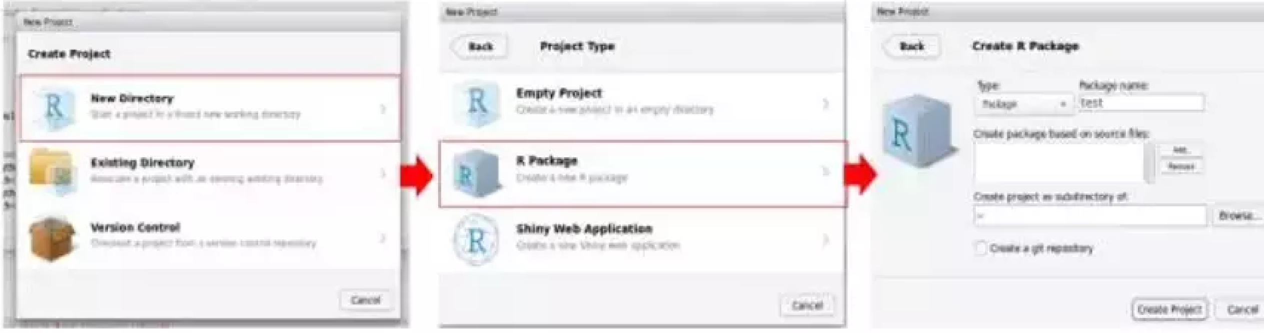
(5) After installing OpenCPU server, the last step is to ensure that OpenCPU is activated on Apache2

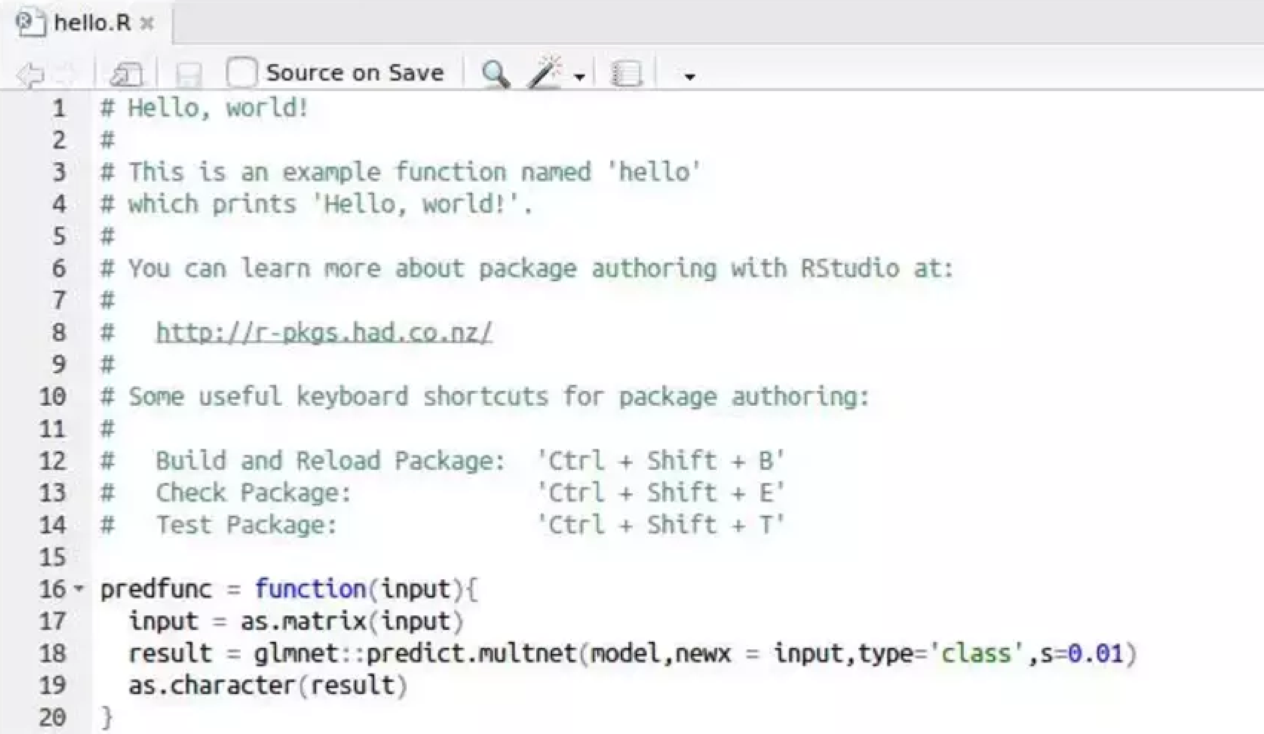
```
sudo a2ensite opencpu

sudo apachectl restart
```

After installation is done, we should be able to open a browser and point it to the /ocpu path at server address e.g: http(s)://your.server.com/ocpu. If the welcome page shows up, the installation has succeeded.

Step 4: Deploy R function in OpenCPU

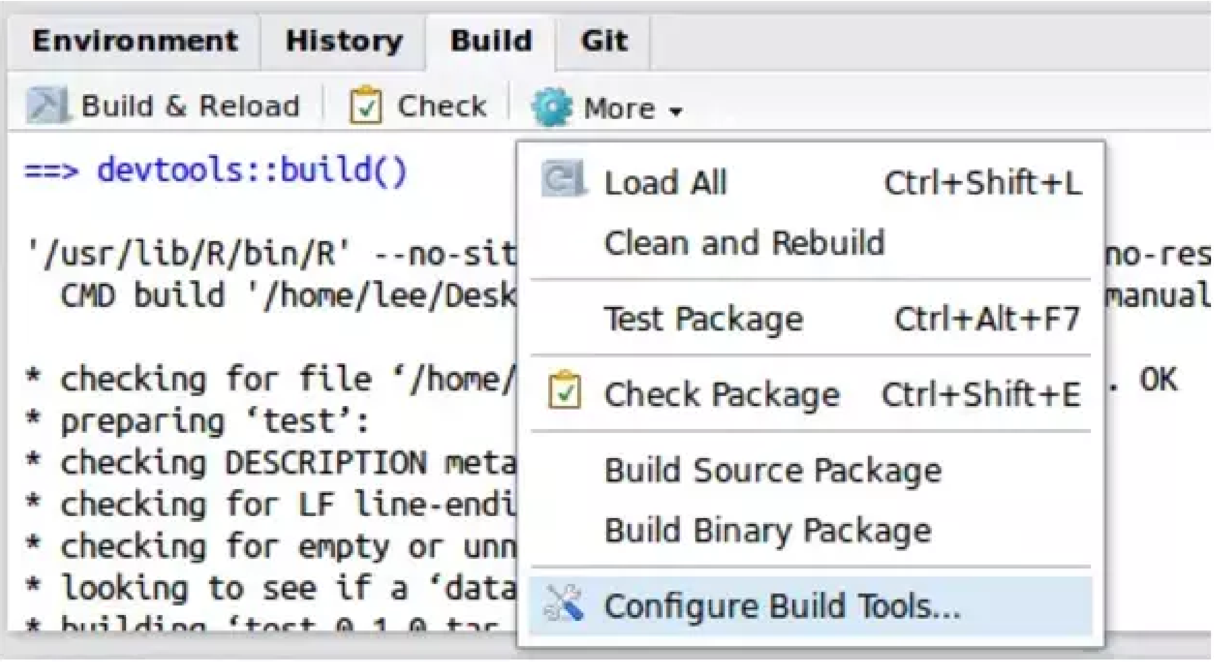
Create new R packages in Rstudio or load R packages(wranglingA,wranglingB,wranglingC) by double clicking Rproj file.


New Project
Create Project
New Directory
Start a project in a brand new working directory
Existing Directory
Associate a project with an existing working directory
Version Control
Checkout a project from a version control repository
Cancel
New Project
Back
Project Type
Empty Project
Create a new project in an empty directory
R Package
Create a new R package
Shiny Web Application
Create a new Shiny web application
Cancel
New Project
Back
Create R Package
Type:
Package
Package name:
test
Create package based on source files:
Add...
Remove
Create project as subdirectory of:
~
Browse...
Create a git repository
Create Project
Cancel



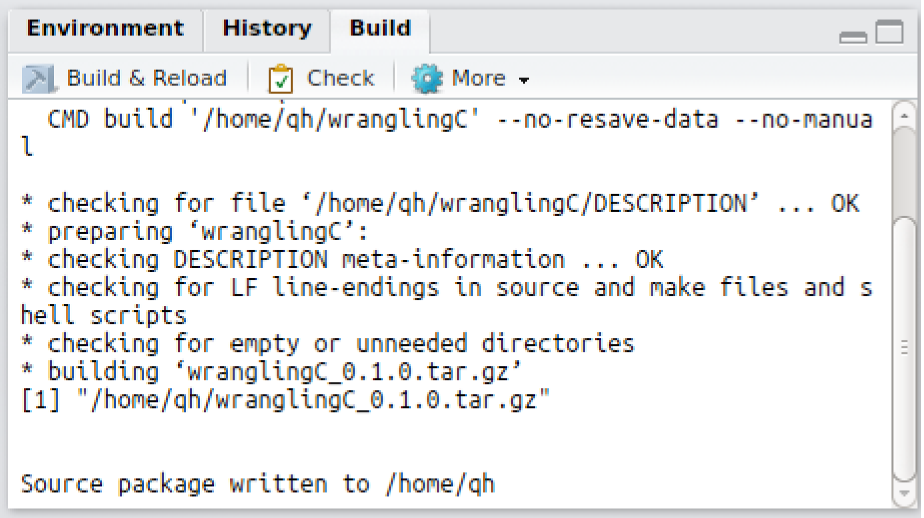
```r
# Hello, world!
#
# This is an example function named 'hello'
# which prints 'Hello, world!'.
#
# You can learn more about package authoring with RStudio at:
#
#   http://r-pkgs.had.co.nz/
#
# Some useful keyboard shortcuts for package authoring:
#
#   Build and Reload Package:  'Ctrl + Shift + B'
#   Check Package:             'Ctrl + Shift + E'
#   Test Package:              'Ctrl + Shift + T'

predfunc = function(input){
  input = as.matrix(input)
  result = glmnet::predict.multnet(model,newx = input,type='class',s=0.01)
  as.character(result)
}
```

Then click Build → Build source package



```
==> devtools::build()

'/usr/lib/R/bin/R' --no-sit                              no-res
  CMD build '/home/lee/Desk                              manual

* checking for file '/home/                            . OK
* preparing 'test':
* checking DESCRIPTION meta
* checking for LF line-endi
* checking for empty or unn
* looking to see if a 'data
* building 'test_0.1.0.tar
```

After that, we can find a tar.gz file has generated.



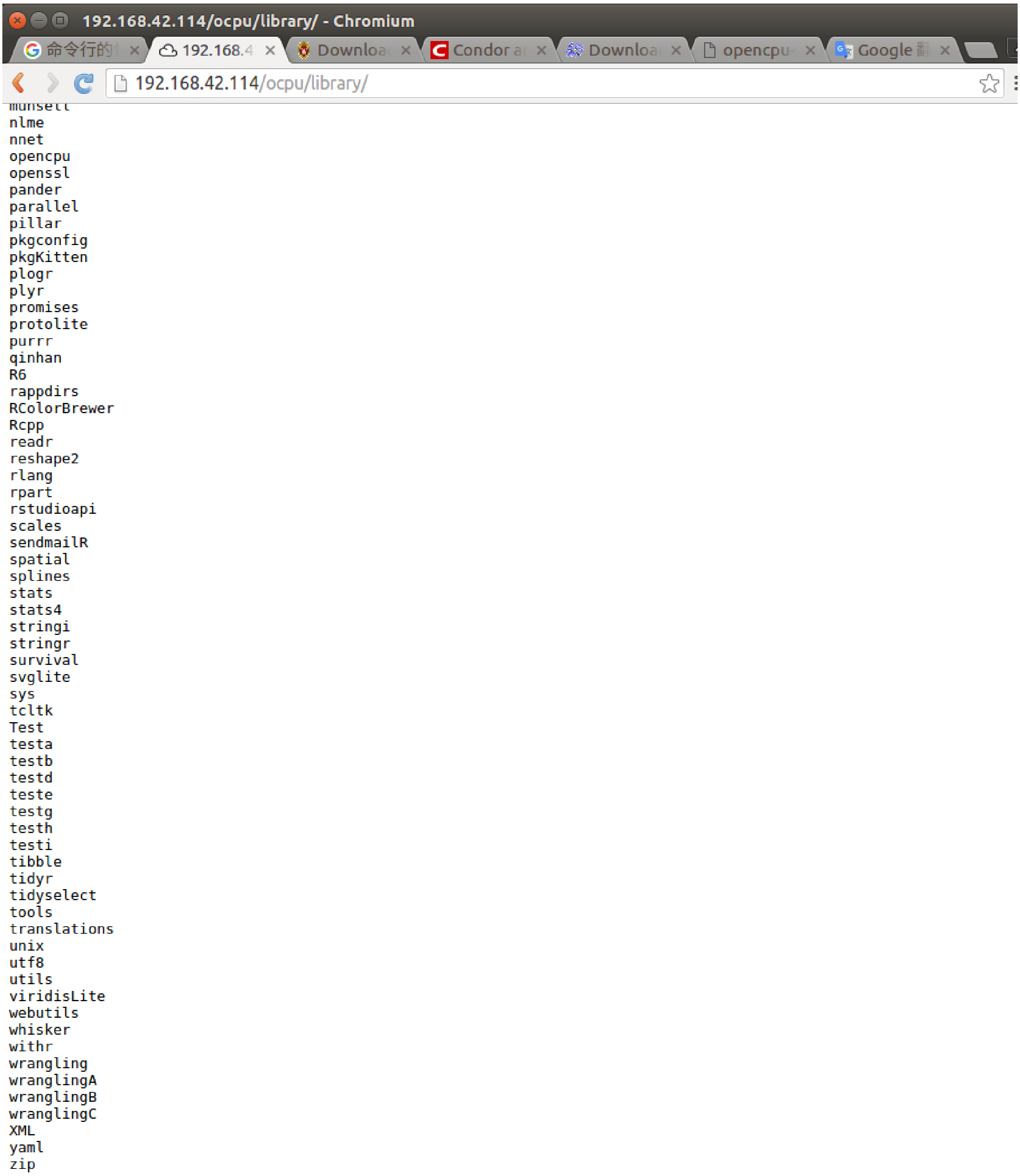
```
  CMD build '/home/qh/wranglingC' --no-resave-data --no-manua
l

* checking for file '/home/qh/wranglingC/DESCRIPTION' ... OK
* preparing 'wranglingC':
* checking DESCRIPTION meta-information ... OK
* checking for LF line-endings in source and make files and s
hell scripts
* checking for empty or unneeded directories
* building 'wranglingC_0.1.0.tar.gz'
[1] "/home/qh/wranglingC_0.1.0.tar.gz"


Source package written to /home/qh
```

Then, deploy this packages(/home/qh/wranglingC_0.1.0.tar.gz) by installing this package under the global library by the following command:

```
sudo R CMD INSTALL your/package/path --library=/usr/local/lib/R/site-library
```

Then, go to the following URL to find the package:

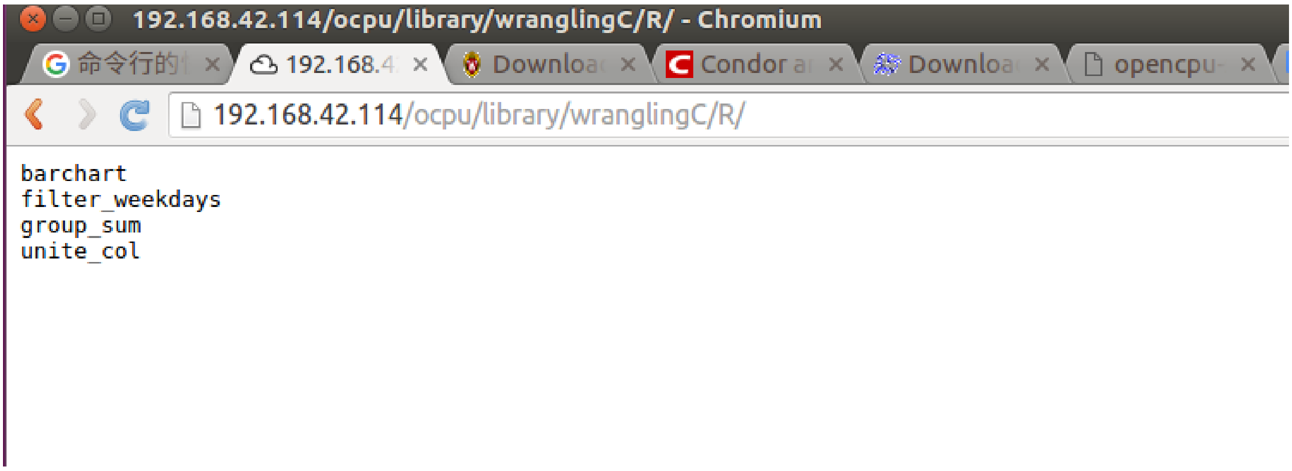
```
https://your.server.com/ocpu/library/
```


192.168.42.114/ocpu/library/ - Chromium
命令行的
192.168.4
Downloa
Condor ar
Downloa
opencpu
Google
192.168.42.114/ocpu/library/

```
munsell
nlme
nnet
opencpu
openssl
pander
parallel
pillar
pkgconfig
pkgKitten
plogr
plyr
promises
protolite
purrr
qinhan
R6
rappdirs
RColorBrewer
Rcpp
readr
reshape2
rlang
rpart
rstudioapi
scales
sendmailR
spatial
splines
stats
stats4
stringi
stringr
survival
svglite
sys
tcltk
Test
testa
testb
testd
teste
testg
testh
testi
tibble
tidyr
tidyselect
tools
translations
unix
utf8
utils
viridisLite
webutils
whisker
withr
wrangling
wranglingA
wranglingB
wranglingC
XML
yaml
zip
```

We can find the wranglingC package in the bottom.

In the following URL, we can find the R operators of wranglingC packages:


192.168.42.114/ocpu/library/wranglingC/R/ - Chromium
命令行的
192.168.4
Downloa
Condor a
Downloa
opencpu-
192.168.42.114/ocpu/library/wranglingC/R/

```
barchart
filter_weekdays
group_sum
unite_col
```

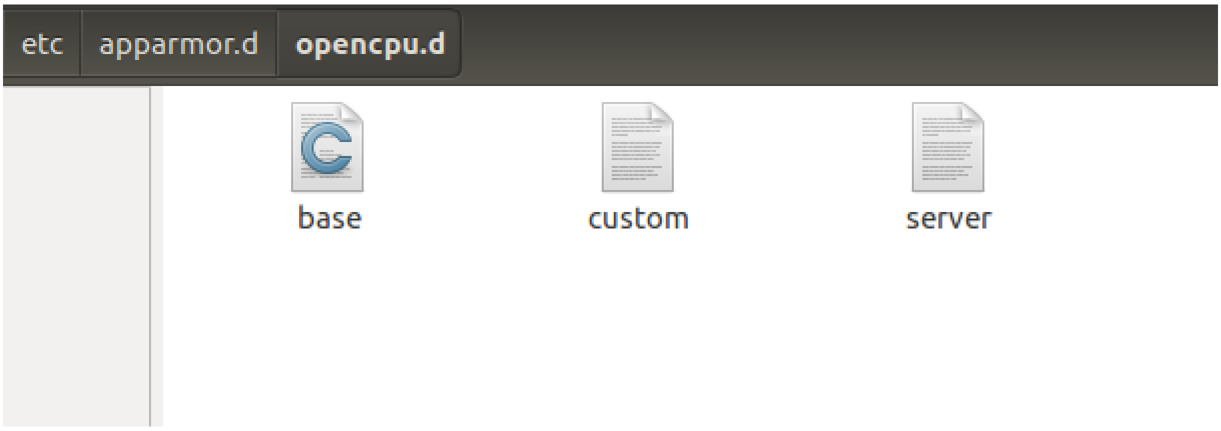
We use curl command to call these R operators like this:

```
curl https://your.server.com/ocpu/library/datawrangling/R/filter -d  "input=a.csv&output=b.csv"
```

"-d" are the parameters required by the data wrangling operators.

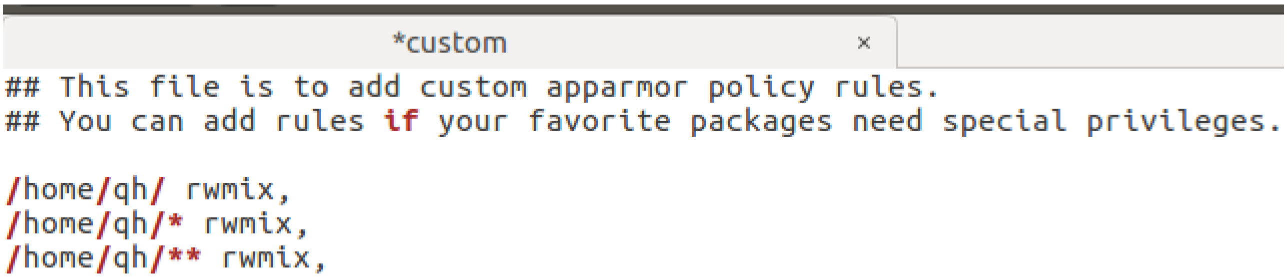
We should deploy all of wranglingA, wranglingB and wranglingC.

After that, we should configure the security file of OpenCPU (at chapter 3.5 of OpenCPU PDF manual). The profiles are stored in /etc/apparmor.d/opencpu.d/. We should change the custom file in this path.



Create a folder called opencputmp, this folder is used to store the temporary data of workflow that need to be processed by wrangling operators.

We should give some permission to the OpenCPU to write and read file in this folder. Add the path of opencputmp folder and the permission in the custom file. Like the following code:



```
## This file is to add custom apparmor policy rules.
## You can add rules if your favorite packages need special privileges.

/home/qh/ rwmix,
/home/qh/* rwmix,
/home/qh/** rwmix,
```

r – read file or directory.

w – write to file or directory.

m – load file in memory.

px – discrete profile execute of executable file.

cs – transition to subprofile for executing a file.

ix – inherit current profile for executing a file.

ux – unconfined execution of executable file (dangerous).

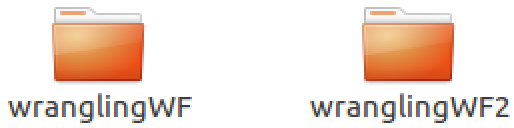
After modifying a security profile, restart AppArmor:

```
sudo service apparmor restart
```

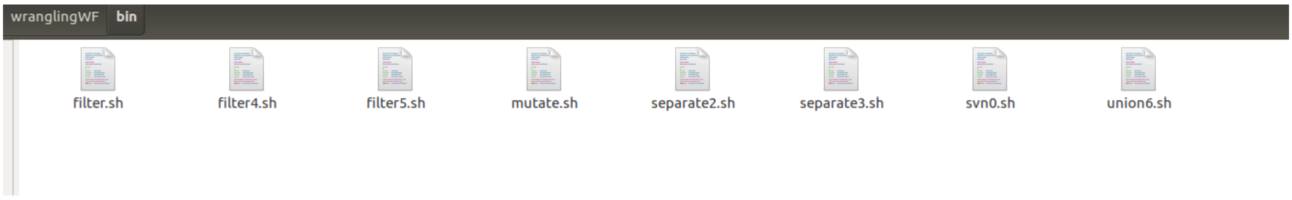
Finally, use chmod command to give read and write permission to the opencputmp fold.

Step 5: Pegasus configuration(at https://pegasus.isi.edu/documentation/tutorial.php)

wranglingWF and wrangling WF2 are two examples of workflow.



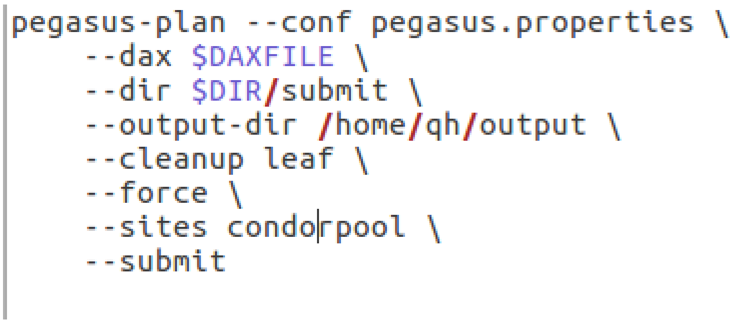
In these files, wrangling.dax is abstract workflow description,it can be generated by daxgen.py file or writing dirctly. Submit is executable workflow. Bin is workflow jobs shell scripts, we need to change the curl address in these scripts to the address of your services and the path of opencputmp file.



plan_dax.sh is planning file, we can modify output path, executable workflow path in there.

plan_dat.sh is used to plan and submit the workflow. And we can change the output direction in this file. Change the output direction to the path you want.

(at https://pegasus.isi.edu/documentation/pegasus-plan.php)

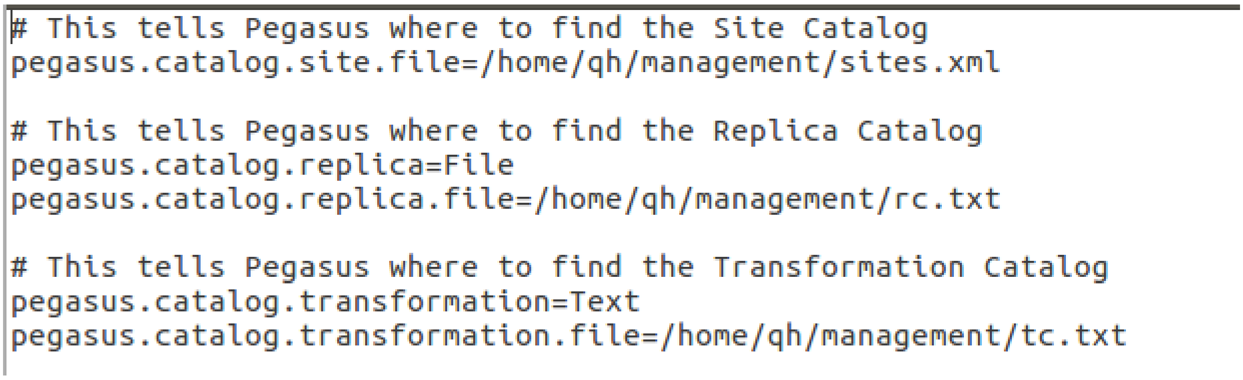


```
pegasus-plan --conf pegasus.properties \
     --dax $DAXFILE \
     --dir $DIR/submit \
     --output-dir /home/qh/output \
     --cleanup leaf \
     --force \
     --sites condorpool \
     --submit
```

In pegasus.properties, we can change the information catalogs path. We have three catalogs, they are in the management file. Before running the workflow, we should change the information catalogs path to your management file’s path.

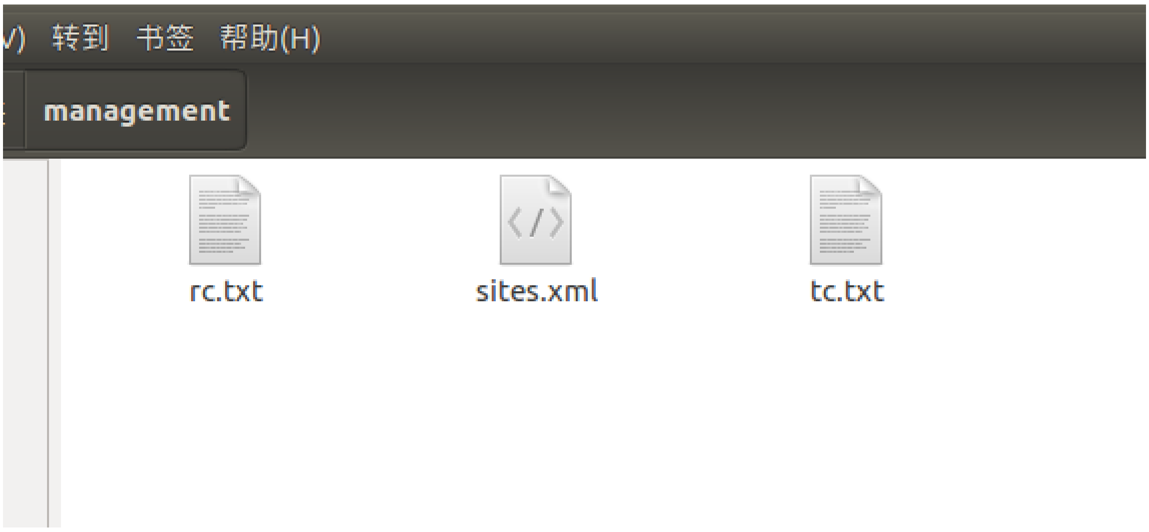
(at https://pegasus.isi.edu/documentation/tutorial_configuration.php)

This is where we need to change the path.



```
# This tells Pegasus where to find the Site Catalog
pegasus.catalog.site.file=/home/qh/management/sites.xml

# This tells Pegasus where to find the Replica Catalog
pegasus.catalog.replica=File
pegasus.catalog.replica.file=/home/qh/management/rc.txt

# This tells Pegasus where to find the Transformation Catalog
pegasus.catalog.transformation=Text
pegasus.catalog.transformation.file=/home/qh/management/tc.txt
```

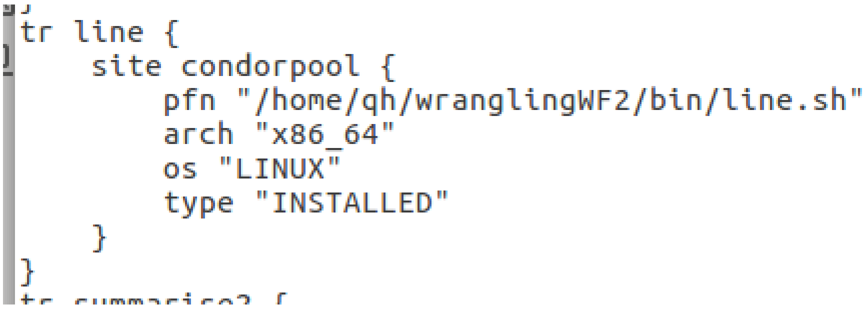
And this is the management file



rc.txt tells pegasus where are the input data sets, but we use svn command to download the data sets from github, so , we do not need to write this.

Sites.xml tells pegasus where is scratch path and the style of condorpool,we do not need to change the condorpool style, we just need to change the scratch path to the place you want.

tc.txt tells pegasus each jobs' shell script path.



```
tr line {
    site condorpool {
        pfn "/home/qh/wranglingWF2/bin/line.sh"
        arch "x86_64"
        os "LINUX"
        type "INSTALLED"
    }
}
```

For example, this means the job ‘line’(in the abstract workflow description file) need to call the shell script ‘line.sh’, and the path of this script is “/home/qh/wranglingWF2/bin/line.sh”. We need to change all of the job scripts paths to your computers scripts paths.

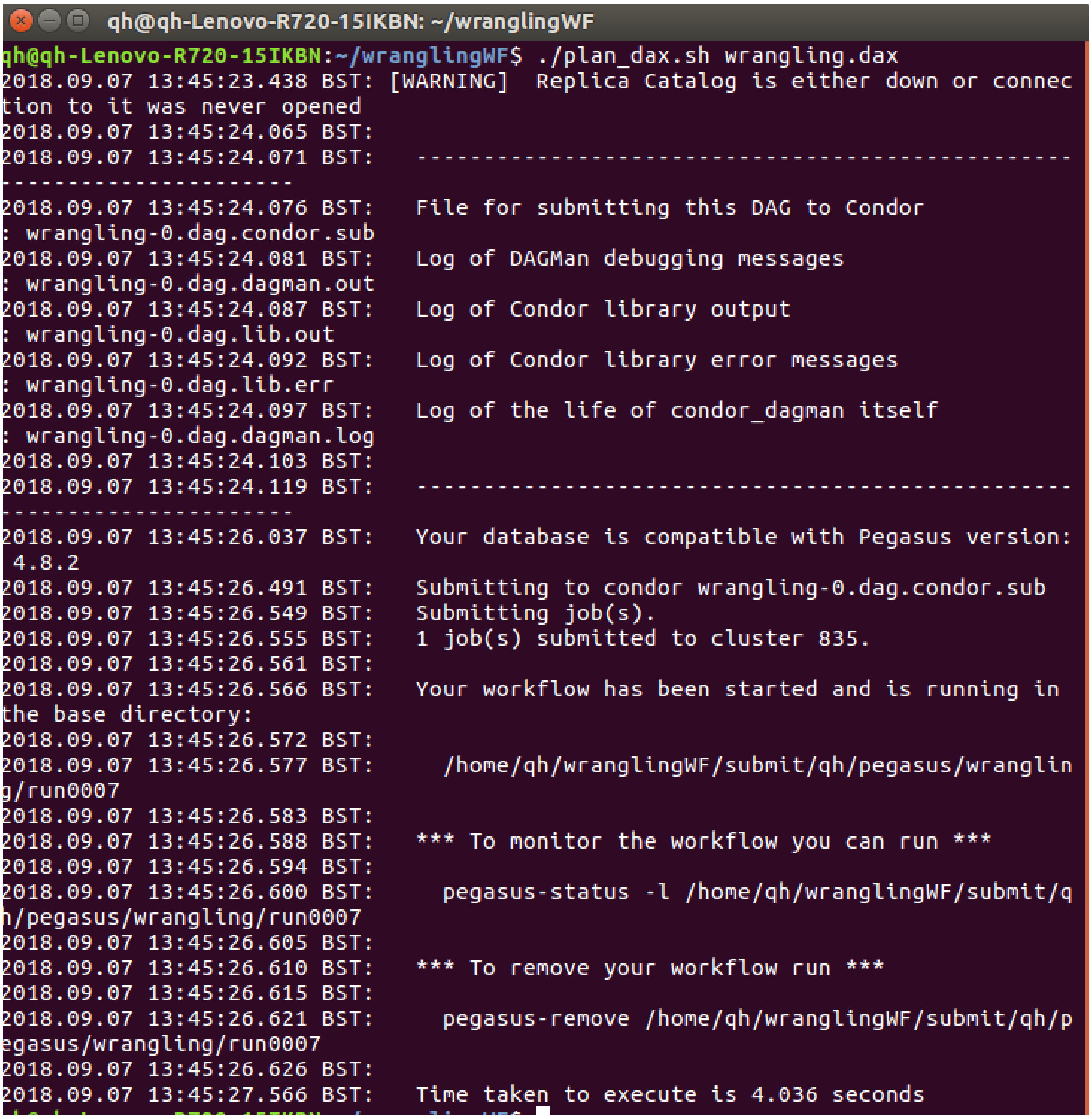
(at https://pegasus.isi.edu/documentation/tutorial_catalogs.php)

Step 6 submitting and running the workflow

Using the following command line to submitting and running the workflow.

```
$ ./plan_dax.sh wrangling.dax
```

And then the submit file will generate a new executable workflow. The path of this workflow will be presented when you use the above command automatically. Like this:



```
qh@qh-Lenovo-R720-15IKBN:~/wranglingWF$ ./plan_dax.sh wrangling.dax
2018.09.07 13:45:23.438 BST: [WARNING]  Replica Catalog is either down or connec
tion to it was never opened
2018.09.07 13:45:24.065 BST:
2018.09.07 13:45:24.071 BST:   -----------------------------------------------
-----------------------
2018.09.07 13:45:24.076 BST:   File for submitting this DAG to Condor
: wrangling-0.dag.condor.sub
2018.09.07 13:45:24.081 BST:   Log of DAGMan debugging messages
: wrangling-0.dag.dagman.out
2018.09.07 13:45:24.087 BST:   Log of Condor library output
: wrangling-0.dag.lib.out
2018.09.07 13:45:24.092 BST:   Log of Condor library error messages
: wrangling-0.dag.lib.err
2018.09.07 13:45:24.097 BST:   Log of the life of condor_dagman itself
: wrangling-0.dag.dagman.log
2018.09.07 13:45:24.103 BST:
2018.09.07 13:45:24.119 BST:   -----------------------------------------------
-----------------------
2018.09.07 13:45:26.037 BST:   Your database is compatible with Pegasus version:
 4.8.2
2018.09.07 13:45:26.491 BST:   Submitting to condor wrangling-0.dag.condor.sub
2018.09.07 13:45:26.549 BST:   Submitting job(s).
2018.09.07 13:45:26.555 BST:   1 job(s) submitted to cluster 835.
2018.09.07 13:45:26.561 BST:
2018.09.07 13:45:26.566 BST:   Your workflow has been started and is running in
the base directory:
2018.09.07 13:45:26.572 BST:
2018.09.07 13:45:26.577 BST:     /home/qh/wranglingWF/submit/qh/pegasus/wranglin
g/run0007
2018.09.07 13:45:26.583 BST:
2018.09.07 13:45:26.588 BST:   *** To monitor the workflow you can run ***
2018.09.07 13:45:26.594 BST:
2018.09.07 13:45:26.600 BST:     pegasus-status -l /home/qh/wranglingWF/submit/q
h/pegasus/wrangling/run0007
2018.09.07 13:45:26.605 BST:
2018.09.07 13:45:26.610 BST:   *** To remove your workflow run ***
2018.09.07 13:45:26.615 BST:
2018.09.07 13:45:26.621 BST:     pegasus-remove /home/qh/wranglingWF/submit/qh/p
egasus/wrangling/run0007
2018.09.07 13:45:26.626 BST:
2018.09.07 13:45:27.566 BST:   Time taken to execute is 4.036 seconds
```

When the workflow is running ,we can use following commands to monitor the workflow

```
pegasus-status -w  /executable/workflow/path
```

(at https://pegasus.isi.edu/documentation/monitoring_debugging_stats.php#workflow_status)

Finally, we can find the output data sets in the output path.