

Introduction to SLAC Analysis Computing Facility

Shawfeng Dong, Yeeting Li, Wei Yang

Introduction and resources

- SLAC Unix account, as part of the benefit of becoming a SLAC Lab user.
 - SLAC is a federal government (DOE) facility, so
 - It can take days or even weeks to become a SLAC Laboratory user, so
 - act/apply as early as possible.
- Interactive Login and Remote X-window
- GPFS home directory and private data space
 - 100GB/user home, 2TB/user private data, up to 10TB/user
- RUCIO storage (for ATLAS data products)
- Batch (LSF)
- New things: JupyterLab/PyRoot, etc. Xcache to access data
 - Think of them as β test stage

Contents of next 10+ slides

After this, there will be tutorials on how to get started at BNL and SLAC Analysis Facility (aka. Shared Tier 3)

As you work with the tutorial materials, details in the next 10+ slides will serve as references to the SLAC environment

- We will quickly fly over those detail slides

For SLAC specific questions, Email to

- SLAC AF user mailing group: atlas-us-slac-acf@cern.ch
- Unix related questions: unix-admin@slac.stanford.edu
 - SLAC personnels are in both lists

Where to start

Become a SLAC Lab User **FIRST**, then apply a SLAC unix account

- <https://atlas.slac.stanford.edu/atlas-support-center>

2019 SLAC Summer Institute

See past related SLAC events »

USER INFORMATION

User Registration

Computer Account

Using the SLAC computing
resources



SLAC hosts a U.S. ATLAS Center for the community.

- We organize jamborees on physics analysis, analysis
- We provide hands-on experimental opportunities for u

Detailed computing documents

- <https://confluence.slac.stanford.edu/display/Atlas/Computing>

Interactive Login, GPFS space

```
ssh -Y <user>@rhel6-64.slac.stanford.edu    or
ssh -Y <user>@centos7.slac.stanford.edu — preferred, ATLAS moves to CentOS 7
```

```
$ cd /gpfs/slac/atlas/fs1/ u /yangw ← user "yangw" home directory
```

```
$ df -h .
```

Filesystem	Size	Used	Avail	Use%	Mounted on
remote-atlas-fs1	100G	219M	100G	1%	/gpfs/slac/atlas/fs1

```
$ cd /gpfs/slac/atlas/fs1/ d /bdouglas ← user "bdouglas" private data space
```

```
$ df -h .
```

Filesystem	Size	Used	Avail	Use%	Mounted on
remote-atlas-fs1	5.0T	0	5.0T	0%	/gpfs/slac/atlas/fs1

In GPFS (only), this is disk quota

CVMFS, ATLAS environment

CVMFS is available on all interactive login nodes and batch nodes

- Example of a very minimum ATLAS environment setup via CVMFS:

```
export RUCIO_ACCOUNT="yangw"  
export  
ATLAS_LOCAL_ROOT_BASE=/cvmfs/atlas.cern.ch/repo/ATLASLocalRootBase  
source ${ATLAS_LOCAL_ROOT_BASE}/user/atlasLocalSetup.sh  
localSetupRucioClients # optional, setup RUCIO client tools  
localSetupPandaClient # optional, setup Panda client tools  
...
```

All nodes also have /scratch for you to store temporary data

- **Create and use your own directory under /scratch !**
- Clean your debris when you are done
- SLAC will periodically delete files under /scratch that are 7+ day old

IBM Platform LSF batch system

In LSF, your batch jobs inherit all of your current working environment:

- Working directory
- Unix environment variables
- But not Shell aliases



This is different from the
Condor batch system

Trick on X509 proxy:

- Before submitting a job, put a X509 proxy under your home directory

```
export X509_USER_PROXY=$HOME/.globus/x509up_u$(id -u)  
voms-proxy-init -voms atlas
```

- This way all of your batch jobs can access your X509 proxy
 - Note that your job may be pending in the batch queue for a while. So don't forget to renew your X509 proxy

Your batch jobs

Submit a job

```
$ bsub -q atl-analq -R 'select[centos7]' < myjob.sh
Job <59535> is submitted to queue <atl-analq>.
```

Check the job

```
$ bjobs 59535
```

JOBID	USER	STAT	QUEUE	FROM_HOST	EXEC_HOST	JOB_NAME	SUBMIT_TIME
59535	yangw	RUN	atl-analq	cent7d	kiso0057	myjob.sh	Jul 31 23:49

```
$ bjobs 59535
```

JOBID	USER	STAT	QUEUE	FROM_HOST	EXEC_HOST	JOB_NAME	SUBMIT_TIME
59535	yangw	DONE	atl-analq	cent7d	kiso0057	myjob.sh	Jul 31 23:49

Kill the job

```
$ bkill 59535
```

```
Job <59535>: Job has already finished
```

Detail and more complex examples: <https://confluence.slac.stanford.edu/display/Atlas/Run+batch+jobs>

Submit a Panda/Grid Analysis Job

[Panda can run many type of jobs on the Grid](#) ← your entrance point of docs

- “prun” is one of them: [ROOT/general jobs](#) (Run “prun --help”)
 - General: ROOT (CINT, C++, PyROOT) jobs, python jobs or [AthenaROOTAccess jobs \(ARA\)](#)

Very simple example ARA jobs (from the above prun URL)

- Test on local machine: `python aratest.py - AOD1.pool.root,AOD2.pool.root`
- Submit to Panda: `prun --exec "python aratest.py - %IN" --useAthenaPackages \`
`--inDS valid1.006384.PythiaH120gamgam.recon.AOD.e322_s412_r577/ \`

Your input dataset. Panda will submit multiple Grid jobs to process all input files in the input dataset **in parallel**, each job creates an output file (below)

```
--outDS user.me.test01 --outputs out.root \
```

Your output dataset: contains all output files user.me.test01._00001.out.root, etc.

```
--destSE SWT2_CPB_LOCALGROUPDISK
```

Send all your entire output dataset to a US LOCALGROUPDISK, such as this one

Monitor the status of your Panda jobs: https://bigpanda.cern.ch/dash/analysis/#cloud_US

Playing with RUCIO

List files in a dataset:

RUCIO Data Identifier: **scope** : **file**

```
$ rucio list-files data18_13TeV:data18_13TeV.00348885.physics_Main.deriv.DAOD_EXOT12.f937_m1972_p3553_tid14278917_00
+-----+-----+-----+-----+-----+
| SCOPE:NAME                               | GUID                               | ADLER32   | FILESIZE | EVENTS |
+-----+-----+-----+-----+-----+
| data18_13TeV:DAOD_EXOT12.14278917._000001.pool.root.1 | F1ED3FDA-79BA-6F4E-9DAA-36151F48BD63 | ad:1bc9a126 | 17.338 MB | 363 |
| ...                                       | ...                               | ...       | ...      | ...   |
+-----+-----+-----+-----+-----+
Total files : 83
Total size : 9.593 GB
Total events : 274085
```

```
$ rucio list-file-replicas --protocol=root --pfn data18_13TeV:DAOD_EXOT12.14278917._000001.pool.root.1
root://eosatlas.cern.ch:1094//eos/atlas/atlasdatadisk/rucio/data18_13TeV/da/ea/DAOD_EXOT12.14278917._000001.pool.root.1
root://lfn1.in2p3.fr:1094//dpm/in2p3.fr/home/atlas/atlasdatadisk/rucio/data18_13TeV/da/ea/DAOD_EXOT12.14278917._000001.pool.root.1
root://gk06.atlas-swt2.org:1094//xrd/atlasscratchdisk/rucio/data18_13TeV/da/ea/DAOD_EXOT12.14278917._000001.pool.root.1
root://dcgftp.usatlas.bnl.gov:1094//pnfs/usatlas.bnl.gov/LOCALGROUPDISK/rucio/data18_13TeV/da/ea/DAOD_EXOT12.14278917._000001.pool.root.1
```

- There are multiple copies, including two US sites: **SWT2_CPB** and **BNL**
- Note: Some of these functions are integrated into the **pnfs_ls.py** (See Shuwei's tutorial)

RUCIO storage at SLAC

Names: SLACXRD_{DATADISK, SCRATCHDISK, LOCALGROUPDISK}

- All T1s and T2s, as well as SLAC & BNL T3/AF have them, in different prefix
 - No direct write access to RUCIO disks
- Please use [R2D2](#) to move ATLAS data product in and out
- Or use “rucio upload” to upload your own dataset
- Or --destSE in your Panda job

To access data in RUCIO storage from SLAC interactive/batch nodes:

- Use RUCIO command to find the location:
 - `root://griddev03.slac.stanford.edu:2094//xrootd/atlas/...`
- **If the file is at SLAC**, replace the host name (above) in the URL
 - By `root://atlrdr1//xrootd/atlas/...`
 - Access directly using `TFile *f = TFile::Open("root://atlrdr1//...")`
 - copy to scratch `xrdcp root://atlrdr1//.../file.root /scratch/me/file.root`

Access Remote data: using Xcache

This is experimental at both SLAC and BNL

- We don't have permanent machines to run Xcache yet
- Subject to changes

Current Xcache hosts:

- atlfax.slac.stanford.edu
- xrootd03.usatlas.bnl.gov

Use one of the following URLs (change the obvious mistake here of course)

1. `root://atlfax.slac.stanford.edu//atlas/rucio/data18_13TeV:DAOD_EXOT12.14278917._000001.pool.root.1`

- Only work for RUCIO managed files
- You don't need to know where is the data source
- Data will likely coming from SWT2_CPB because they are close to SLAC

Telling Xcache that this path is a global Logical Filename (gLFN)

2. `root://atlfax.slac.stanford.edu//root://dcgftp.usatlas.bnl.gov:1094//pnfs/usatlas.bnl.gov/LOCALGROUPDISK/rucio/data18_13TeV/da/ea/DAOD_EXOT12.14278917._000001.pool.root.1`

- Work for both RUCIO managed file and private files.
- You do need to know where is the data source

Remote Xwindow Display

If you are close to SLAC

- `ssh -Y centos7.slac.stanford.edu /usr/bin/xeyes`

If you are far away from SLAC, use FastX to speed up remote Xwindow display

- <https://confluence.slac.stanford.edu/pages/viewpage.action?pageId=205985167>

If you are roaming

- Try JupyterLab (next)
- JupyterLab also has lots of modern data science tools

JupyterLab

Experimental

- We are ready to provide these service though you will be our first real users

JupyterLab runs on precious, **shared**, powerful HW w/ GPUs

- Terminal in JupyterLab is convenient, but not a replacement of login nodes
- Don't waste the resource for tasks that can be done elsewhere, such as "cp"

Entrance Screen

<https://jupyter.slac.stanford.edu>

Spawner Options

SuperCDMS Images

- CDMS JupyterLab Image - v 0.1.5
- CDMS JupyterLab Image - v 0.1.6 Beta

SLAC Machine Learning Images

- SLAC JupyterLab Image (GPU) v20190610.2
- SLAC JupyterLab Image (GPU) v20190606.1
- SLAC JupyterLab Image (GPU) v20190514.0
- SLAC JupyterLab Image (GPU) v20190425.1

ATLAS Images

- ATLAS Jupyterlab Image - v01

LSST Isstsqre/sciplat-lab Images

updated at Thu May 9 16:54:30 2019 UTC

- Weekly 2019_22
- Weekly 2019_21
- Weekly 2019_20
- Weekly 2019_19
- Weekly 2019_18
- Release 17_0_1

Cryo-EM CryoSPARC Images

- cryoSPARC v2.5.0-5 (GPU)

Spawn

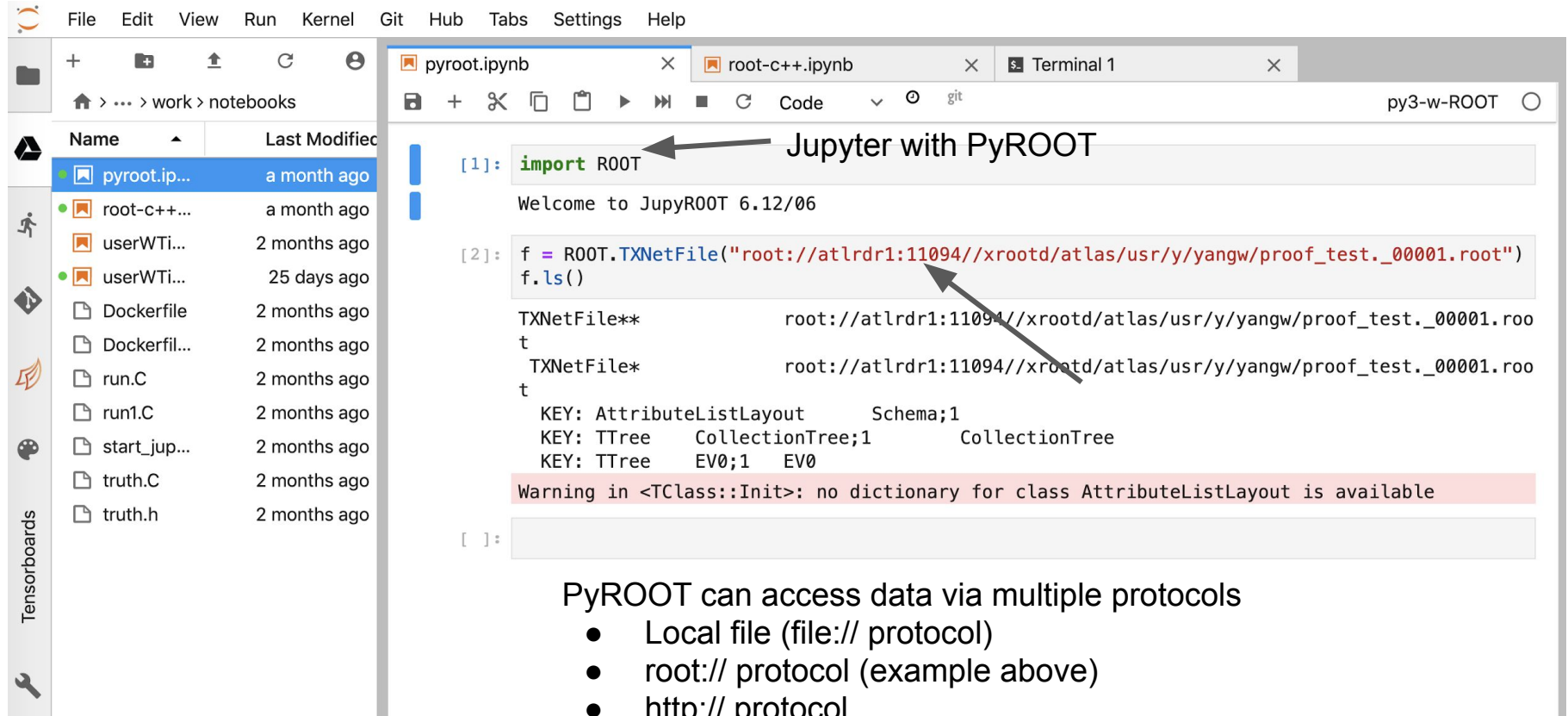
Logged in using my SLAC unix account

A list of ATLAS images can be added at here

- All are Docker images
- Better to run on dedicated ATLAS hardware

Start your Jupyter container

JupyterLab with PyROOT



The screenshot shows the JupyterLab interface with a notebook titled 'pyroot.ipynb'. The left sidebar shows a file browser with a list of notebooks and files. The main area displays the notebook content:

```
[1]: import ROOT
Welcome to JupyROOT 6.12/06

[2]: f = ROOT.TXNetFile("root://atlrdr1:11094//xrootd/atlas/usr/y/yangw/proof_test._00001.root")
f.ls()

TXNetFile**      root://atlrdr1:11094//xrootd/atlas/usr/y/yangw/proof_test._00001.root
t
TXNetFile*       root://atlrdr1:11094//xrootd/atlas/usr/y/yangw/proof_test._00001.root
t
KEY: AttributeListLayout      Schema;1
KEY: TTree      CollectionTree;1      CollectionTree
KEY: TTree      EV0;1      EV0

Warning in <TClass::Init>: no dictionary for class AttributeListLayout is available

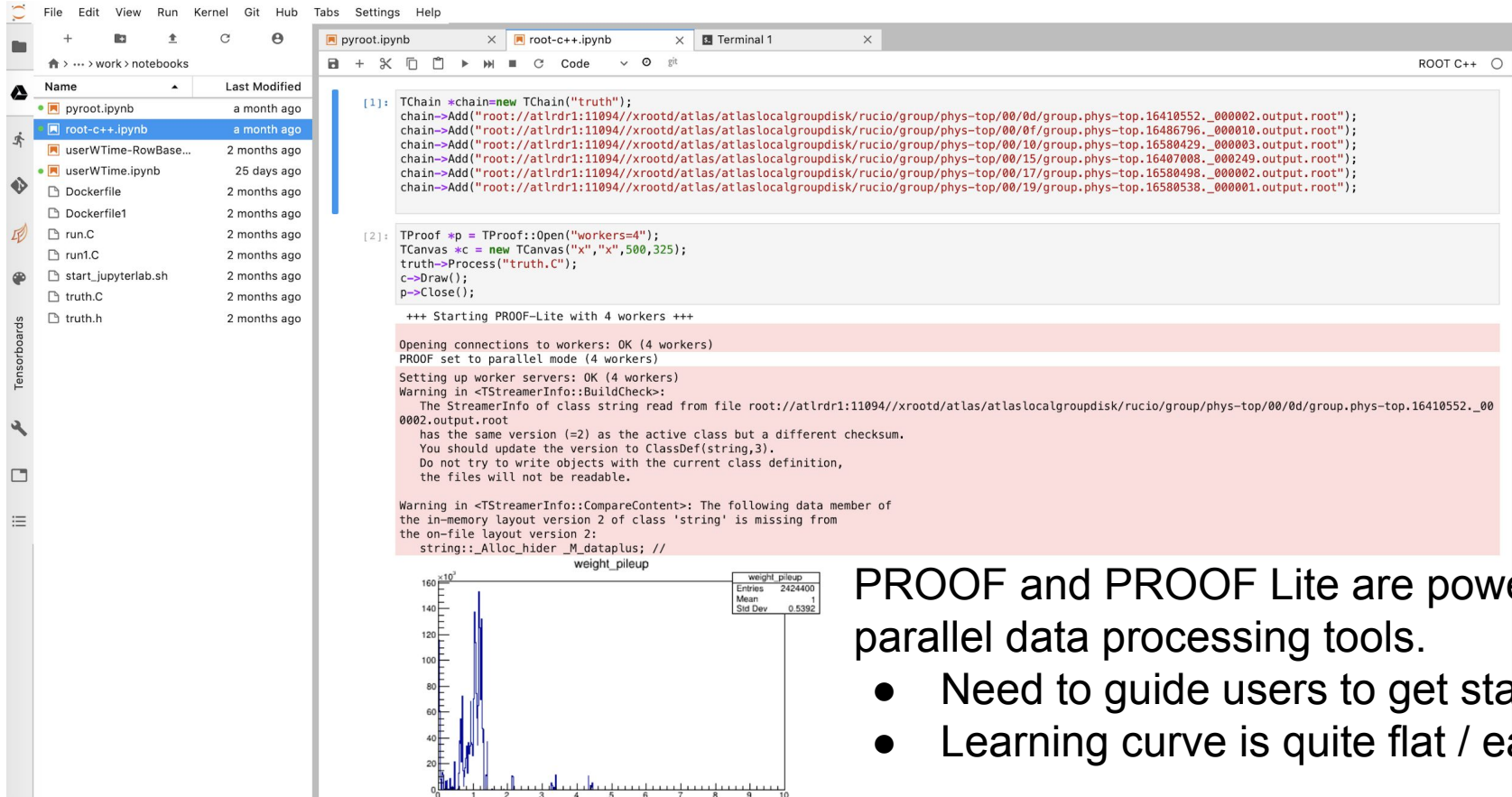
[ ]:
```

Annotations in the image include an arrow pointing to the `import ROOT` line with the text "Jupyter with PyROOT", and another arrow pointing to the file path in the code with the text "PyROOT can access data via multiple protocols".

PyROOT can access data via multiple protocols

- Local file (file:// protocol)
- root:// protocol (example above)
- http:// protocol

ROOT C++ notebook with PROOF Lite



The screenshot shows a JupyterLab interface with a file browser on the left and a notebook editor on the right. The notebook contains two code cells. The first cell defines a TChain and adds several file paths. The second cell creates a TProof object with 4 workers, a TCanvas, and a process. Below the code, there is a terminal output showing the start of PROOF-Lite and a warning about a version mismatch in the string class. At the bottom, a histogram plot is shown with a statistics box.

```
[1]: TChain *chain=new TChain("truth");
chain->Add("root://atlrdr1:11094//xrootd/atlas/atlas.localgroupdisk/rucio/group/phys-top/00/0d/group.phys-top.16410552._000002.output.root");
chain->Add("root://atlrdr1:11094//xrootd/atlas/atlas.localgroupdisk/rucio/group/phys-top/00/0f/group.phys-top.16486796._000010.output.root");
chain->Add("root://atlrdr1:11094//xrootd/atlas/atlas.localgroupdisk/rucio/group/phys-top/00/10/group.phys-top.16580429._000003.output.root");
chain->Add("root://atlrdr1:11094//xrootd/atlas/atlas.localgroupdisk/rucio/group/phys-top/00/15/group.phys-top.16407008._000249.output.root");
chain->Add("root://atlrdr1:11094//xrootd/atlas/atlas.localgroupdisk/rucio/group/phys-top/00/17/group.phys-top.16580498._000002.output.root");
chain->Add("root://atlrdr1:11094//xrootd/atlas/atlas.localgroupdisk/rucio/group/phys-top/00/19/group.phys-top.16580538._000001.output.root");

[2]: TProof *p = TProof::Open("workers=4");
TCanvas *c = new TCanvas("x","x",500,325);
truth->Process("truth.C");
c->Draw();
p->Close();

+++ Starting PROOF-Lite with 4 workers +++

Opening connections to workers: OK (4 workers)
PROOF set to parallel mode (4 workers)
Setting up worker servers: OK (4 workers)
Warning in <TStreamerInfo::BuildCheck>:
  The StreamerInfo of class string read from file root://atlrdr1:11094//xrootd/atlas/atlas.localgroupdisk/rucio/group/phys-top/00/0d/group.phys-top.16410552._000002.output.root
  has the same version (=2) as the active class but a different checksum.
  You should update the version to ClassDef(string,3).
  Do not try to write objects with the current class definition,
  the files will not be readable.

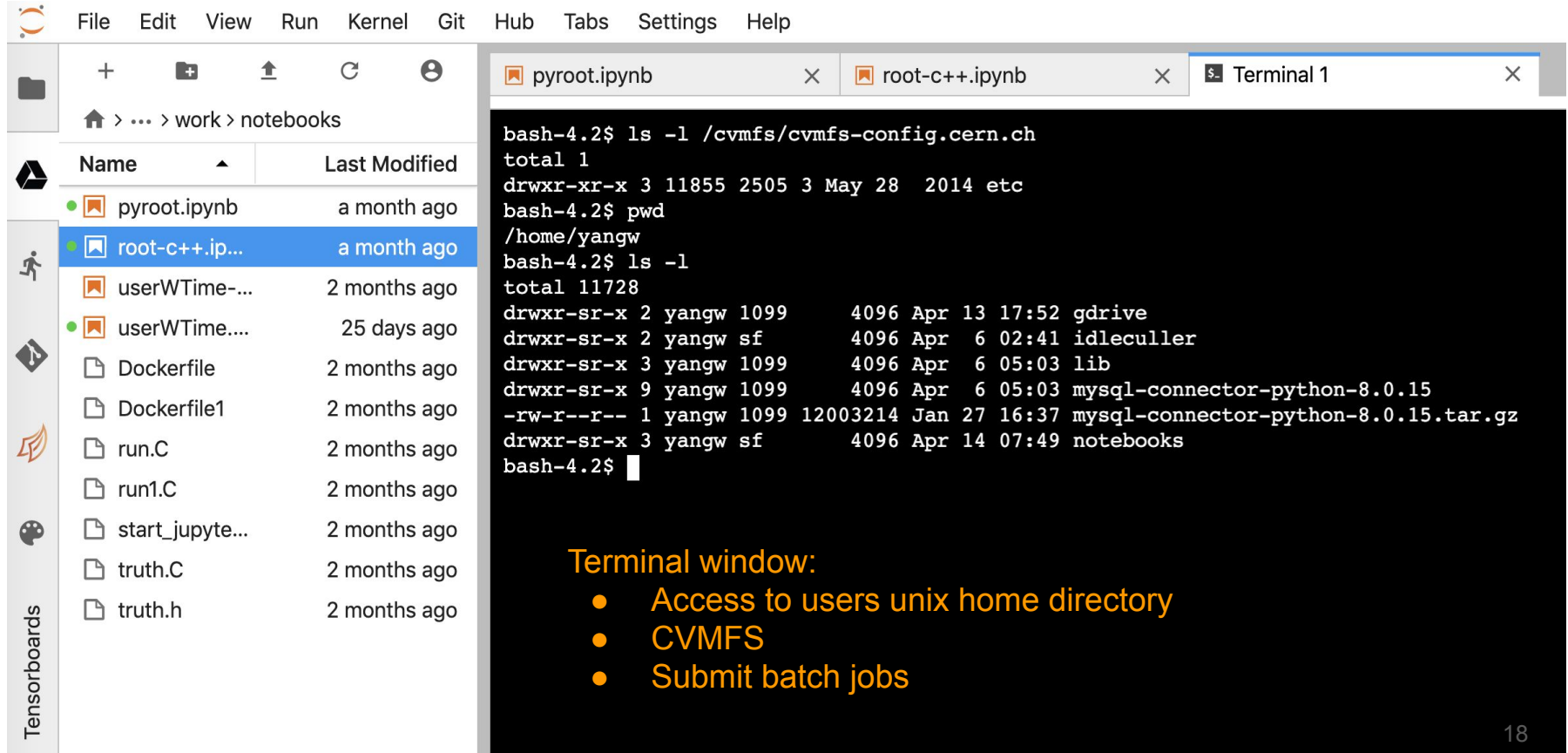
Warning in <TStreamerInfo::CompareContent>: The following data member of
the in-memory layout version 2 of class 'string' is missing from
the on-file layout version 2:
string::_Alloc_hider_M_dataplus; //
weight_pileup
```

weight_pileup	
Entries	2424400
Mean	1
Std Dev	0.5392

PROOF and PROOF Lite are powerful parallel data processing tools.

- Need to guide users to get started
- Learning curve is quite flat / easy.

JupyterLab as interactive login node



File Edit View Run Kernel Git Hub Tabs Settings Help

work > notebooks

Name	Last Modified
pyroot.ipynb	a month ago
root-c++.ip...	a month ago
userWTime-...	2 months ago
userWTime....	25 days ago
Dockerfile	2 months ago
Dockerfile1	2 months ago
run.C	2 months ago
run1.C	2 months ago
start_jupyte...	2 months ago
truth.C	2 months ago
truth.h	2 months ago

```

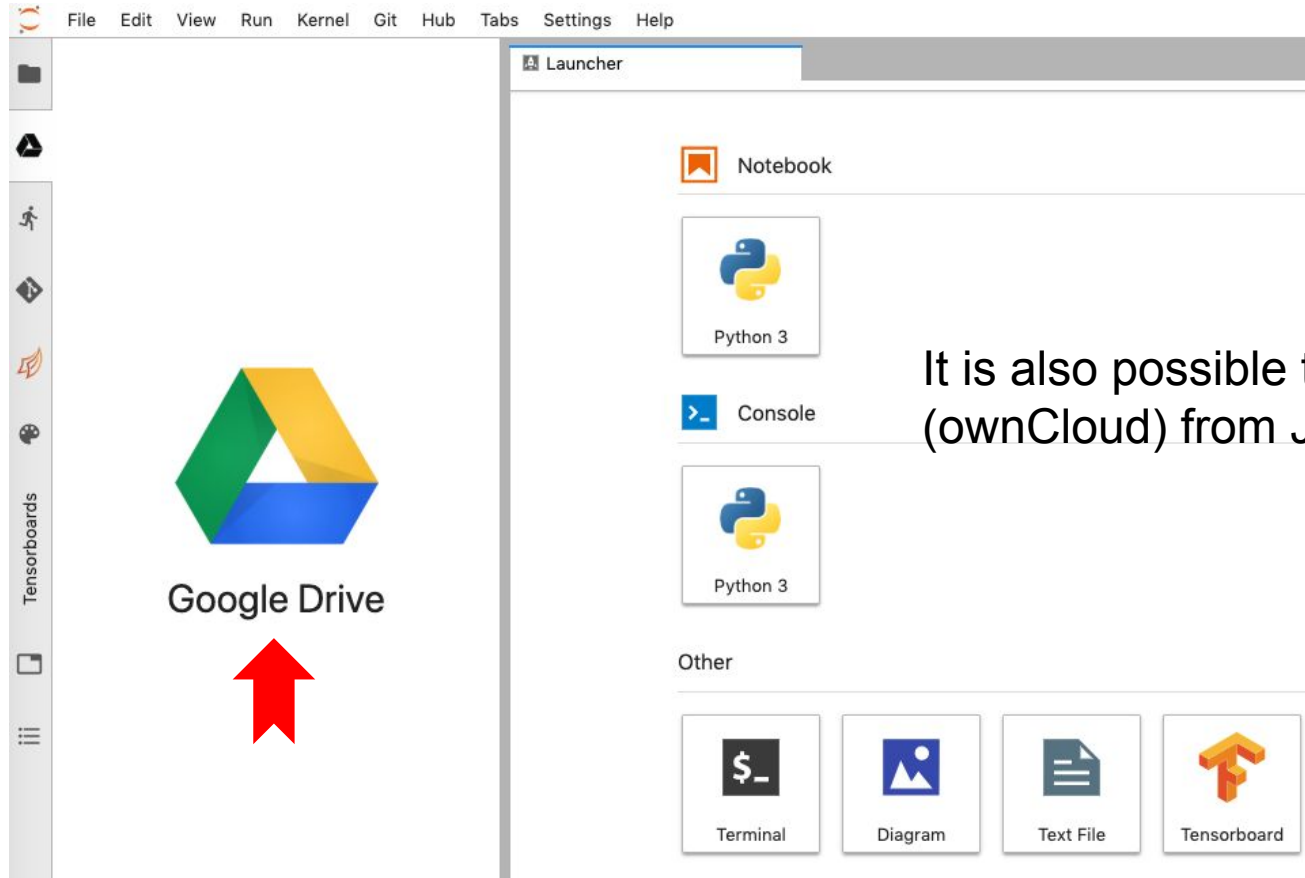
bash-4.2$ ls -l /cvmfs/cvmfs-config.cern.ch
total 1
drwxr-xr-x 3 11855 2505 3 May 28 2014 etc
bash-4.2$ pwd
/home/yangw
bash-4.2$ ls -l
total 11728
drwxr-sr-x 2 yangw 1099 4096 Apr 13 17:52 gdrive
drwxr-sr-x 2 yangw sf 4096 Apr 6 02:41 idleculler
drwxr-sr-x 3 yangw 1099 4096 Apr 6 05:03 lib
drwxr-sr-x 9 yangw 1099 4096 Apr 6 05:03 mysql-connector-python-8.0.15
-rw-r--r-- 1 yangw 1099 12003214 Jan 27 16:37 mysql-connector-python-8.0.15.tar.gz
drwxr-sr-x 3 yangw sf 4096 Apr 14 07:49 notebooks
bash-4.2$

```

Terminal window:

- Access to users unix home directory
- CVMFS
- Submit batch jobs

JupyterLab Integration with Google Drive



It is also possible to access BNL Box (ownCloud) from JupyterLab