ROOTaaS
ROOT as a Service

E. Tejedor, D. Piparo, P. Mató for the ROOT Team

Jupyter Technical Discussion

09/12/2015

This Talk

- The ROOTaaS project
- Integration of ROOT with the Notebook technology
  - ROOTbooks for data analysis
- A new service: ROOT & Jupyter Notebooks within the CERN IT services' portfolio
  - Spotlight on storage

The ROOTaaS Project

Data analysis with ROOT “as a service”

Interface: Notebooks

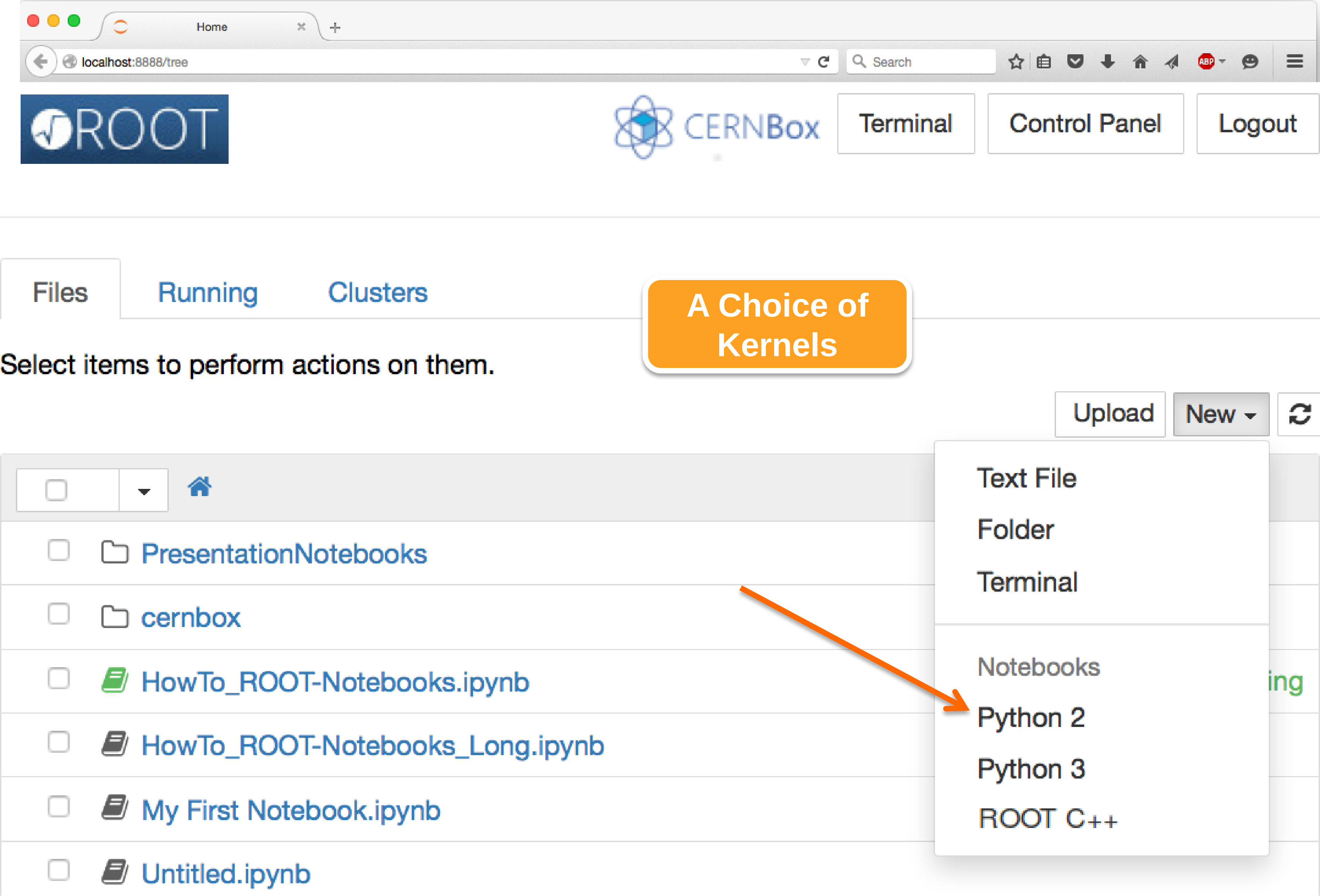
Goals:

- Use ROOT only with a web browser
  - Platform independent ROOT-based data analysis
  - Calculations, input and results “in the cloud”
- Allow easy sharing of scientific results: plots, data, code
  - Storage is crucial
- Simplify teaching of data processing and programming
- Potential integration with other analysis ecosystems: R, Python, …

Integration of ROOT with Notebooks

ROOT

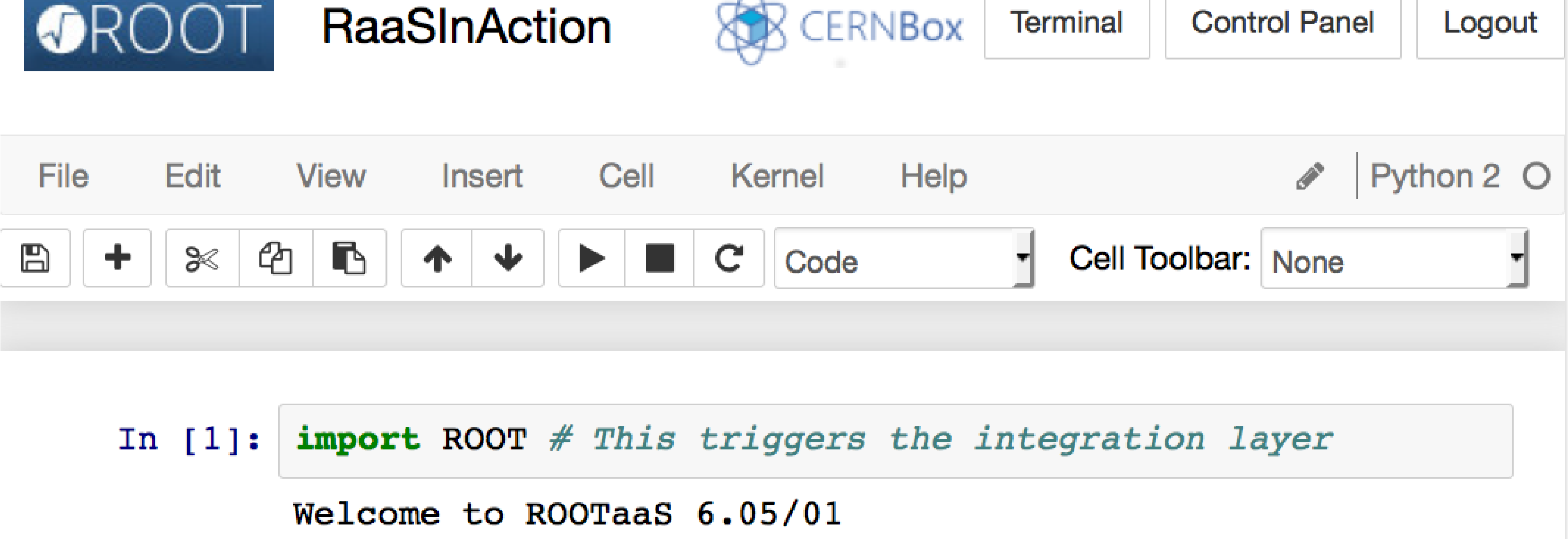
JupyROOT
(ROOT-Jupyter integration)



A Choice of Kernels

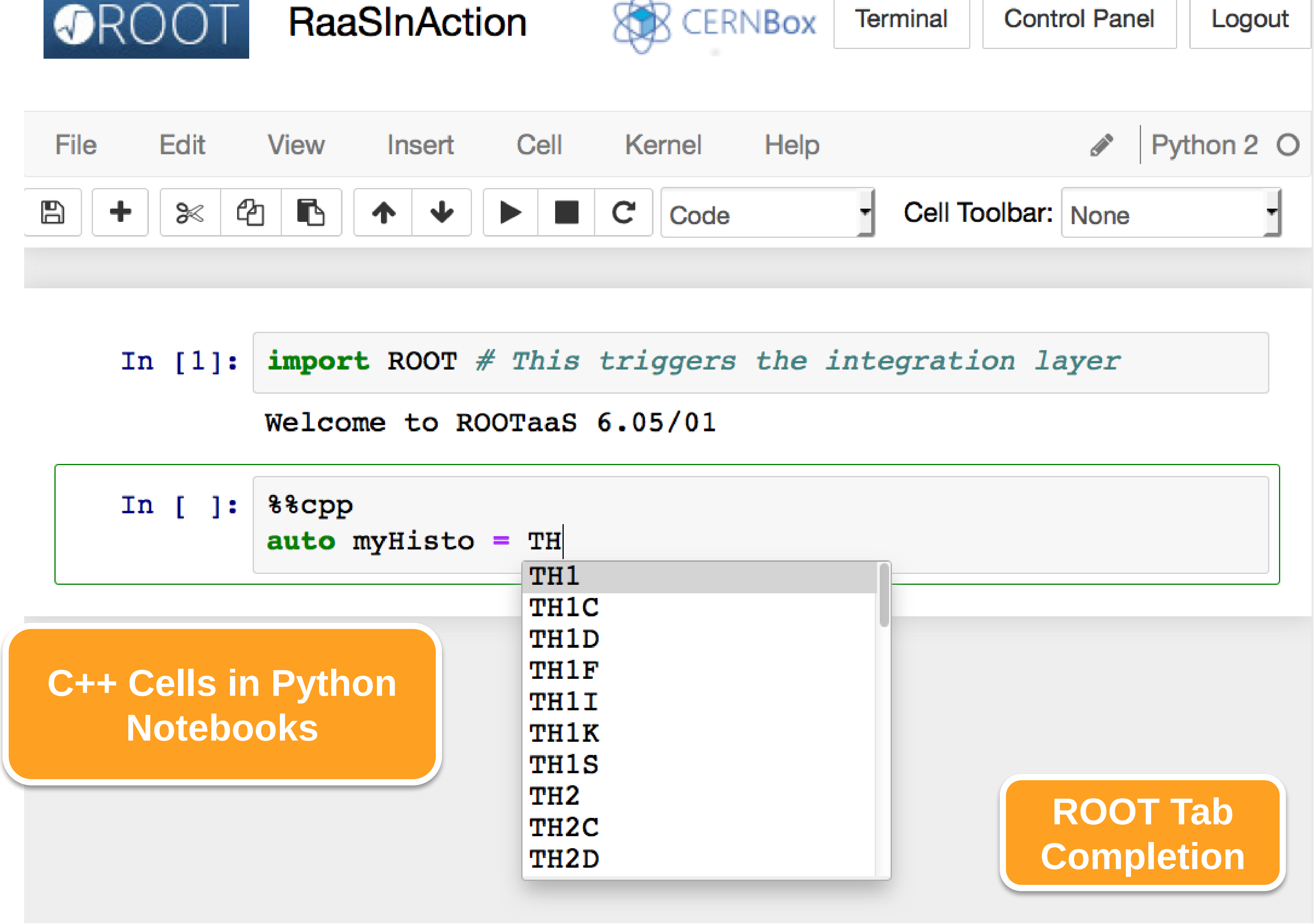
Select items to perform actions on them.

- PresentationNotebooks
- cernbox
- HowTo_ROOT-Notebooks.ipynb
- HowTo_ROOT-Notebooks_Long.ipynb
- My First Notebook.ipynb
- Untitled.ipynb



```
import ROOT # This triggers the integration layer
```

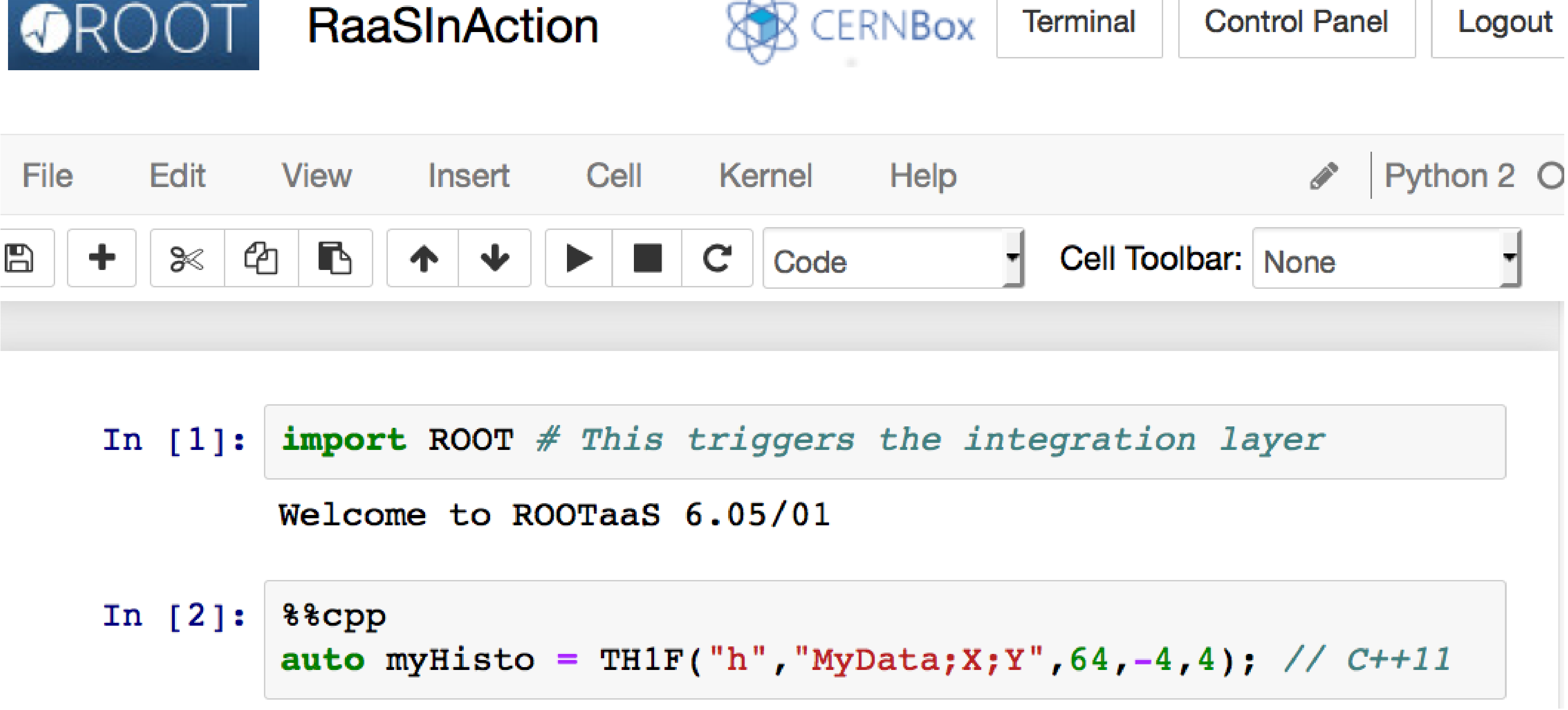
```
Welcome to ROOTaaS 6.05/01
```



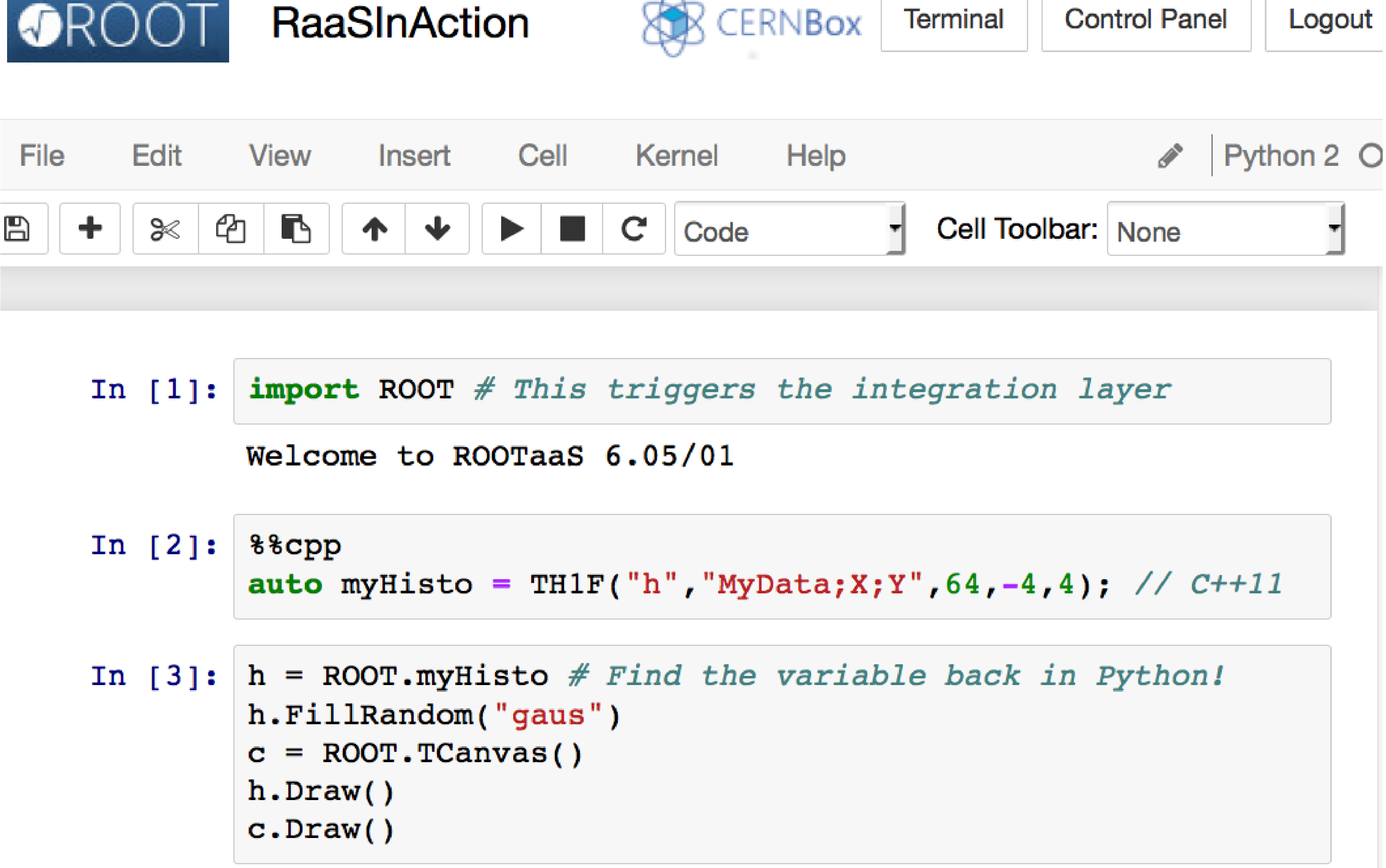
```
In [1]: import ROOT # This triggers the integration layer

        Welcome to ROOTaaS 6.05/01

In [ ]: %%cpp
        auto myHisto = TH
```

C++ Cells in Python Notebooks

ROOT Tab Completion

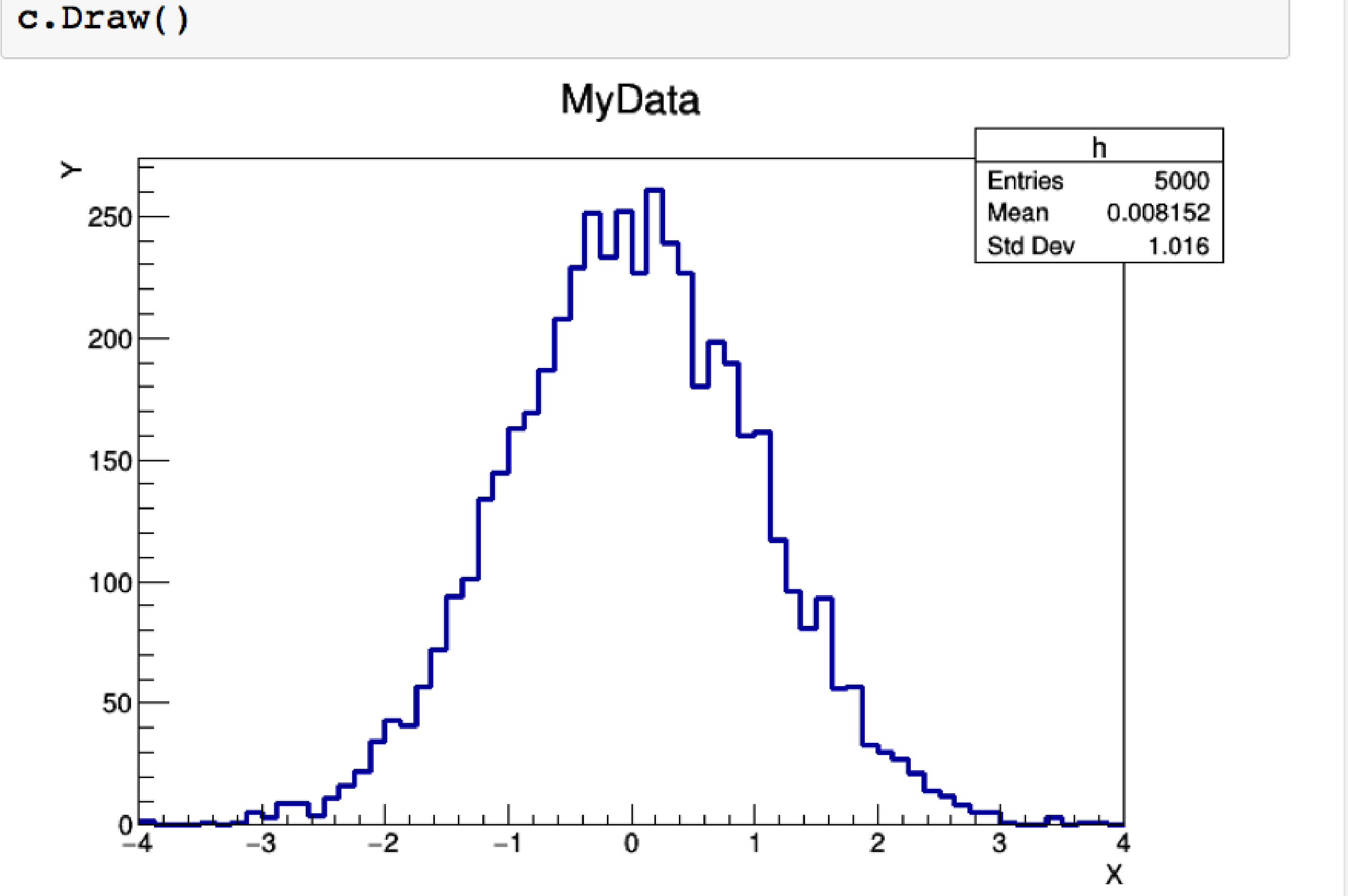


```
In [1]: import ROOT # This triggers the integration layer

        Welcome to ROOTaaS 6.05/01

In [2]: %%cpp
        auto myHisto = TH1F("h","MyData;X;Y",64,-4,4); // C++11
```



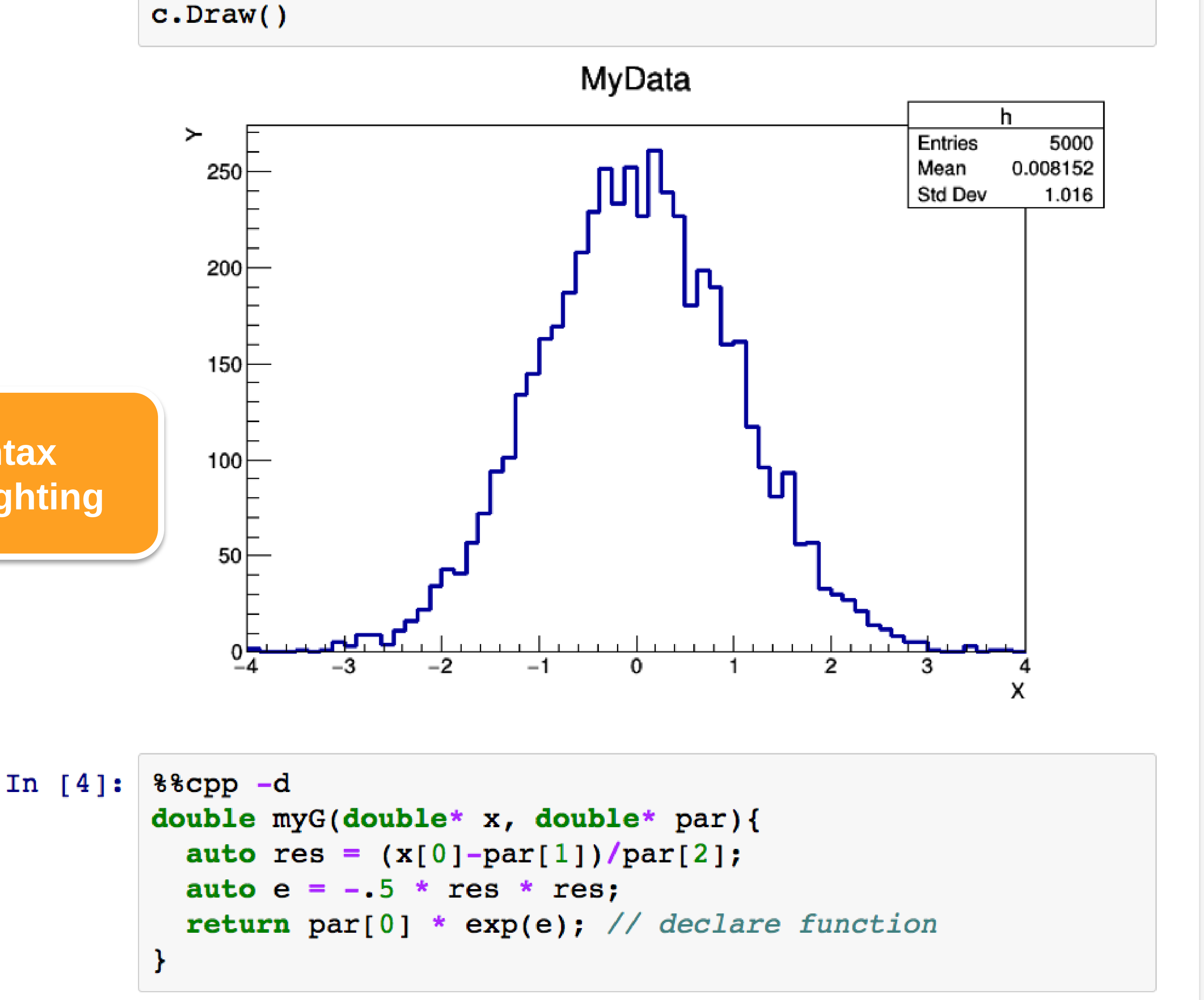
```
In [1]: import ROOT # This triggers the integration layer

        Welcome to ROOTaaS 6.05/01

In [2]: %%cpp
        auto myHisto = TH1F("h","MyData;X;Y",64,-4,4); // C++11

In [3]: h = ROOT.myHisto # Find the variable back in Python!
        h.FillRandom("gaus")
        c = ROOT.TCanvas()
        h.Draw()
        c.Draw()
```

C++-Python Interoperability



```
c.Draw()
```

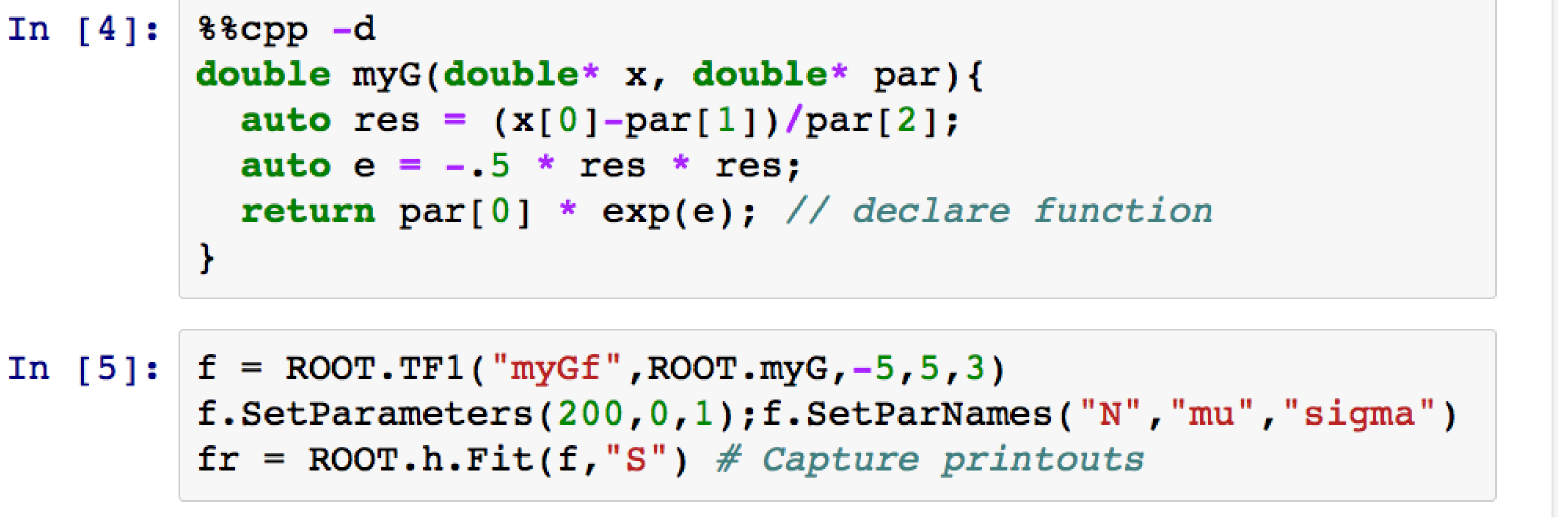
Seamless display of graphics



```
c.Draw()
```

```
In [4]: %%cpp -d
        double myG(double* x, double* par){
          auto res = (x[0]-par[1])/par[2];
          auto e = -.5 * res * res;
          return par[0] * exp(e); // declare function
        }
```

Syntax Highlighting

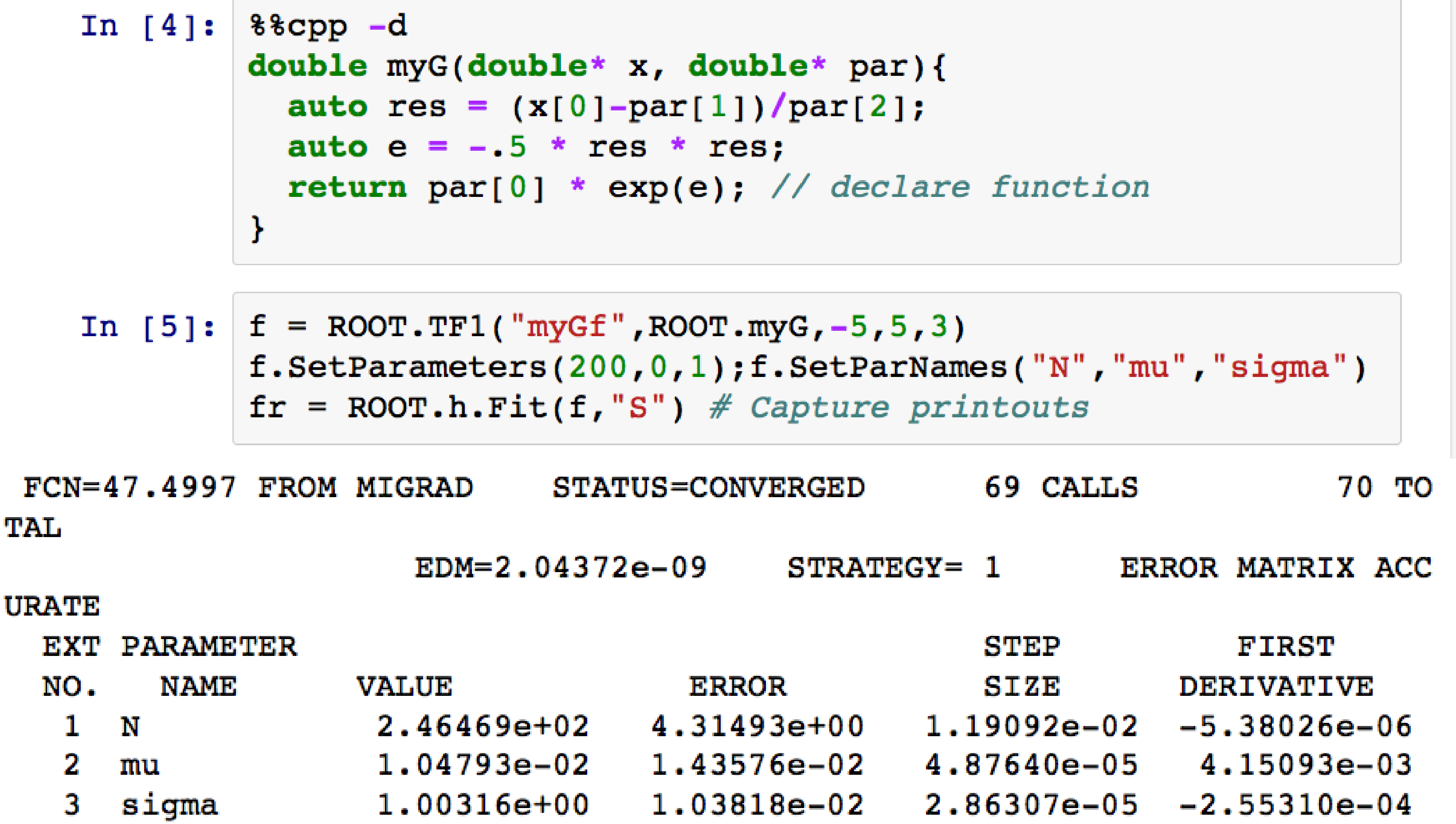


In [4]:

```
%%cpp -d
double myG(double* x, double* par){
  auto res = (x[0]-par[1])/par[2];
  auto e = -.5 * res * res;
  return par[0] * exp(e); // declare function
}
```

In [5]:

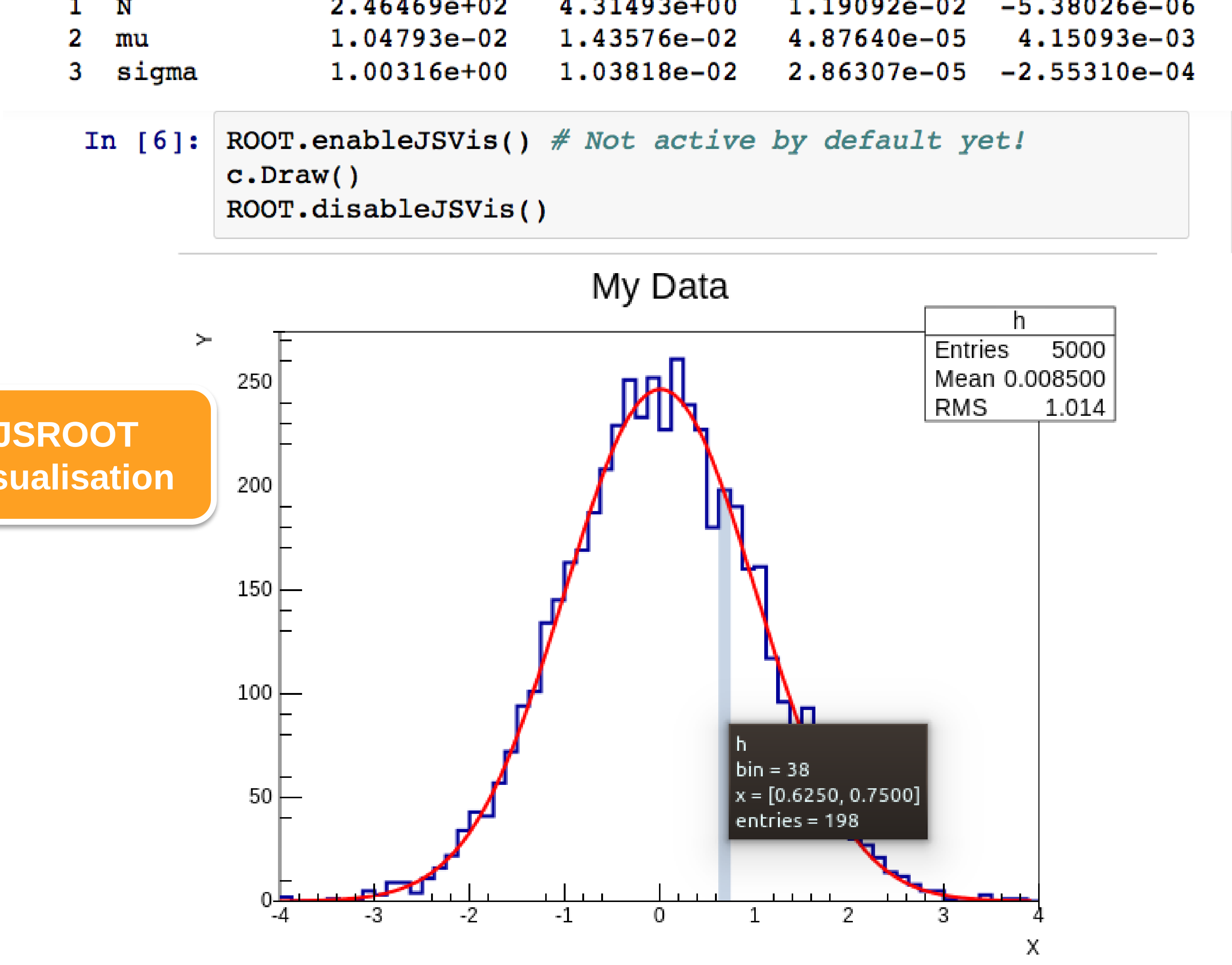
```
f = ROOT.TF1("myGf",ROOT.myG,-5,5,3)
f.SetParameters(200,0,1);f.SetParNames("N","mu","sigma")
fr = ROOT.h.Fit(f,"S") # Capture printouts
```



```
In [4]: %%cpp -d
        double myG(double* x, double* par){
          auto res = (x[0]-par[1])/par[2];
          auto e = -.5 * res * res;
          return par[0] * exp(e); // declare function
        }
```

```
In [5]: f = ROOT.TF1("myGf",ROOT.myG,-5,5,3)
        f.SetParameters(200,0,1);f.SetParNames("N","mu","sigma")
        fr = ROOT.h.Fit(f,"S") # Capture printouts
```

```
 FCN=47.4997 FROM MIGRAD    STATUS=CONVERGED      69 CALLS          70 TO
TAL
                     EDM=2.04372e-09    STRATEGY= 1      ERROR MATRIX ACC
URATE
  EXT PARAMETER                                   STEP         FIRST
  NO.   NAME      VALUE            ERROR          SIZE      DERIVATIVE
   1  N           2.46469e+02   4.31493e+00   1.19092e-02  -5.38026e-06
   2  mu          1.04793e-02   1.43576e-02   4.87640e-05   4.15093e-03
   3  sigma       1.00316e+00   1.03818e-02   2.86307e-05  -2.55310e-04
```

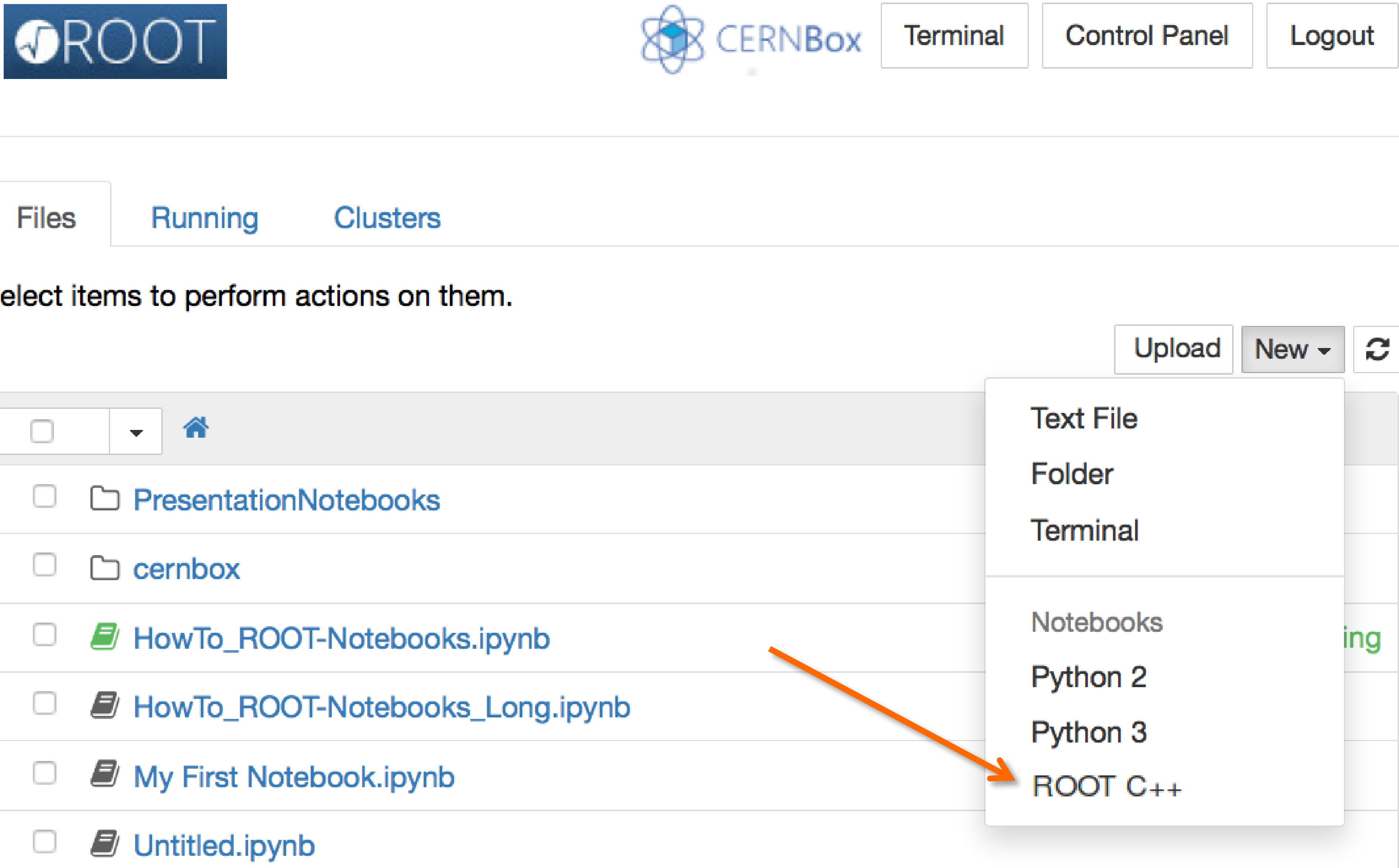


```
1  N          2.46469e+02   4.31493e+00   1.19092e-02  -5.38026e-06
2  mu         1.04793e-02   1.43576e-02   4.87640e-05   4.15093e-03
3  sigma      1.00316e+00   1.03818e-02   2.86307e-05  -2.55310e-04
```

In [6]:

```
ROOT.enableJSVis() # Not active by default yet!
c.Draw()
ROOT.disableJSVis()
```

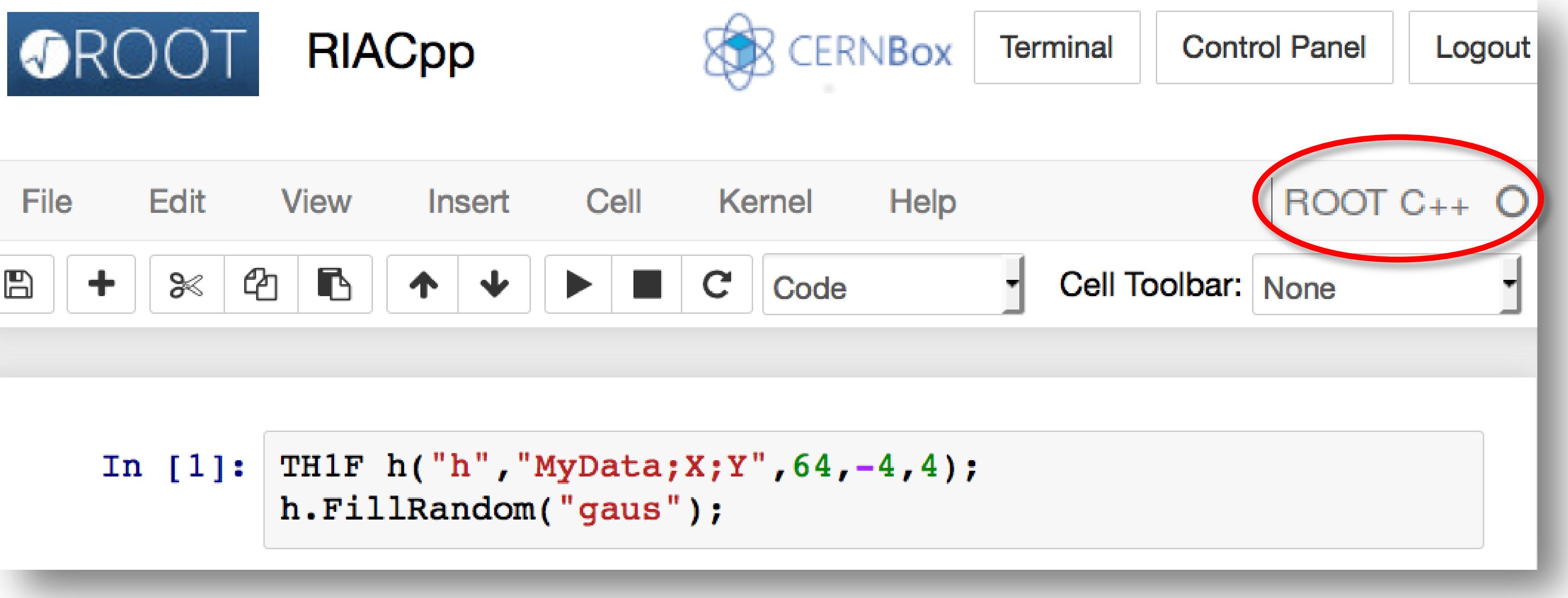
JSROOT Visualisation



Select items to perform actions on them.

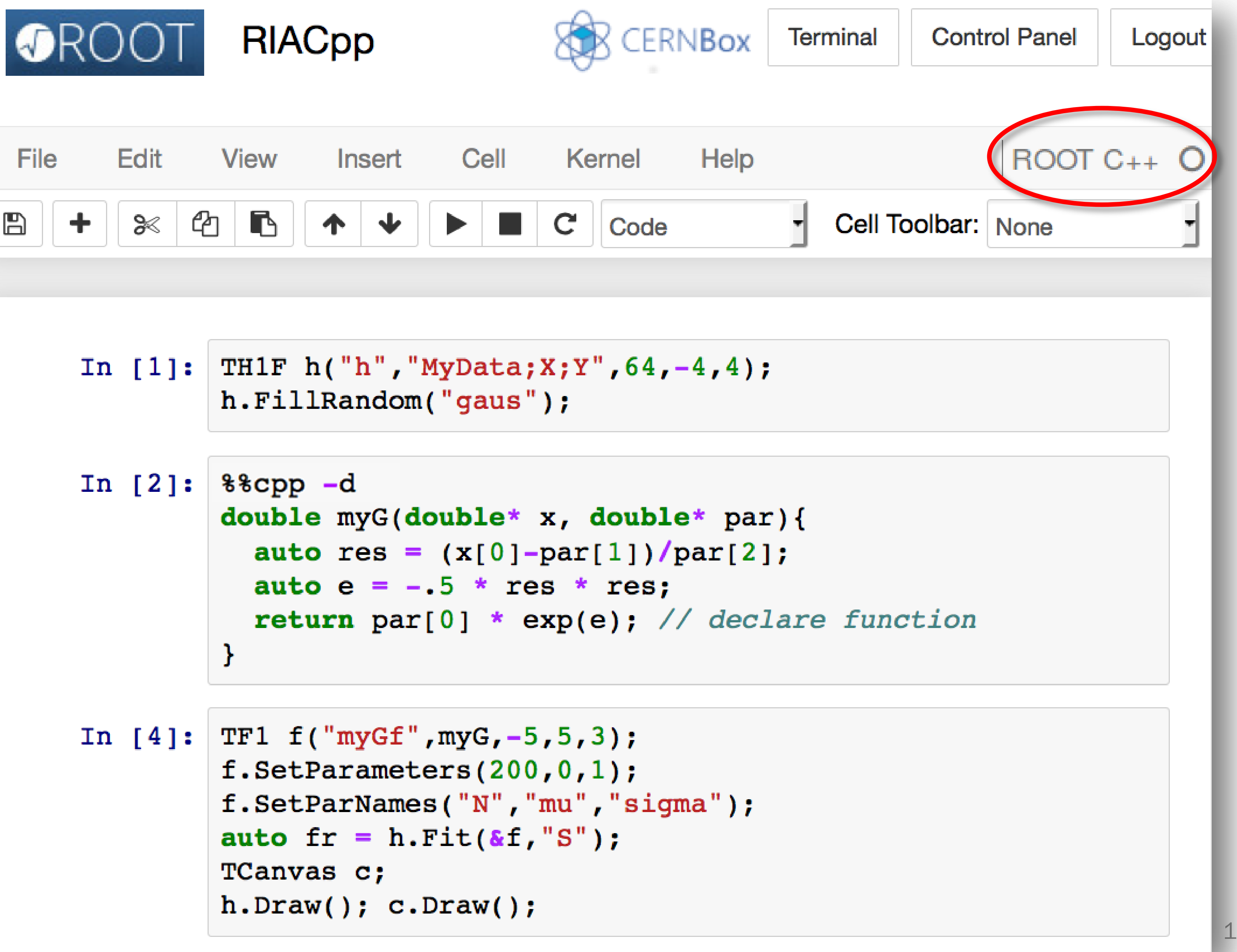
- PresentationNotebooks
- cernbox
- HowTo_ROOT-Notebooks.ipynb
- HowTo_ROOT-Notebooks_Long.ipynb
- My First Notebook.ipynb
- Untitled.ipynb

A C++ ROOTbook



In [1]:
```
TH1F h("h","MyData;X;Y",64,-4,4);
h.FillRandom("gaus");
```

A C++ ROOTbook



In [1]:

```
TH1F h("h","MyData;X;Y",64,-4,4);
h.FillRandom("gaus");
```

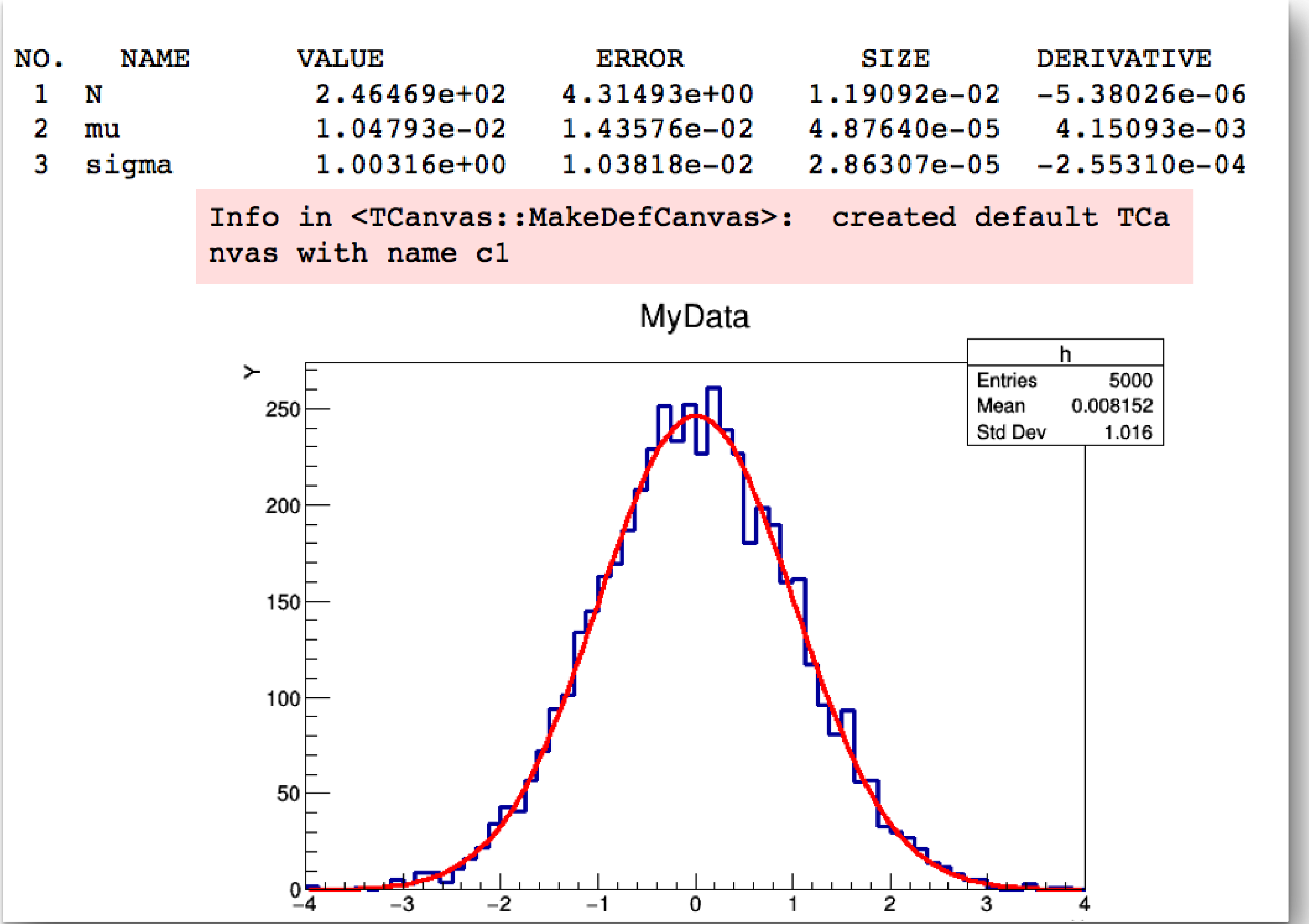
In [2]:

```
%%cpp -d
double myG(double* x, double* par){
  auto res = (x[0]-par[1])/par[2];
  auto e = -.5 * res * res;
  return par[0] * exp(e); // declare function
}
```

In [4]:

```
TF1 f("myGf",myG,-5,5,3);
f.SetParameters(200,0,1);
f.SetParNames("N","mu","sigma");
auto fr = h.Fit(&f,"S");
TCanvas c;
h.Draw(); c.Draw();
```

A C++ ROOTbook



```
NO.    NAME         VALUE            ERROR          SIZE      DERIVATIVE
  1  N            2.46469e+02   4.31493e+00   1.19092e-02  -5.38026e-06
  2  mu           1.04793e-02   1.43576e-02   4.87640e-05   4.15093e-03
  3  sigma        1.00316e+00   1.03818e-02   2.86307e-05  -2.55310e-04
```

```
Info in <TCanvas::MakeDefCanvas>:  created default TCa
nvas with name c1
```

Summary

Integration of ROOT & Jupyter Notebooks **delivered**

- Python flavour
  - import ROOT
  - %%cpp magic
- ROOT C++ Kernel
  - Thanks to Omar Zapata
- Goodies
  - Tab completion
  - Display of graphics
  - Syntax highlighting

Documentation, Links

- **ROOTbooks How-Tos**

https://root.cern.ch/howtos#Jupyter%20Notebooks

- **ROOT bindings for Jupyter**

https://github.com/root-mirror/root/tree/master/bindings/pyroot/JupyROOT

- **ROOT C++ Kernel**

https://github.com/ipython/ipython/wiki/IPython-kernels-for-other-languages

- **Examples** (15 already) from the *new* ROOT Tutorials can be found at:

https://root.cern.ch/code-examples#notebooks

both in Python and C++ (and mixed!)

Try It Out!

Install ROOT, install Jupyter and…

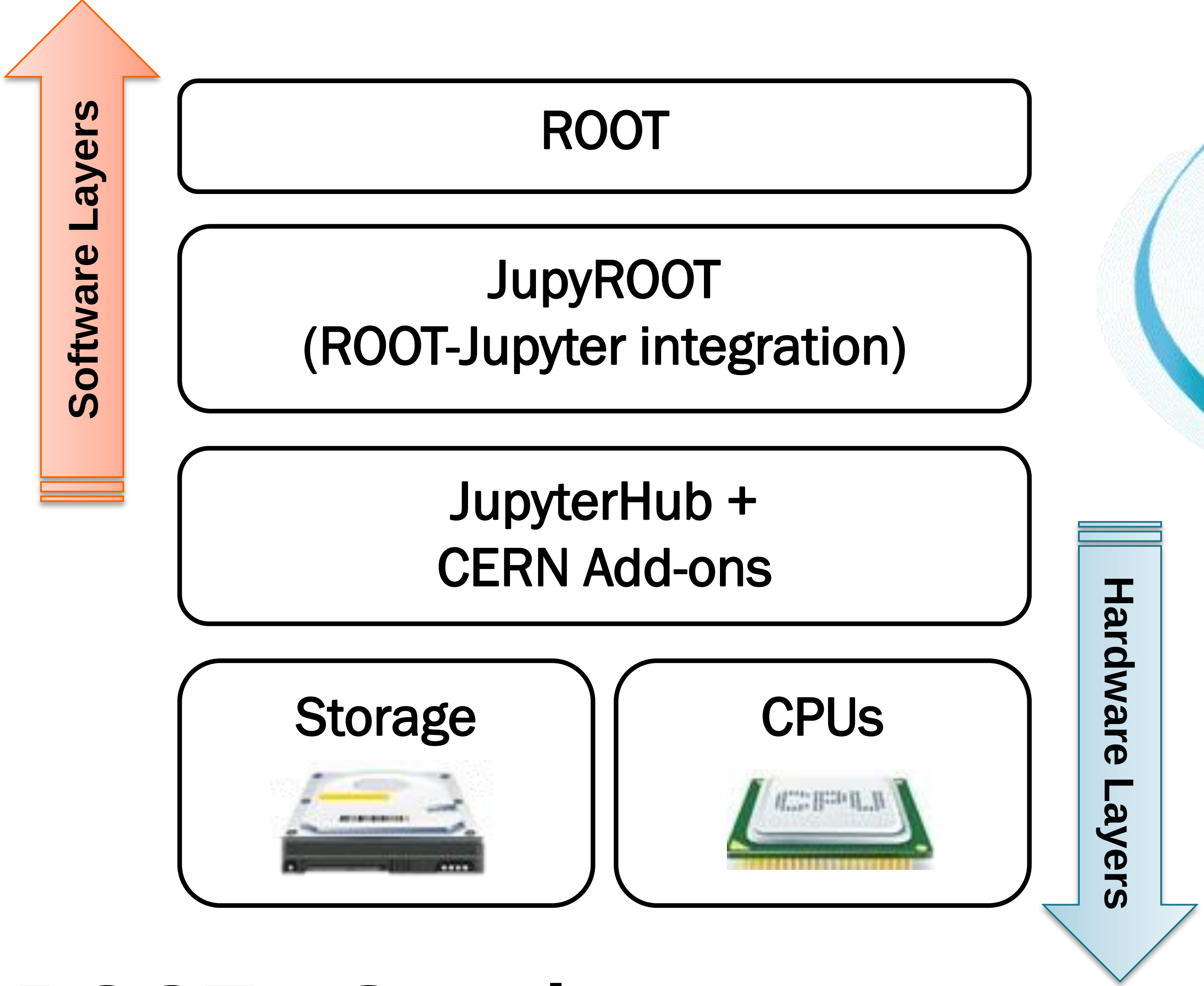
```
$ root --notebook
```

This command:

1. Starts a local notebook server
2. Connects to it via the browser

Provides a ROOT C++ kernel and the rest of ROOTbook goodies

ROOTaaS and CERN services' Portfolio



Software Layers

ROOT

JupyROOT
(ROOT-Jupyter integration)

JupyterHub +
CERN Add-ons

Storage

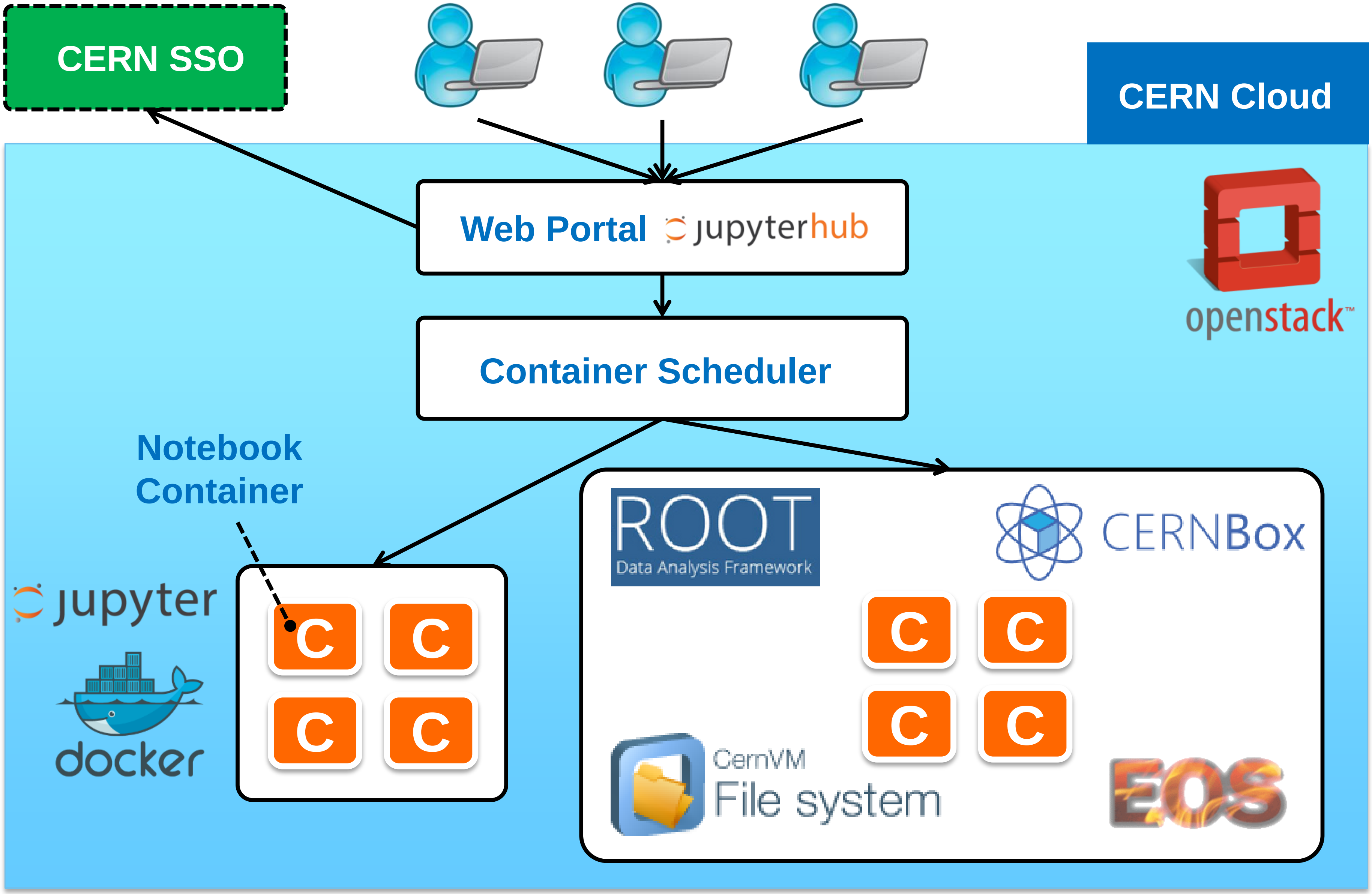
CPUs

Hardware Layers

Integration With CERN's Ecosystem

- Centrally provide ROOT as a Service
- Authentication with CERN credentials (*IT Sec*)
- Infrastructure: virtual machines in OpenStack Cloud (*IT-OIS*)
- Software distribution (*PH-SFT, IT-PES*): CVMFS
- Storage access (*IT-DSS*): CERNBox, EOS
  - All data potentially available!
- Synergy with document sharing (*IT-CIS*)
  - Notebook visualiser already available in Indico

Pilot Service



Conclusions, Plans

- ROOT is now integrated with notebooks
  - Python and C++ interactive shells
  - Tab completion, C++/Python integration, syntax highlighting, graphics inlining
  - Available now!
  - Future: leverage ROOT-R integration, …
- Integration with the CERN services portfolio
  - Collaborating with IT department
    - Special thanks to IT-DSS
  - Plan to deliver a Pilot Service in 2016