files without borders

exploring Internet-connected storage for research

Fabio Hernandez

fabio@in2p3.fr

IN2P3/CNRS computing center, Lyon , France

CCIN2P3

Saclay, December 2nd 2014

Preamble

This talk covers an ongoing exploratory work

Your feedback is very much appreciated

Part of this work was funded by the Institute of High Energy Physics (Beijing, China)

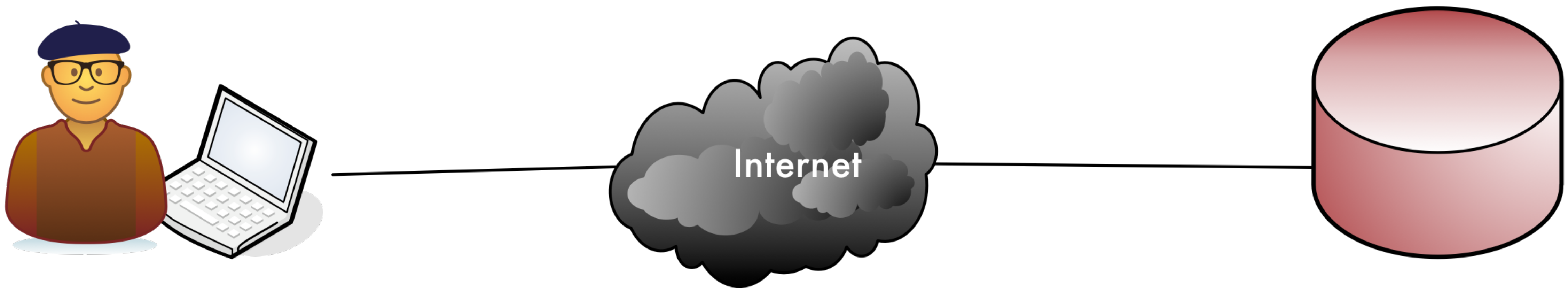
Motivation

- Can we collectively provide to IN2P3 staff the means for accessing their data transparently wherever they are connected?

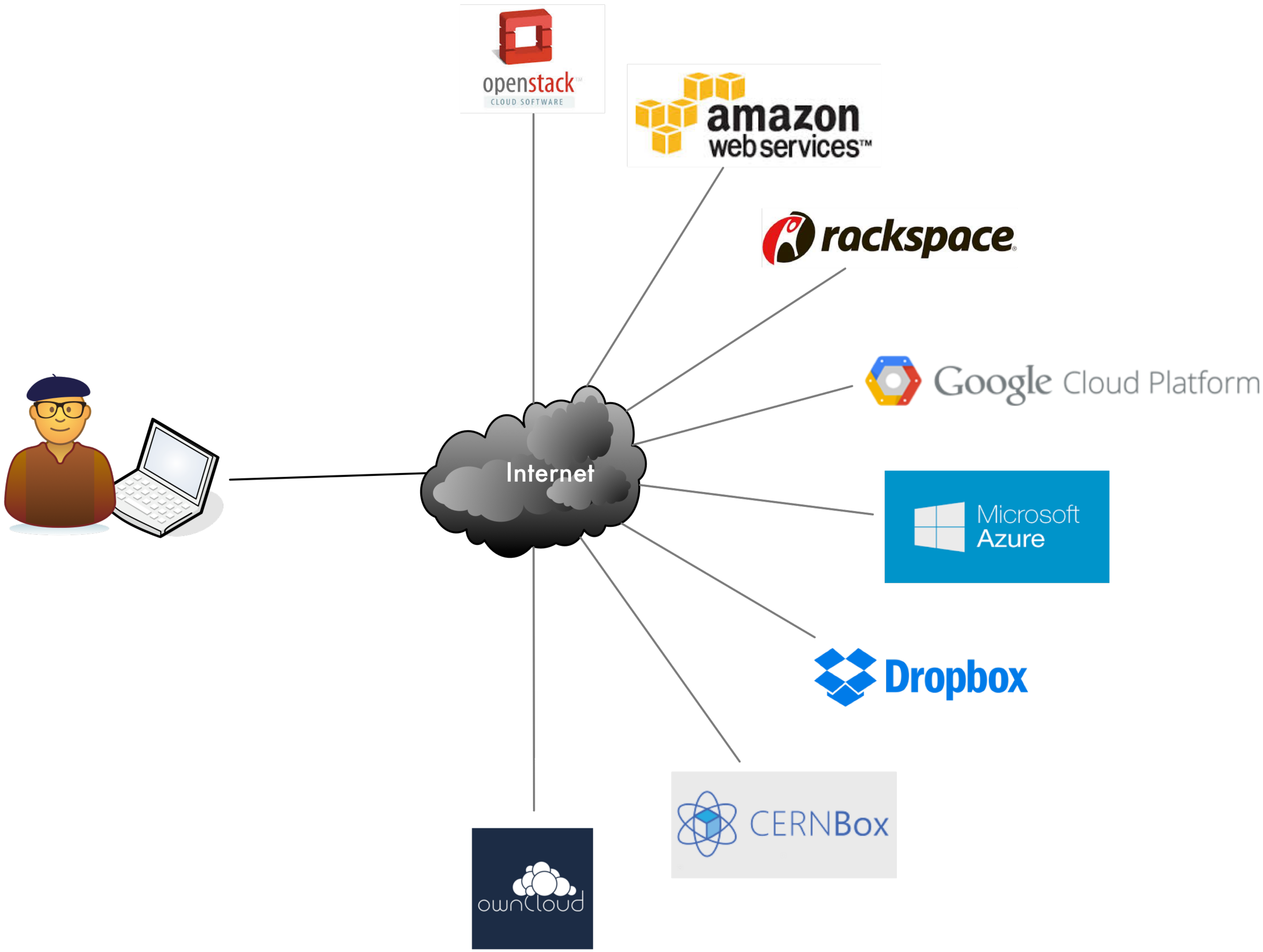
  no site-specific barriers, SSH connections, tunnelling, VPNs, …

- In other words, can we provide them an **Internet-connected personal storage device**?

Motivation (cont.)



Motivation (cont.)



OK, but why?

- I want to **access** my data from any of my connected devices
- I want to **easily share** selected data with my colleagues, in the next office or across the world
- I want to use convenient, **familiar tools** on my personal computer also for analysing my data

Lack or demand or lack of offer?

- This idea is neither new nor original. Still, we are not offering (nor getting) this kind of service yet
- So, what is missing?
- Would it add value to our users?

Ingredients

- Good network connectivity

  IN2P3 sites are very well interconnected

  enough bandwidth and low latency (< 10ms)

- Standard protocols and reliable storage backends

- Convenient client-side tools

  well integrated to the operating system of the personal computers

- Experience operating round-the-clock, storage-intensive services at significant scale

Let's try, then

- Goal of this work

  Explore how to implement Internet-connected storage in the context of scientific research

  Identify what use-cases this model is good for, if any

  ***Convenience first**, performance second*

"If you're not embarrassed by the first version of your product, you've launched too late."

Reid Hoffman, founder of LinkedIn

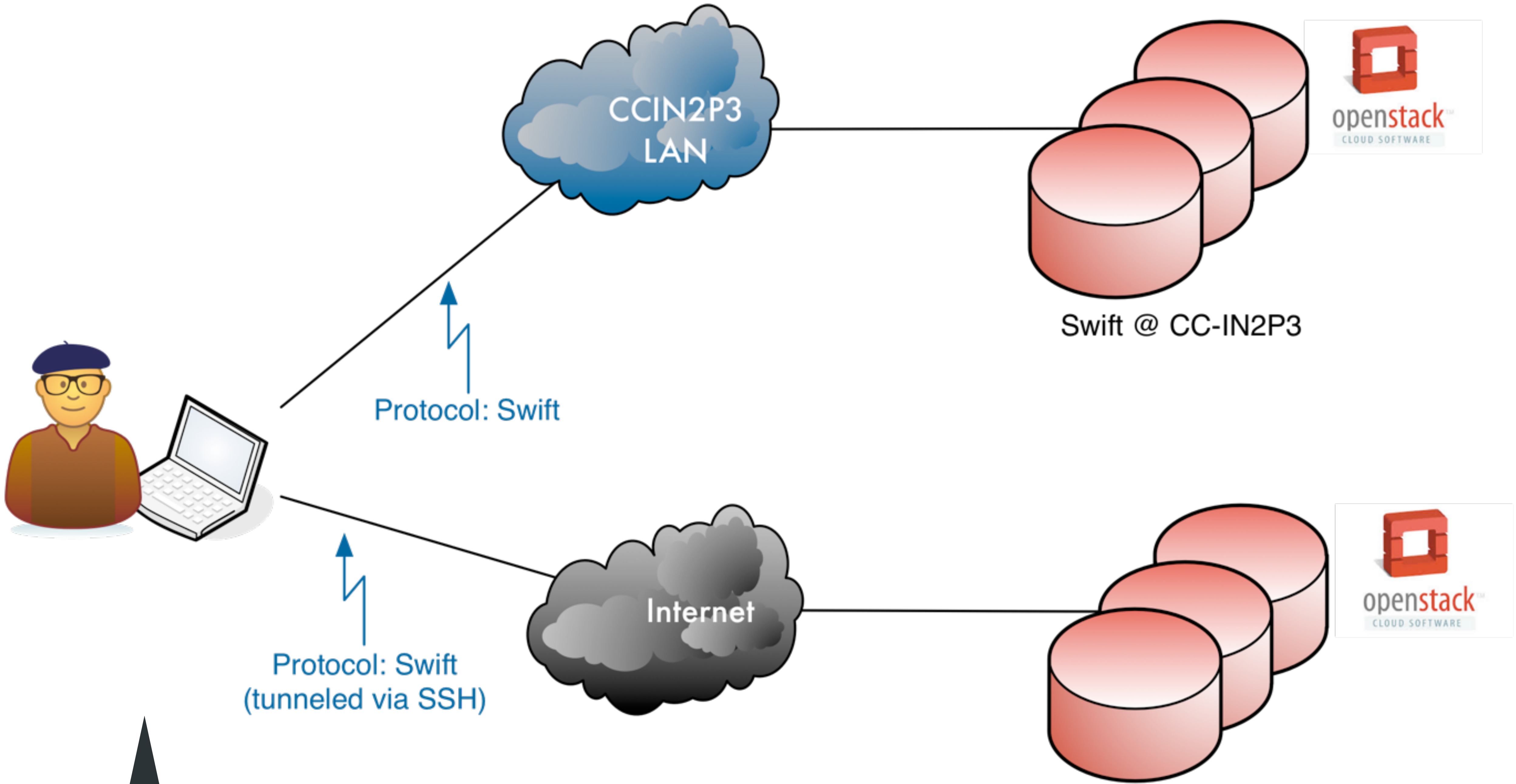
Outline

- Personal federation of remote storage
- Cloud storage basics
- Cloud storage & ROOT
- Perspectives
- Conclusion

Personal storage federation

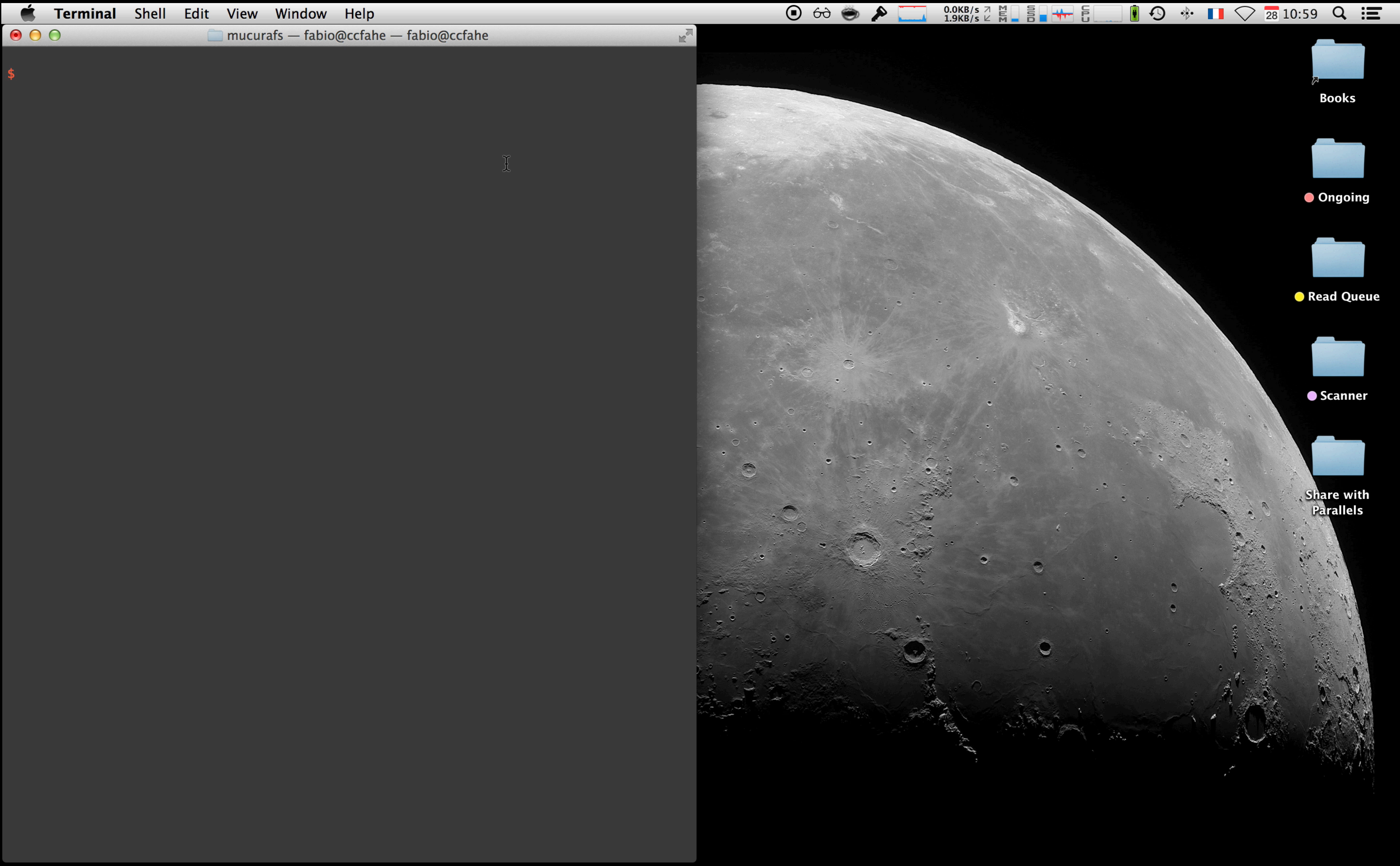
Demo 1

Demo environment



Synthetic file system mounted here

Federates several remote storage endpoints under same namespace



This video is available here

Personal storage federation

- Ongoing development work for implementing a **personal federation of remote storage** endpoints

  runs on your personal computer (currently Linux and MacOS X)

  initial target back-ends OpenStack Swift and Amazon S3

- Application-agnostic

  *applications **transparently read remote files** as if they were local files, except for latency*

  FUSE-based synthetic file system, emulates POSIX API

  *same software usable for **mounting cloud storage repositories on your personal computer** and for **(auto) mounting on worker nodes, virtual machines and Docker containers***

- Example real-life use case

  grid jobs running in Wuhan read BES III random trigger data (2GB, binary files) stored in Beijing (1150 Km)

  direct benefit: event reconstruction can be performed at compute-only remote sites

- Modern development environment

  Go programming language, designed with built-in concurrency, self-contained compiled executable

Cloud-based storage basics

Cloud-based storage

- Object storage system

  well documented programming interface

  *on top of **standard protocols** (HTTP)*

  accessible through wide area network

- Advantages for service providers

  elasticity, standard protocols, tuneable durability by redundancy, scalability, possibility of using commodity hardware, public or on-premise

- Typical use cases

  *well suited for "**write-once read-many**" type of data: images, videos, documents, static web sites, …, HEP data*

- Introduced in 2006, significant development over the last few years

  Amazon S3: 2 trillion objects, 1.1M requests/sec (as of April 2013)

  Microsoft Azure: 8.5 trillion objects, 0.9M requests/sec (as of July 2013)

  other big players: Google, Rackspace, Tencent, …

  open source implementations: OpenStack, Eucalyptus, …

Cloud storage model

- **Immutable** objects (i.e. files)

  file update is not supported; rewrite the whole file

  file versioning supported by some implementations

- Flat structure: no directories, only **containers** and **objects**

  objects are stored in containers (a.k.a. buckets) and uniquely identified

`https://fsc.ihep.ac.cn:8443/randomtrg/round05/120601/run_0028410_RandomTrg_file001_SFO-1.raw`

container name

object name

Cloud storage & ROOT

Cloud storage & ROOT

- Improved support for S3 protocol built-in from ROOT v5.34.05 (Feb. 2013)
- We developed an **extension** to ROOT which adds transparent support of cloud-based protocols

  no modification to ROOT source code nor to experiment's code is required

  currently supports both OpenStack Swift and Amazon S3

  tested against Amazon S3, Google Storage, Rackspace, OpenStack Swift, Huawei UDS

  backwards compatible *with legacy versions of ROOT: from v5.24 to v6*

- Features

  installable by ***unprivileged user*** *on a* ***private or shared ROOT installation***

  partial reads, web proxy handling, data caching, HTTP and HTTPS, connection reuse

  lightweight shared object library (500KB) + `TFile` *plugin*

Cloud storage & ROOT (cont.)

- Usage

*open cloud-based files for **reading** as if they were local*

```
TFile* f = TFile::Open("swift://myContainer/name/of/my/file.root")
```

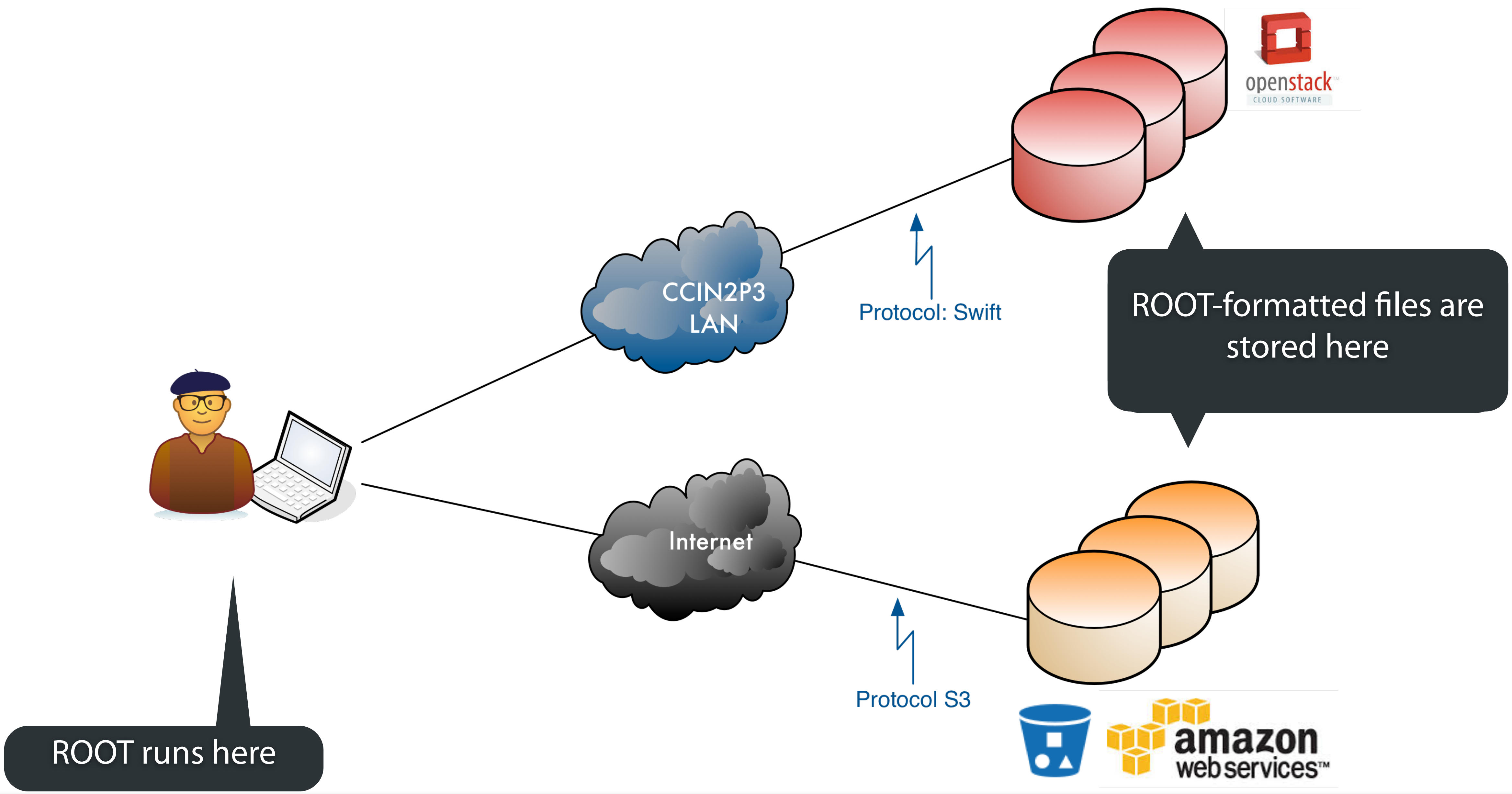
share URLs to their cloud files with other ROOT users

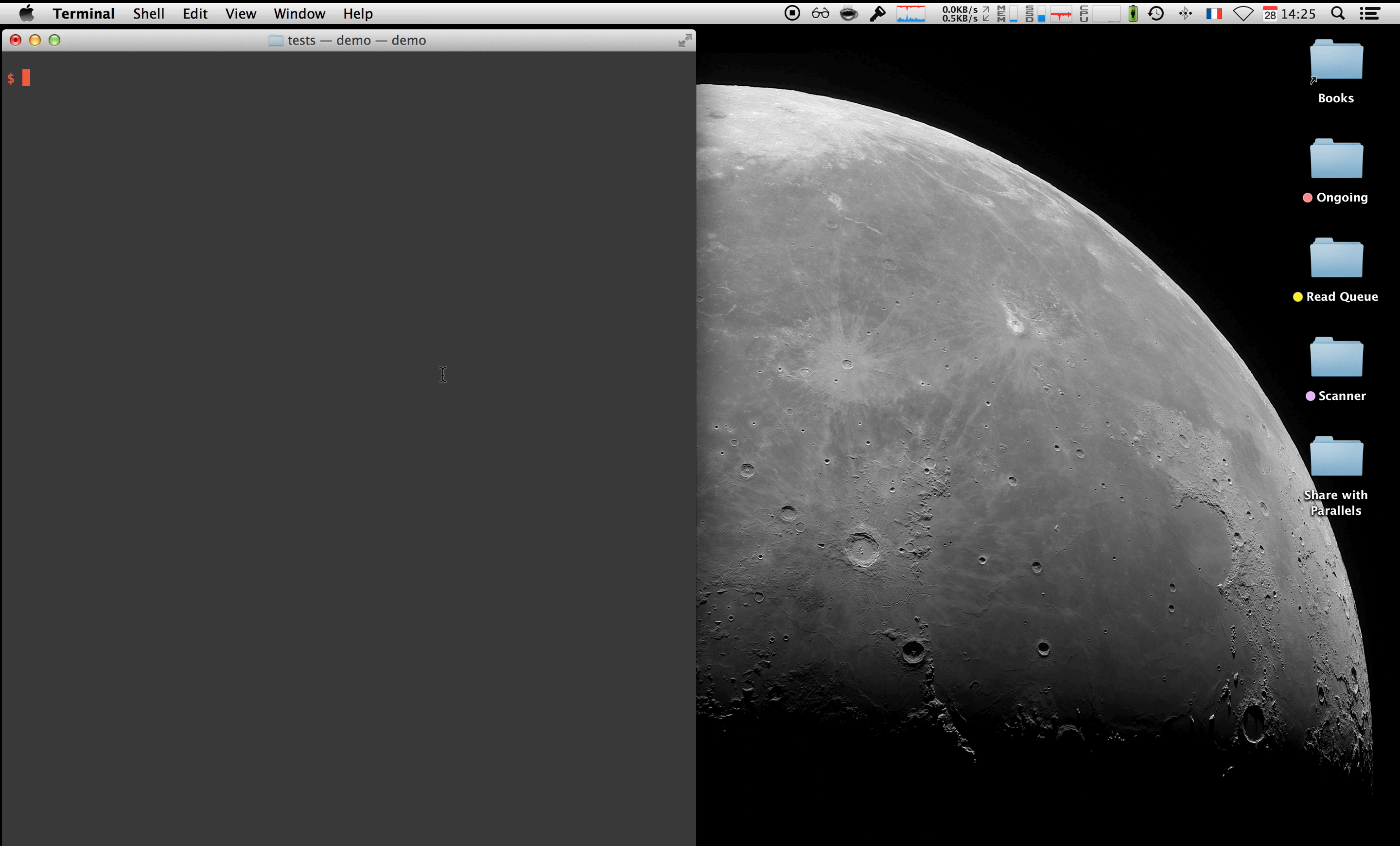
"Look at my plot at `s3://s3.amazonaws.com/myBucket/myHisto.root`*"*

Demo 2

Demo environment

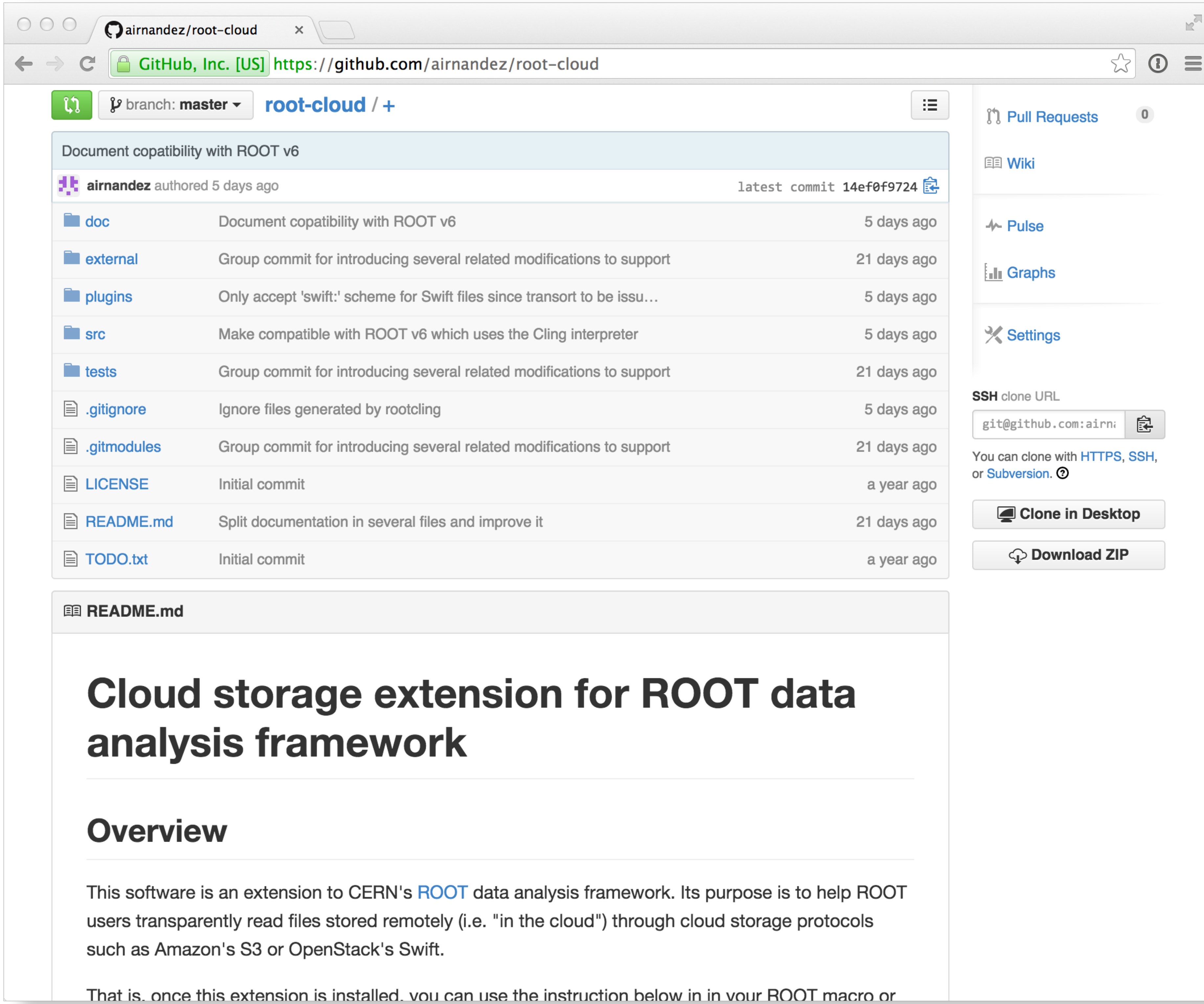
Goal: demonstrate usage of ROOT cloud extension for transparently reading remote files





This video is available here

Cloud storage & ROOT (cont.)



https://github.com/airnandez/root-cloud

Evaluation

- Quantify performance of cloud storage cluster in local area network

  performance with small-sized files

  efficiency of access protocol

  performance and scalability when used by real BES III jobs

- For full details, please refer to the paper

http://iopscience.iop.org/1742-6596/513/4/042050

20th International Conference on Computing in High Energy and Nuclear Physics (CHEP2013) IOP Publishing
Journal of Physics: Conference Series **513** (2014) 042050 doi:10.1088/1742-6596/513/4/042050

Integration of cloud-based storage in BES III computing environment

L Wang[1], F Hernandez[1,2] and Z Deng[3]

[1]IHEP computing centre, P.O. Box 918-7, 19B Yuquan Road, Beijing 100049, China

[2]IN2P3/CNRS computing centre, 43, bd du 11 Novembre 1918, 69622 Villeurbanne Cedex, France

[3]IHEP experimental physics centre, P.O. Box 918-1, 19B Yuquan Road, Beijing 100049, China

E-mail: Lu.Wang@ihep.ac.cn

Abstract. We present an on-going work that aims to evaluate the suitability of cloud-based storage as a supplement to the Lustre file system for storing experimental data for the BES III physics experiment and as a backend for storing files belonging to individual members of the collaboration. In particular, we discuss our findings regarding the support of cloud-based storage in the software stack of the experiment. We report on our development work that improves the support of CERN's ROOT data analysis framework and allows efficient remote access to data through several cloud storage protocols. We also present our efforts providing the experiment with efficient command line tools for navigating and interacting with cloud storage-based data repositories both from interactive sessions and grid jobs.

1. Introduction

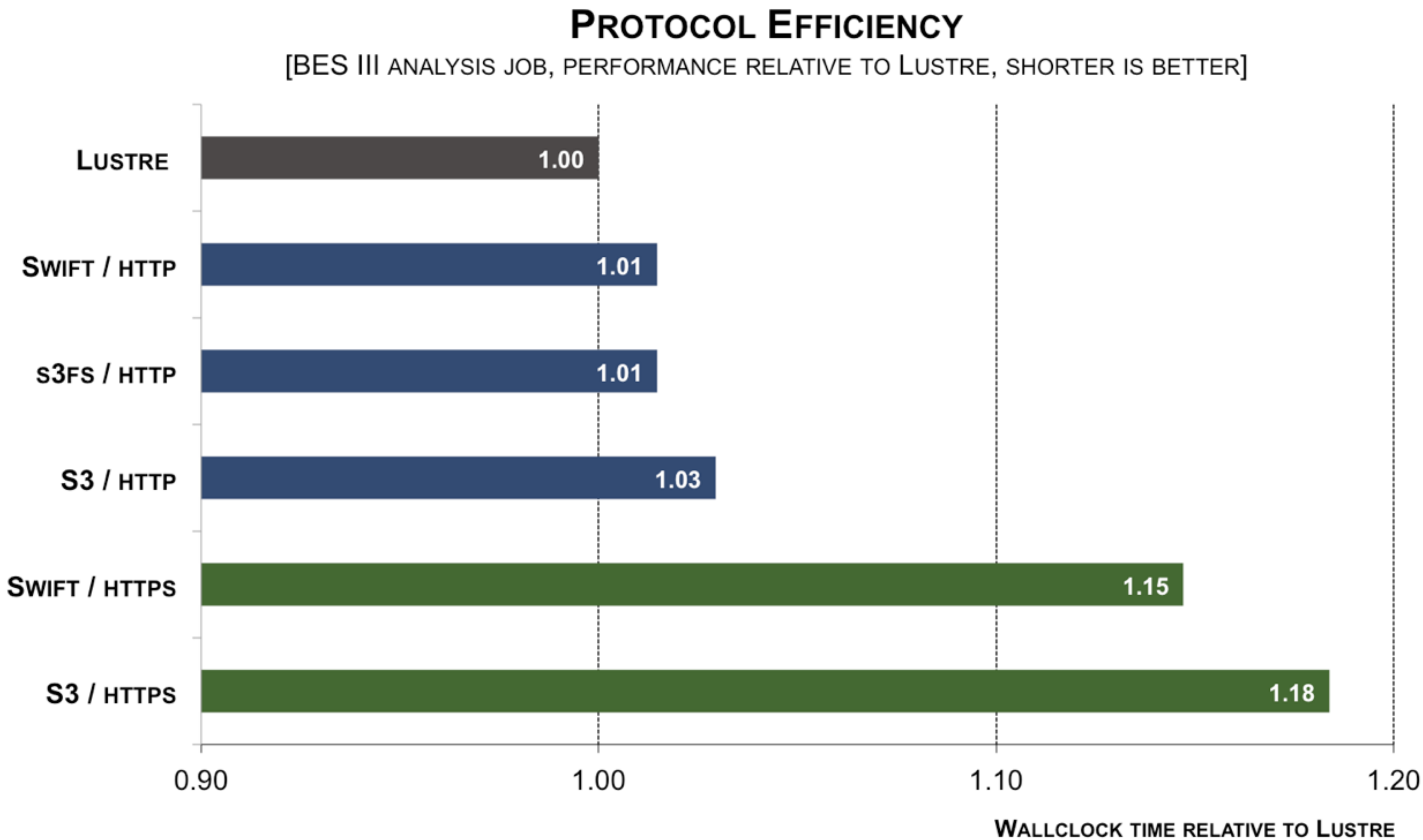
Object storage systems such as Amazon's Simple Storage Service (S3) [1] have substantially developed in the last few years. The scalability, durability and elasticity characteristics of those systems make them well suited for a range of use cases where data, in the form of files, is written, seldom updated and frequently read. Storage of images, static web sites and backup systems are examples of the use cases where remote object storage systems have proven effective [2]. In the rest of this paper we use the term *cloud storage* to refer to object storage systems that expose a well-documented interface on top of standard protocols such as HTTP so that remote clients can interact with the systems both over local or wide area networks.

Generally speaking, experimental physics data are stored as immutable files, which are read several times for the purposes of filtering and analysis according to the experiment-specific data processing workflows. This *write-once read-many* access pattern seems well suited for cloud storage systems.

We present in this paper an on-going work that aims to evaluate the suitability of cloud-based storage as a supplement to the Lustre file system [3] for storing experimental data for the BES III physics experiment [4] and as a backend for storing files belonging to individual members of the collaboration. For this evaluation, we deployed a test bed of OpenStack Swift [5], an open source, community-driven implementation of a cloud storage system used in production by several commercial cloud storage providers.

 1

Protocol efficiency with BES III jobs



Low overhead of both native Swift and S3 over HTTP
Noticeable penalty when using HTTPS

Perspectives

What's next

- Implement client-side caching mechanism

  for both metadata and data

  allows for disconnected operation

- Add write capabilities
- Explore client-side encryption
- Better integration with operating system

  e.g. certificate management, credential management

- Credential management for jobs
- Add support for other popular back-ends

Summary of features

- Synthetic file system conveniently exposes data as if it was locally stored

  uniform access to data from personal computer , worker nodes and virtual machines

  convenience first, performance second

- Federation of several distinct repositories into the same namespace

  each repository potentially speaking a different protocol

Potential use-cases

- Storage backend for personal files

  individual user files (software, analysis results, plots, papers, …)

  individual storage repository accessible not only on-site but also remotely through wide area network

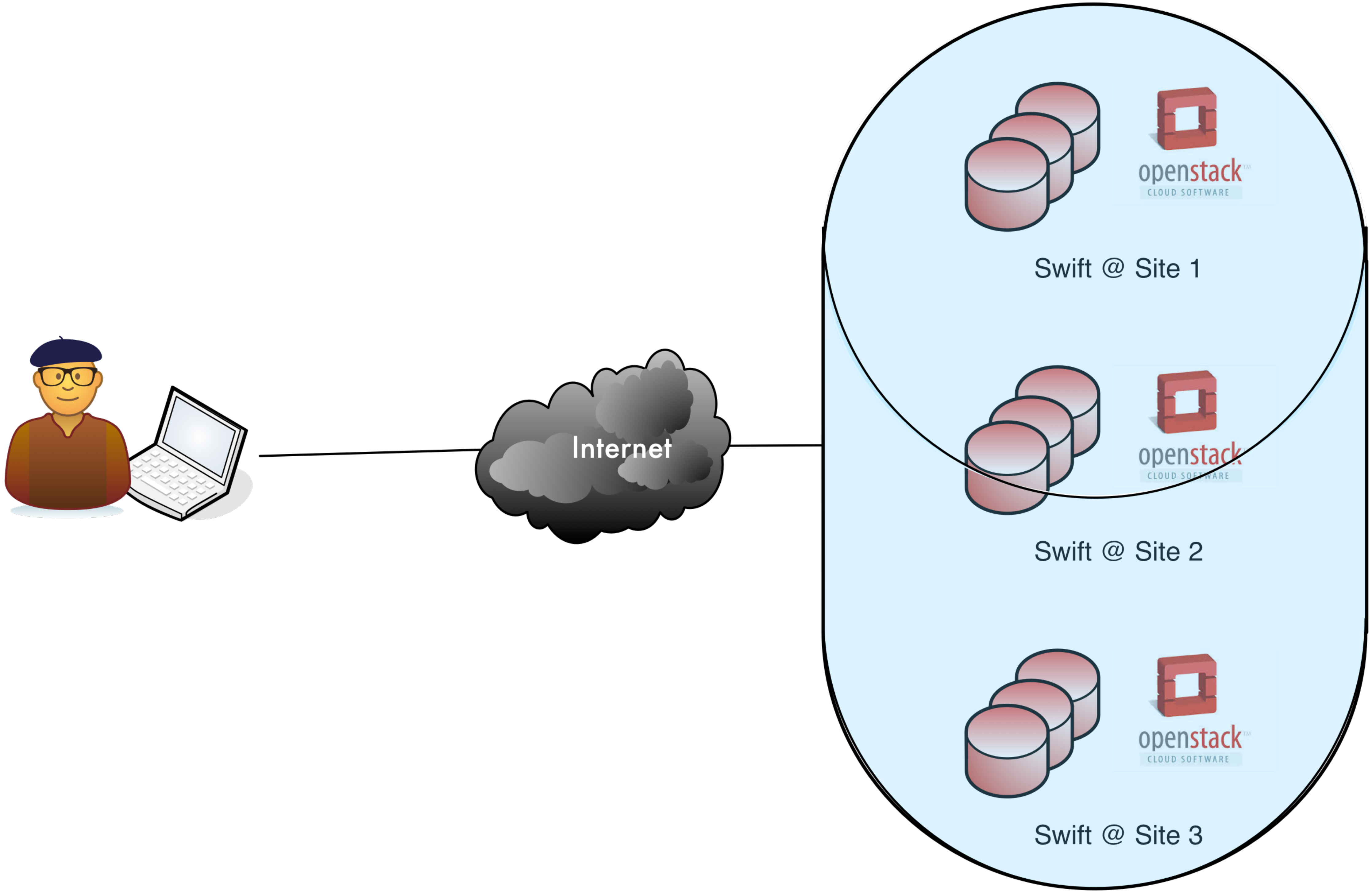
  uniform access methods from personal computer and from (grid) jobs

- Repository for sharing files among several individuals

  cloud storage acts as reference data repository

  accessible from anywhere, from any connected device

Potential use-cases (cont.)



Conclusions

- With a working prototype, demonstrated that Internet-connected storage and adequate client-side tools and add value to individual workflows

  still a lot of work remains, but preliminary results are encouraging

- Demonstrated that it is possible to integrate cloud storage backends into a running physics experiment's workflows, without disruption

  without modification to the experiment's software framework

  using real-world physics analysis jobs

Questions & Comments

References

- Part of this work was presented at the conference Computing in High Energy Physics (CHEP2013), Amsterdam, Oct. 2013

  Slides: *http://indico.cern.ch/conferenceTimeTable.py?confId=214784#20131014*

  Paper: *http://iopscience.iop.org/1742-6596/513/4/042050*

- Other presentations on the same subject

  https://speakerdeck.com/airnandez

Backup

Cloud storage vs. file system

	File system	Cloud storage
Storage unit	file	object
Container of data	directory	container (a.k.a bucket)
Name space hierarchy	multi-level `/dir1/dir2/.../dirn/file`	2 levels container(obj
File update	allowed	not allowed
Consistency	individual `write()` are atomic and immediately visible to all clients	updates eventually consistent
Access protocol	POSIX file protocol `file://dir1/dir2/dir3/file1`	cloud protocol over HTTP(S) `s3://hostname/bucket/object`
Command line interface	`cp`, `mkdir`, `rmdir`, `rm`, `ls`, ...	`s3curl.pl`, `s3cmd`, `swift`, `...`

OpenStack Swift testbed at IHEP

- Head Node x2

  10Gb Ethernet, 24GB RAM, 24 CPU cores

- Storage Node x4

  1 Gb Ethernet, 24GB RAM, 24 CPU cores

  3 x 2TB SATA disks

- Aggregated raw storage capacity: 24TB
- Max read throughput： 480MB/s
- Access protocols

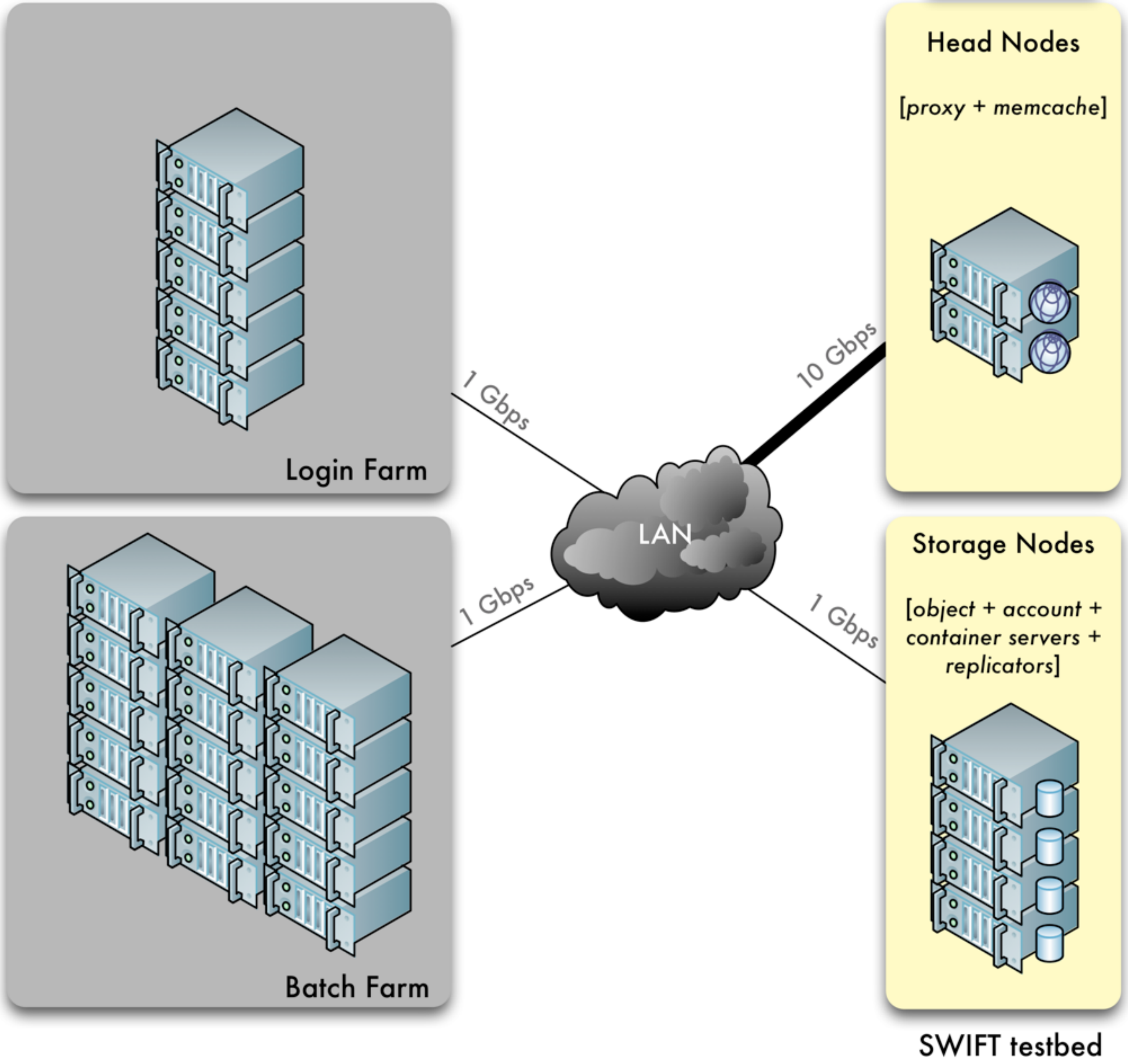
  native Swift

  Amazon S3 (partial support with 'swifts3' plugin)

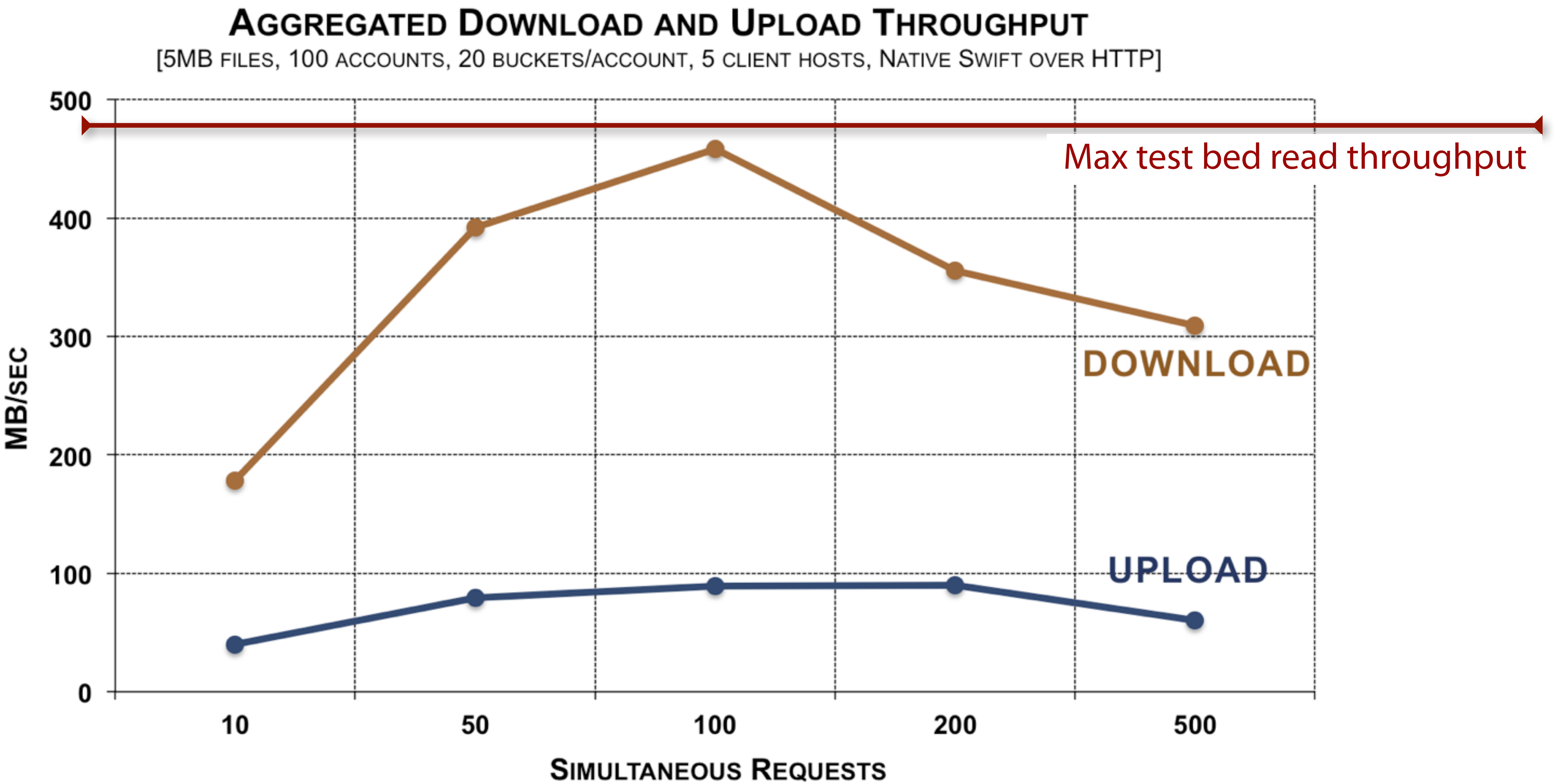
- Software

  OpenStack Swift v1.7.4

  Scientific Linux v6



Throughput with small-sized objects



Replication impacts write performance

Cloud storage extension for ROOT