

next_AIM kickoff meeting



17 Feb 2022, 09:00 → 18 Feb 2022, 13:30 Europe/Rome

ML_INF N project and INF N-Cloud

Daniele Spiga (INF N-PG)
On behalf of INF N-Cloud and ML_INF N teams

Outline



- Introduction to ML_INFNN and INFNN-Cloud projects: why they are together
- Supported workflow, access to accelerated hardware and interfaces
- Raising the bar: what's the benefit of having ML_INFNN on top of INFNN-Cloud... what could be the benefit for the users/communities (you)

Disclaimer: I'm not talking about Cloud technology at all, sysadmin details, configurations etc... if we are doing things correctly you shouldn't see "anything" about that (as a service)

- Any question is welcome

“Machine Learning at INFN” (ML_INF) project, is a INFN CSN5 funded project for the years 2020-2022, P.I. Tommaso Boccali

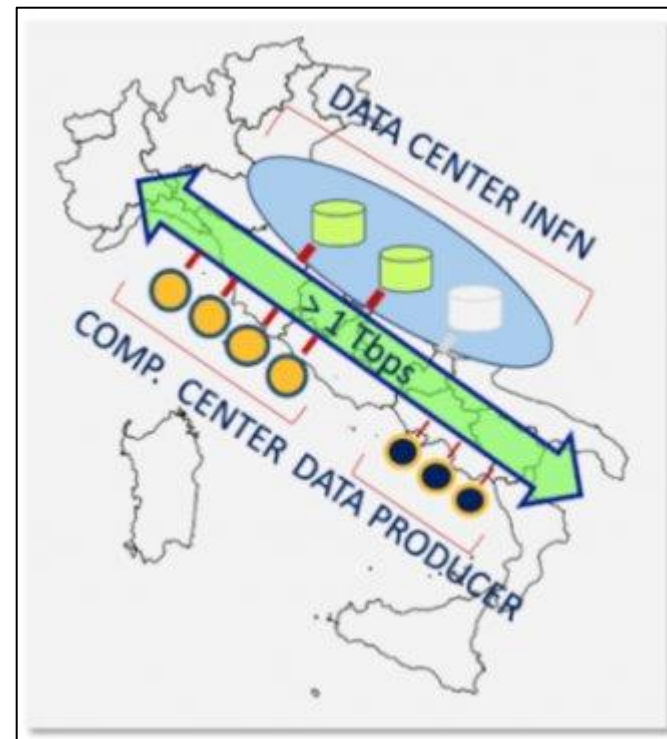
Three assets:

- Organize training/hackathons events addressing ML related theoretical aspects and use case specific implementation
- Create a knowledge base to collect/promote/share already existing studies in our domain (and in literature)
- Build and promote the adoption of performant and tailored technological platforms because a effective platform for Machine Learning prototyping and developing might represent a technical challenge

The rationale: researchers need **interactive environment**, the **access to hardware accelerators** (GPUs) and possibly a non-limiting **access to data** (i.e. training data). Handles to **create user tailored environment** is a key and finally groups need to **collaborate and share resources, data and code**.

Meanwhile... INFN-Cloud was born

- An **internal effort** at the INFN level in order to manage a (large) fraction of the INFN resources, in order to decouple user needs from the availability of local and dedicated hardware
- An attempt to rationalize the access to hardware, and optimize its use
 - From “1 GPU on each desk, used 5% of the time” to “shared resources optimally used”
 - It is the same direction we saw in the change “buy me 1000 dedicated computers” to “let’s build a GRID and use it with definite priority settings”
- A way to “equalize” INFN users in the access of resources, regardless from the (richness of the) experiment, the vicinity to a powerful computing centre, the capability to administer a complex resource such as those with GPUs etc



(some) Objectives of INFN-Cloud



To provide solutions for a wide range of user/community needs :

- a set of pre-developed distributed computing solutions, from the simplest (“I need a Linux PC for some uses, I do not want to buy one”) to open source composable components that allow INFN users to use, build and develop modern computing models and related resources.
- **A "high-level" mechanism for the adaptation and evolution of the service portfolio according to the needs and requests of users.**



- **Scientific Computing**
- **Development and R&D, testing of new services**
- **Training activities**
- Support to INFN data centers (for example for backups of services, etc)

The INFN-Cloud services

Virtual Machines (VM) possibly with external volume for storing data.

Docker containers

Pre-configured environment for **data analytics**

- Spark e/o ElasticSearch e Kibana, R, etc..

Storage solutions: Object storage / posix, possibly connected to high level application layers;

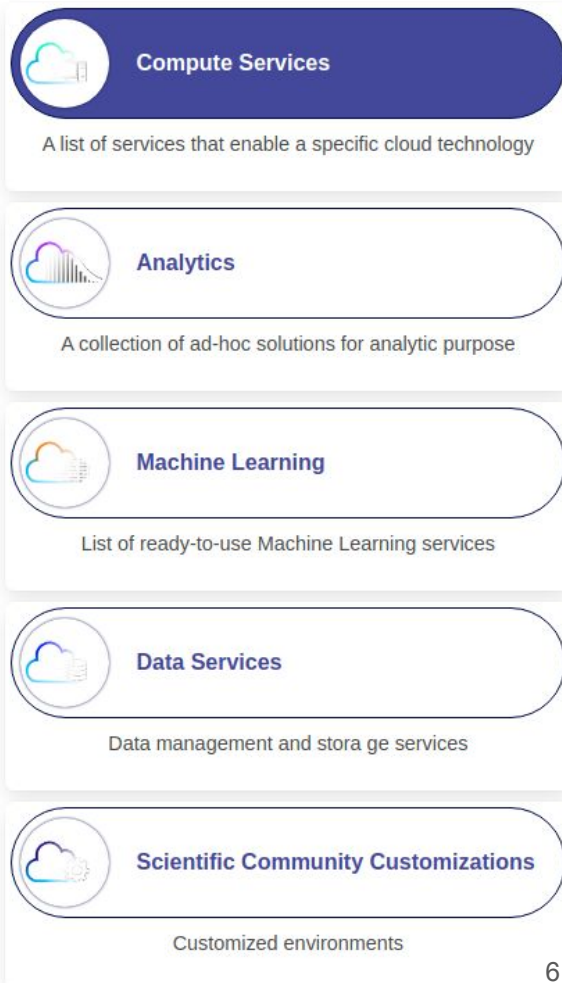
- Jupyter Notebooks with persistent storage (replicated)

Dynamic Clusters even designed and tuned taking into account the specific communities needs;

- HTCondor batch system; environment optimized for ML i.e. equipped with GPUs
- Container orchestrators such as K8s and Mesos

User-level disk encryption **to manage confidential data**

- Certified Cloud IEC/ISO 27001 at CNAF



ML_INFN && INFN-Cloud



ML_INFN chose INFN Cloud to build and to provide users with technological platform.
In particular

- A standardized multi user environment for research groups, based on the INDIGO-IAM Authentication and Authorization tool;
- Fast prototyping of solutions via Jupyter Notebooks and fully multi-user Jupyter Lab tools;
- Specifically procured hardware platforms, with the capability to host multiple accelerators, and equipped with fast NVMe disks to provide the needed bandwidth to data.

... and of course to provide toolkits (the lego bricks) to build and compose even highly customized workflows/pipelines

User perspectives Accessing the services in practice



INFN Cloud Dashboard Deployments Advanced External Links Users beta-testers Daniele Spiga

Search...

Virtual machine 	Docker-compose 	Run docker
Elasticsearch and Kibana 	Apache Mesos cluster 	Kubernetes cluster
Spark + Jupyter cluster 	HTCondor cluster 	RStudio
TensorFlow with Jupyter 	Jupyter with persistence for Notebooks 	Computational environment for Machine Learning INFN (ML-INFN)
Working Station for CYGNO experiment 	Sync&Share aaS 	

You can visit [here](#)
Getting started [here](#)

(if you are/once you get authorized)

Each button is a pre-developed service (self-service)

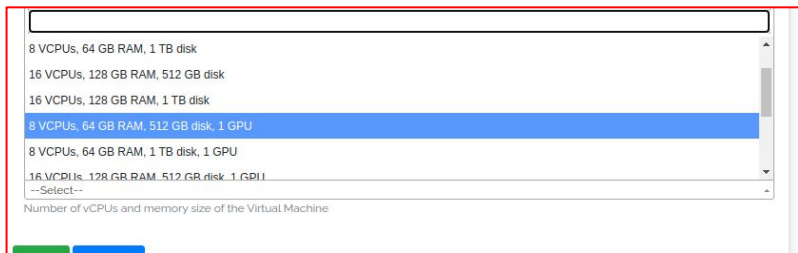
A project specific customization

The ML_INFN customized implementation



Simple high-level configuration template to create your personal environment

- Either for single user and multi users (group activities)
- Ask for CVMFS areas, GPUs, ...



Computational environment for Machine Learning INFN (ML_INFN)

Description: Run a single VM with exposing both ssh access and multiuser JupyterHub interface, integrating the ML-INFN environment

Deployment description

description

General IAM integration Advanced

jupyter_images

dodasts/ml-infn-lab.v1.0.0-snj

Default image for jupyter server

jupyter_use_gpu

True

Enable GPU utilization on jupyter

jupyterlab_collaborative

False

enable the jupyter collaborative service

jupyterlab_collaborative_use_gpu

False

enable the GPU on jupyter collaborative service

jupyterlab_collaborative_image

dodasts/ml-infn-jlab.v1.0.0-snj

Default image for jupyter collaborative service

cvmfs_repos

cms.cern.ch sft.cern.ch atlas.cern.ch

CMFS repositories to mount

ports

Add rule

Ports to open on the VM

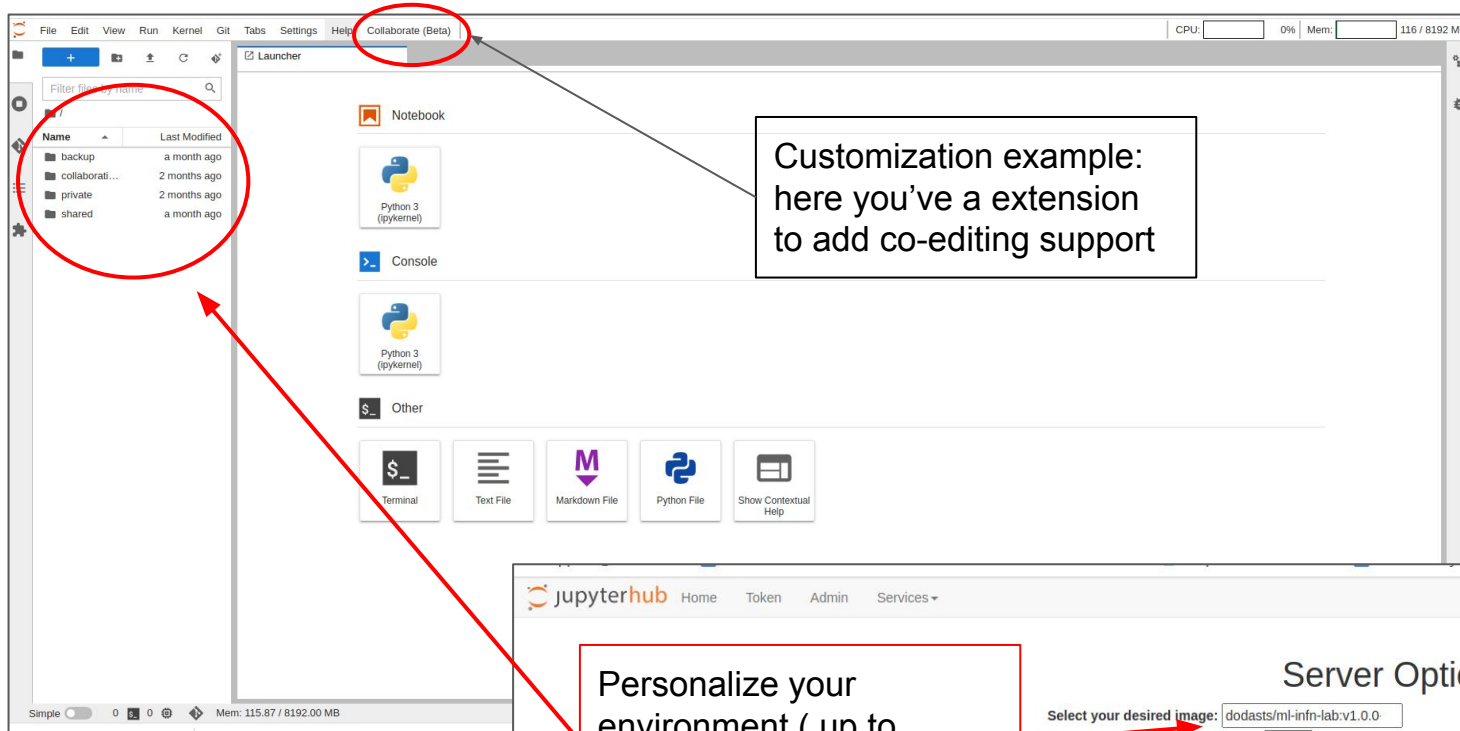
flavor

--Select--

Number of vCPUs and memory size of the Virtual Machine

Submit Cancel

Add here Jupyter view



The screenshot shows the JupyterLab interface. A red circle highlights the 'Collaborate (Beta)' menu item in the top bar. A red arrow points from this circle to a text box that says 'Customization example: here you've a extension to add co-editing support'. Another red arrow points from the same circle to a text box that says 'Personalize your environment (up to filesystem and storage access)'. A third red arrow points from the circle to a text box that says 'Everything starts here'.

File Edit View Run Kernel Git Tabs Settings Help Collaborate (Beta)

CPU: 0% Mem: 116 / 8192 MB

Launcher

Filter by file type

Name	Last Modified
backup	a month ago
collaborati...	2 months ago
private	2 months ago
shared	a month ago

Notebook

Python 3 (pykernel)

Console

Python 3 (pykernel)

Other

Terminal Text File Markdown File Python File Show Contextual Help

Simple 0 0 0 Mem: 115.87 / 8192.00 MB

Personalize your environment (up to filesystem and storage access)

Everything starts here

Server Options

Select your desired image:

Select your desired memory size:

Start

Operational models

There are two main patterns to operate these service and both follow the **as a service** paradigm

- **Single user deployment**

- The researcher (you) who is the “owner” of the resources/service is also the end-user of the service itself.
- Requirement is to be *nominated as system administrator* ([official doc here](#))

- **Multi tenants (i.e. per group) deployment**


- There is one (or a sub-group) person who is responsible for deploying and managing the resource/services
 - Requires the nomination as above
- A collaborator (or a group of..) can use the service (*no sysadmin. Approval required*)

NOTE: additional solutions can be discussed in order to relax the requirement to be “*nominated as system administrator*”, we can address this in details if needed

Data Access

This is a key point and mostly use-case specific (i.e. your data can be anywhere, no unique access pattern...) and thus we do not foresee a one size fits all solution

Several options are available to be exploited and possibly customized as per specific requirements

- **Using local ephemeral cache:** Your data keep staying where they are, a temporary copy can be made (“transparently”) close to the compute resource
- **Data can be pre-staged:** stored “manually” upfront (i.e. on a storage backend “auto-built”) and data can be read from there
 - I.e. a possibility can be to store data on Storage System of INFN-Cloud and process data from there i/e via pre-signed url
- **Posix-mounted remote endpoint** 
 - Beside the support infn-cloud might provide for this solution, this is surely nor a “cloud solution” neither a suggested one for performance purposes. Hower:
 - A suitable approach for a “global data sharing” (access my files anywhere, anytime)
 - Already adopted elsewhere@INFN-Cloud: i.e to grant persistency of notebooks, configurations etc..

Raising the bar...

Today we discussed the ML_INFNC customized implementation which has, so far, a primary focus on supporting the interactive data processing

If one would like to “raise the bar” moving towards a **quasi interactive** up to a **batch like processing** should “just” push on another button

- **Spark based processing** (via jupyter interface) already available ✓
- **HTCondor batch on demand**, already available ✓
 - This is already integrated also with Jupyter Interface but also accessible i.e. from your laptop.

—> based on the composable approach promoted by INFN-Cloud one can **propose and co-design a dedicated solution**