SWAN for Machine Learning

Enric Tejedor for the SWAN team

https://cern.ch/swan

CERN IT ML Infrastructure Workshop

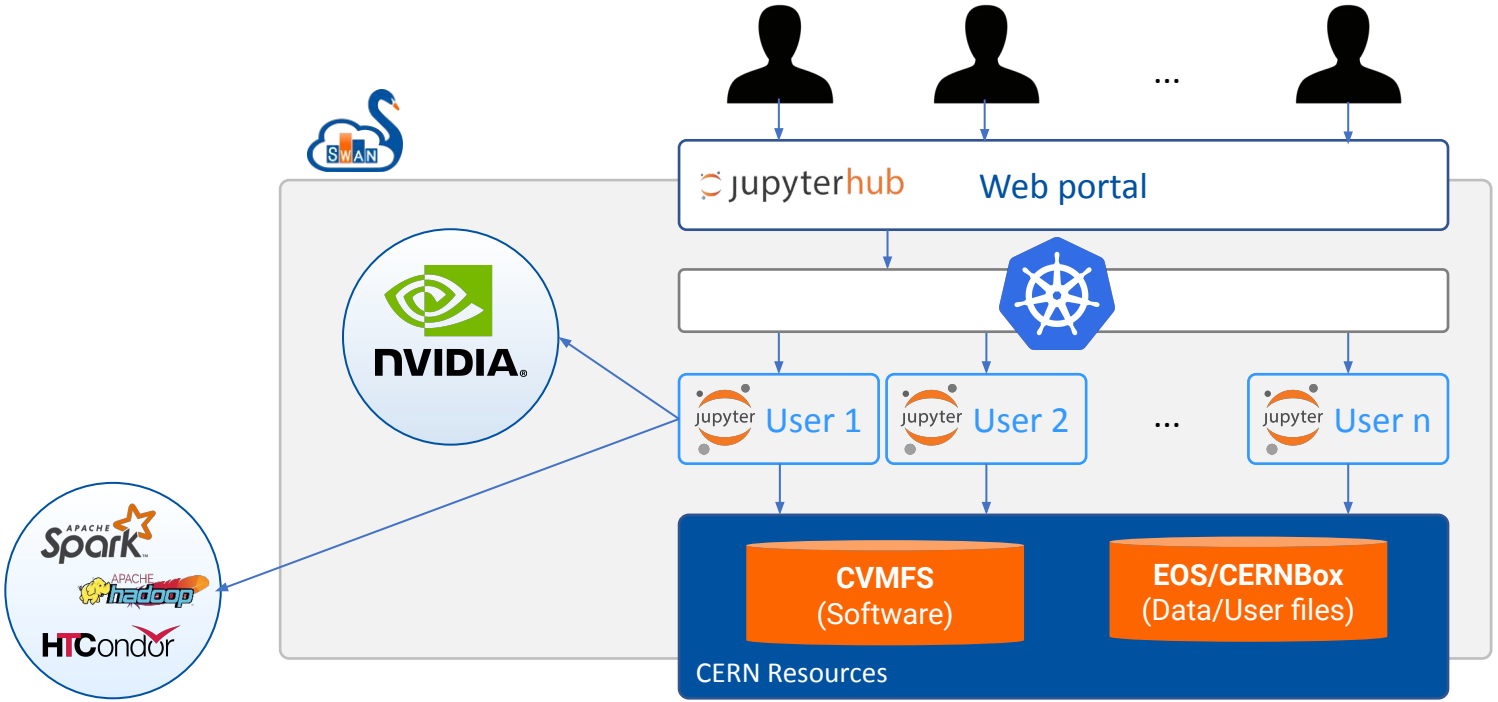
May 10th, 2023

CERN

SWAN in a nutshell

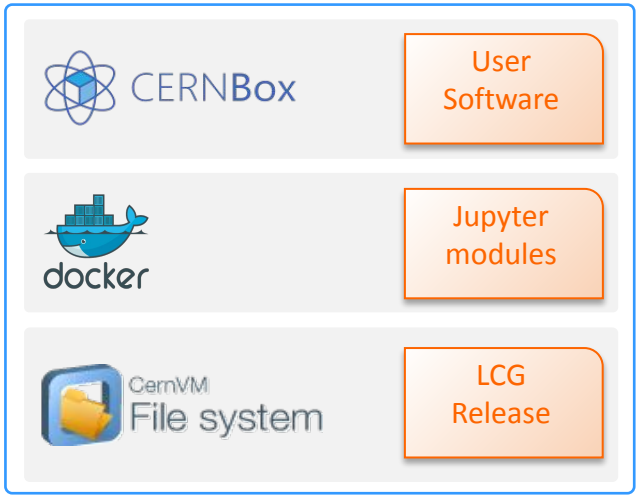
- Service for web-based interactive analysis
  - No local installation needed
  - Calculations, input data and results “in the Cloud”
  - Jupyter notebooks, terminal, file browser
- Good for data analysis and exploration, and also teaching
- Easy sharing of scientific results: plots, data, code
- Integration with CERN resources → added value!
  - Software (CVMFS)
  - Storage (EOS, CERNBox)
  - Computing (**GPU**, Spark, HTCondor)

SWAN architecture (k8s)



ML software on CVMFS

- Software provisioning for ML applications via CVMFS
  - **LCG CUDA** stacks with GPU-enabled software for ML
  - Complemented with EOS for custom software environments



→ custom user env (optional)

→ thin layer (not user defined)

→ main software source

TensorFlow Keras NVIDIA CUDA PyTorch ONNX RUNTIME scikit learn CERN

GPUs for interactive analysis

- SWAN allows to attach a GPU to a user session
- The GPUs are used interactively
  - When starting their session, the user selects a CUDA software stack and gets a GPU
  - GPU-enabled packages (e.g. tensorflow, PyTorch) can then be used in a notebook and offload to the GPU by default

```
In [1]: import tensorflow as tf

        tf.debugging.set_log_device_placement(True)

        # Create some tensors
        a = tf.constant([[1.0, 2.0, 3.0], [4.0, 5.0, 6.0]])
        b = tf.constant([[1.0, 2.0], [3.0, 4.0], [5.0, 6.0]])
        c = tf.matmul(a, b)

        Executing op MatMul in device /job:localhost/replica:0/task:0/device:GPU:0
```

Current GPU resources and users in SWAN

- 12 GPUs now available in production
  - Assigned via the SWAN Openstack project
  - Model: Tesla T4
  - Daily peak of 12 GPUs used, already at capacity!
  - Access controlled by e-group membership (to be reviewed)
- Increasing interest from user community
  - 2-3 new requests per week

I'm working on the Patatrack project

I would use GPUs for CUDA implemented training used for my diploma thesis

GPU access is needed for R&D of ML method for ATLAS analysis

I use graph neural networks to understand its impact on the estimation of the pile-up distribution in the CMS experiment

I would like to occasionally use GPU session for training my DL models at DUNE

I am a PhD working in the ASACUSA experiment in AD. I am trying to implement and train a NN for reconstruction improvement of our detector

The purpose is particle tracking for FCC-ee high energy booster

I would like to have access to GPU on SWAN to train Neural Networks. This would considerably speed up my work

SWAN for teaching with GPUs

- Frequent requests from courses / workshops to use SWAN
  - Notebooks are a good tool for teaching
- Some of them ask for GPUs
  - “I need XX GPUs to use SWAN during a course about ML”
  - Three examples already in 2023: ATLAS ML workshop, iCSC and Italian Teacher Programme
- Current provisioning model is a manual expand + shrink for every event
  - Draining of machines, increase of quota, addition and configuration of new nodes in SWAN k8s cluster – then revert all that
  - Sharing of GPUs would help here – e.g. MIG technology would allow partitioning to increase GPU numbers

Possible bridging to other ML services

- Lxbatch
  - Idea: do the interactive and exploratory phase in SWAN, then move to batch for longer runs
  - “Run me on batch” button in a SWAN notebook, e.g. to train on a bigger GPU or with a larger dataset
  - Condor packages already available in the LCG releases on CVMFS
- Spark
  - Idea: use SWAN as entry point for launching Spark ML workflows with GPUs
  - Would benefit from UI for connecting to Spark clusters and monitoring jobs
  - Spark jobs for data preparation and ML on CPU are already possible
- Kubeflow
  - Idea: submit ML workflows / run inference with Kubeflow from SWAN
  - Kubeflow Pipelines package (kfp) already available in the LCG releases on CVMFS
  - Possibility to exchange OAuth tokens for Kubeflow in SWAN

Final remarks

- SWAN offers a friendly interface for interactive usage of GPUs
  - For ML (and other use cases!)
  - Focus on your application: preparation of data, exploration, trial and error, tuning
  - Needed software is already provided – if not, it can be added
  - Data persistency guaranteed by EOS, sharing of results via CERNBox
- For long ML computations, it can help users connect to other ML services / computing resources at CERN
- SWAN would benefit from a more flexible provisioning model to handle spikes in user load
  - Shared pool of GPUs? How are they obtained?
  - Hybrid model (statically + dynamically assigned GPUs per service)?
  - Sharing at the GPU level (i.e. physical partitioning)?
  - Offloading to the Cloud?