Locally Managed Galaxy Instances with Access to External Resources in a Supercomputing Environment

Nuria Lozano[1,2], Oscar Lozano[2], Beatriz Jorrin[1], Juan Imperial[3], Vicente Martin[2]

[1] Center for Biotechnology and Genomics of Plants (CBGP), Technical University of Madrid
[2] Madrid Supercomputing and Visualization Center (CeSViMa), Technical University of Madrid
[3] Center for Biotechnology and Genomics of Plants (CBGP), Technical University of Madrid and CSIC

1. Introduction

Galaxy is a tool that can use resources well beyond those available to a research group or even to a research center, especially when huge datasets are going to be used. In this case the access to shared resources, like those available in supercomputers centers is a welcome addition to Galaxy capabilities. However, privacy or flexibility requirements might impose the need for a locally managed Galaxy installation, something not possible in a typical, batch-mode oriented facility such as a supercomputer. In these cases a way to communicate a local instance of Galaxy, managed by the research lab, with the supercomputer such that all the heavy work can be offloaded would be a solution.

Here we present the solution adopted by the Technical University of Madrid (UPM). UPM manages the Research Campus of Montegancedo, where the Center for Plant Genomics and the Madrid Supercomputing and Visualization Center (CeSViMa) are located.

2. Cluster: Magerit

CeSViMa manages a large heterogeneous cluster, called Magerit, of about 4,000 Power7 cores and 1,000 Intel ones. Some of the nodes have special characteristics, like a large memory footprint or accelerators (either Xeon Phi or Nvidia K20x).

As it is standard in these installations, Magerit resources are accessed in batch mode. The resource manager used is SLURM and the scheduler is MOAB. The standard way to run a job in Magerit involves to log into one of the interactive nodes, prepare a job command file and then submit it to one of the batch queues of the machine.

The challenge was to be able to seamlessly use this system through a Galaxy front-end.

3. Solution: Collaboration VPS – Cluster

- Production local instance installed in a Virtual Private Server.
- Jobs are sent to a Cluster (Magerit) and executed there.
- A shared filesystem between the application server (VPS) and the cluster nodes is required.
- The path to Galaxy must be exactly the same on both the nodes and the application server. It is the one required by Magerit.
- *Galaxy_user* must have the same name and IDs on both. It is a real and unique Magerit user.
- Some kind of communication is needed between VPS and Magerit → Command-line/shell interface (CLI) runner chosen:
  - Job plug-in specific for SLURM developed
  - A jobfile is created by VPS in the shared filesystem and sent to Magerit queue using ssh.
  - Galaxy process needs to know each job state.
  - RSA key used in order to avoid typing SSH password.
- Data and results are stored in the shared filesystem.
- PostgreSQl Database added. Located outside VPS filesystem.
- Apache Proxy to serve web page to the internet.

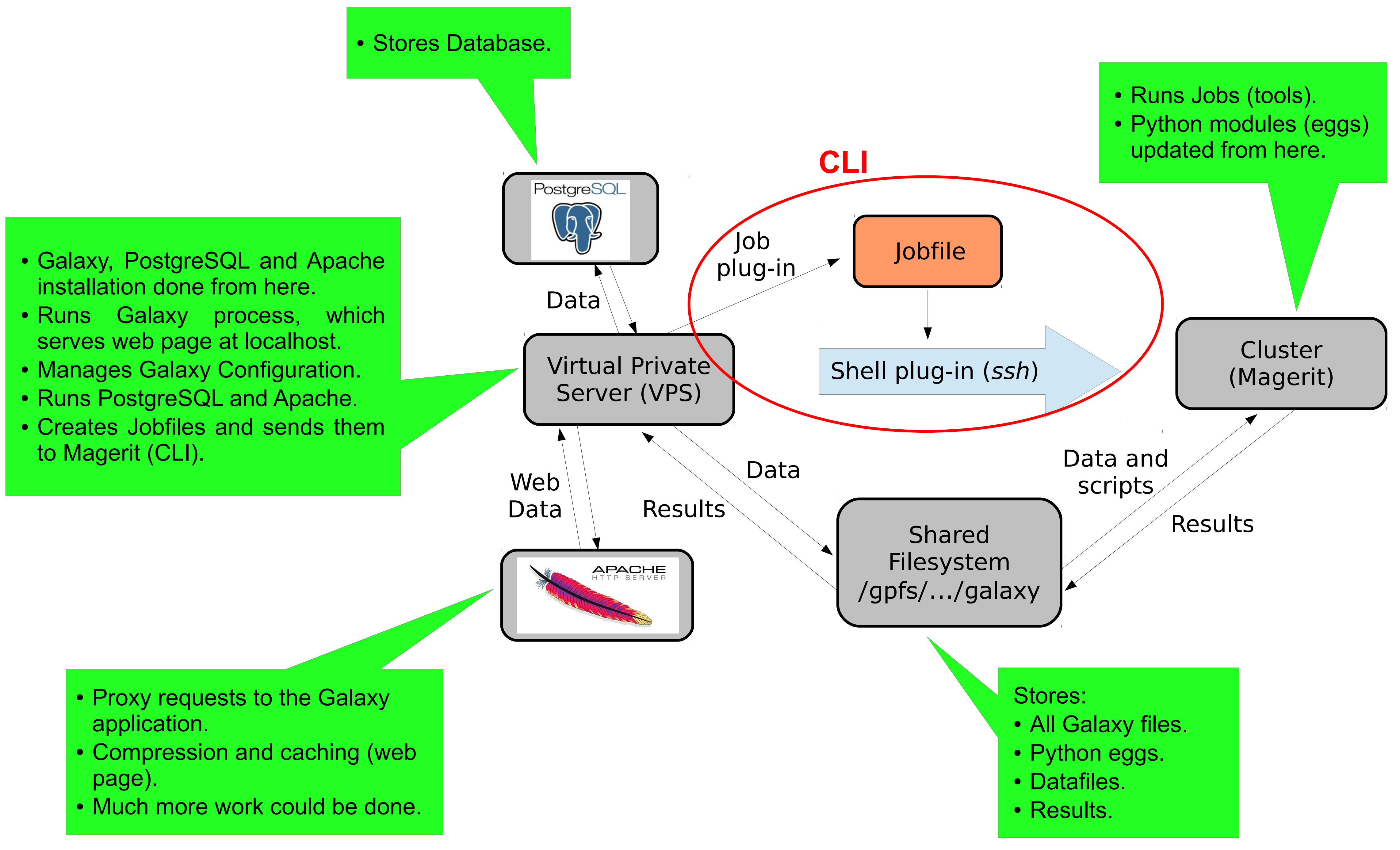
• Stores Database.
• Galaxy, PostgreSQL and Apache installation done from here.
• Runs Galaxy process, which serves web page at localhost.
• Manages Galaxy Configuration.
• Runs PostgreSQL and Apache.
• Creates Jobfiles and sends them to Magerit (CLI).
• Runs Jobs (tools).
• Python modules (eggs) updated from here.
PostgreSQL
CLI
Job plug-in
Jobfile
Data
Virtual Private Server (VPS)
Shell plug-in (*ssh*)
Cluster (Magerit)
Web Data
Results
Data
Data and scripts
Results
Shared Filesystem /gpfs/…/galaxy
APACHE
• Proxy requests to the Galaxy application.
• Compression and caching (web page).
• Much more work could be done.
Stores:
• All Galaxy files.
• Python eggs.
• Datafiles.
• Results.

Fig. 1 Collaboration VPS - Cluster

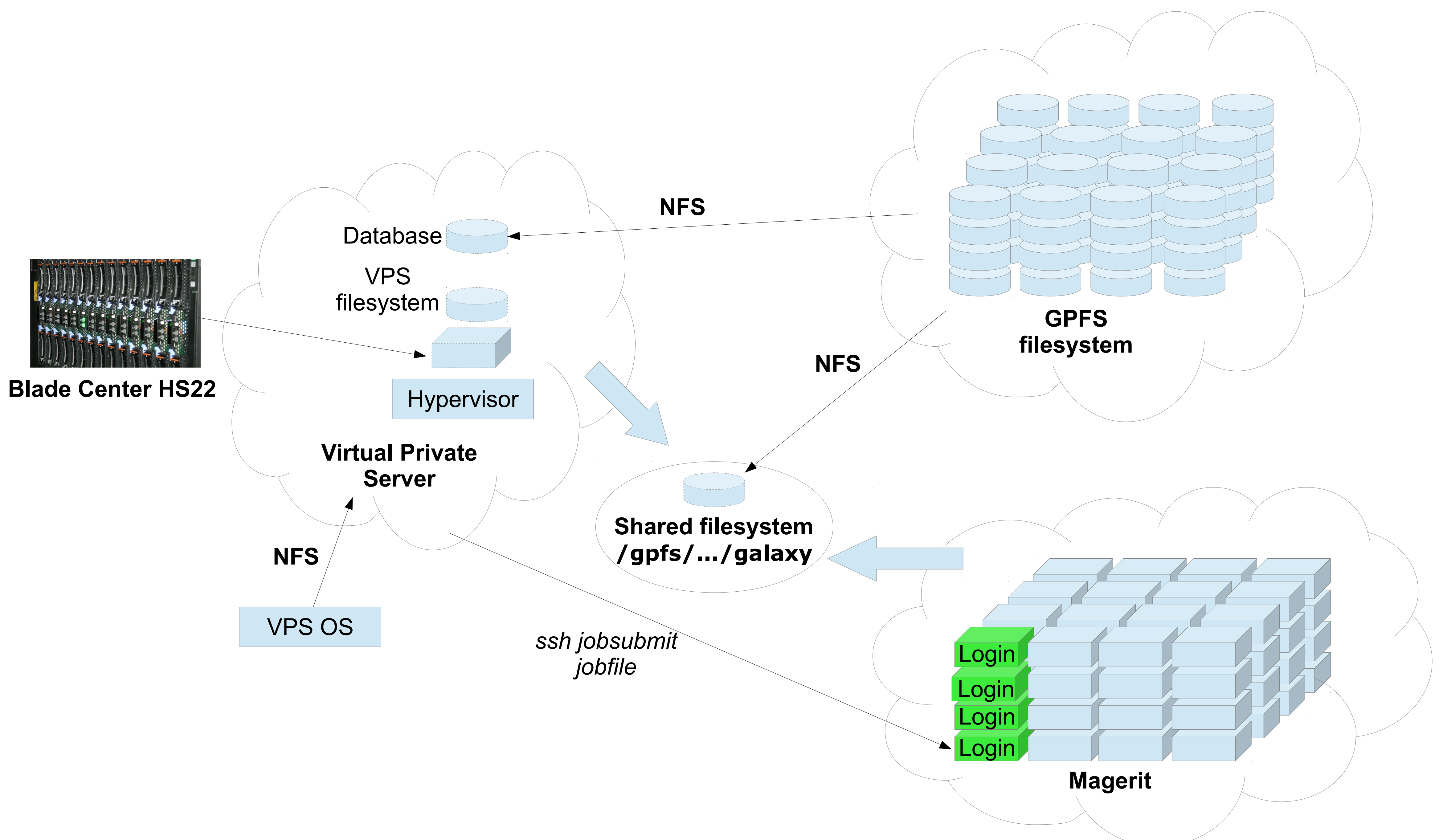

Fig. 2 Physical implementation abstraction

4. Configuration to send Jobs

Command-line/shell interface plugins

- Shell plug-in. Provided by Galaxy. Uses SSH to connect to Magerit as the assigned user.
- Job plug-in. Developed slurm.py. Creates Jobfiles adapted to Magerit requirements and interacts with Slurm.
  Location: `galaxy-dist/lib/galaxy/jobs/runners/cli_job/`

Job configuration file

- CLI and local job runners defined.
- Default tools destination uses CLI runner as explained above.
- UCSC tools, which require external communication, are mapped to local destination, using therefore local runner.

5. Production Configuration

Apache server

- Install Apache on VPS.
- Use mod_rewrite and mod_proxy.
- Redirect to `localhost:port`.

PostgreSQL

- Install PostgreSQL on VPS.
- Initialize Database on the external storage path.
- Edit *universe_wsgi.ini*:
  `database_connection = postgres://<dbuser>:<password>@localhost:5432/<dbname>`

Galaxy users

- Users must be logged (no anonymous users allowed).
- In addition, users can only be created by Admin. → More security.

6. Conclusions.

By means of this approach, research group members are fully responsible of deploying and maintaining their own Galaxy Local Instance, while the heavy work is offloaded to external computing resources, in this case to CeSViMa (Virtual Server, HPC, Storage).

7. Acknowledgements

The author thankfully acknowledges the computer resources, technical expertise and assistance provided by the Supercomputing and Visualization Center of Madrid (CeSViMa).