Centre de Calcul de l'Institut National de Physique
Nucléaire et de Physique des Particules
(CC-IN2P3)

Web Administration Interface for High Performance Storage System

Lyon August 2006
Gilberto Diaz

Contents

1 Introduction

This report shows the work made in Centre de Calcul de l'Institut National de Physique Nucléaire et de Physique des Particules (CC-IN2P3) in Lyon - France under the program "High-Energy Latin-European Network" which is a scientific cooperation agreement, signed by CERN, DESY, labs/universities from eight latin-american countries (Argentina, Brasil, Chile, Colombia, Cuba, México, Perú y Venezuela), and seven european countries (Spain, France, Portugal, Italy, Greece, Sweden, UK).

This work consisted in develop a web interface prototype to administrate a mass storage system installed at the IN2P3 (High Performance Storage System, HPSS) in order to have a remote mechanism that provides common administration tasks capability. HPSS is used as data repository to support experiment calculations from several areas physics, bioscience, astrophysics, etc. and the treatment of data comming from accelerators, observations, etc.

HPSS has a graphical user interface and some commands to administrate the system. However, none of them is a good option when system administrator are in remote places and they don't have all functionalities to execute the common daily tasks. Therefore, it is necessary implement a mechanism that provide remote administration considering all security aspects.

2 HPSS Description

HPSS is a hierarchical scalable parallel storage manager, a commercial software from IBM. It is used in environments where large data storage is needed.

The main aim of HPSS is provides a high performance storage for environments that requires

- Store large amount of data
- Large single logical name space
- Distributed, secure access to data

HPSS utilizes a multi level hierarchy to store data. Usually, data most recently accessed is stored in low latency (high cost) media, while data less recently accessed is moved progressively to higher latency (lower cost) devices. Thus, space in low latency media is free up to be available for other clients. The figure 3 shows a classical configuration.

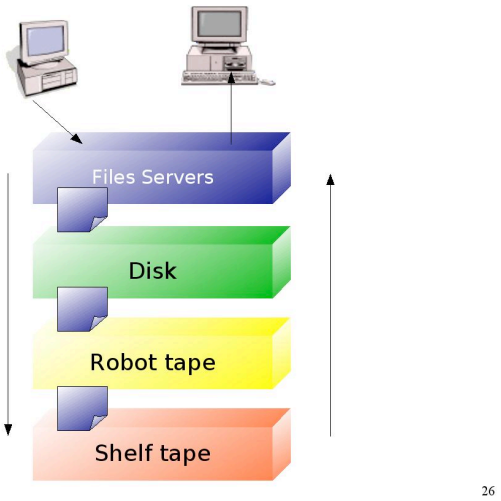


Figure 1: HPSS hierarchy example

HPSS has different logical elements:

- **Storage Classes** A logical set of similar media intended to store files with similar characteristics (file size, tape technology, access frequency, etc).
- **Storage Hierarchies** consist of different levels. Each level is a separate storage class. New files are stored at the top level. A migrated file could be purged in order to free up storage space. Files are copied to the top level on demmand.
- **Classes of Service** This is the way for an user for choosing management policy. It Associates several performance related characteristics, like file size and migration policies, with a hierarchy. Every client utilizes a class of service according its requirements.

3 Web Interface

This section describes the features and design of the web interface prototype. As the initial function set to be added to this prototype, the storage team of

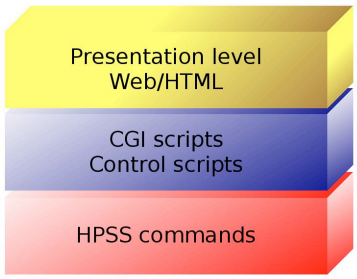
the CC-IN2P3 considered some common system administration tasks related to the management of the tapes and the process that use them. These functions are listed bellow.

- List virtual volumes in order to see its state and other features
- List available drives in the system and check its state
- List jobs using some resources in the system
- Lock or unlock virtual volumes
- Lock or unlock tape drives
- Check if the HPSS daemons are running

In this sense, the web interface incorporates these functions in order to provides a remote way to execute this kind of administrative tasks.

3.1 Software Architecture

The software architecture of this prototype has three parts. The figure 2 shows the different layers.



11

Figure 2: Software Architecture

The lowest layer is conformed by HPSS commands which are used to execute all the functions of the web interface. The functions of the middle layer

is: load all the enviroment variables needed by HPSS commands, execute them, parse the output and control the parameters that come from the web interface. The top layer convert the output in HTML language in order to displayed in the browser.

3.2 User Interface

From the user's point of view the web interface is conformed by three frames:

- Commands frame: Contain all of the buttons and links to initiate every action described above.
- Display frame: Is the part where all output, comming from the different CGIs, are displayed
- Monitor frame: Display the state of two HPSS daemons (hpss_ssmsm and hpss_ssmds) in charge of the control of the functions considered above.

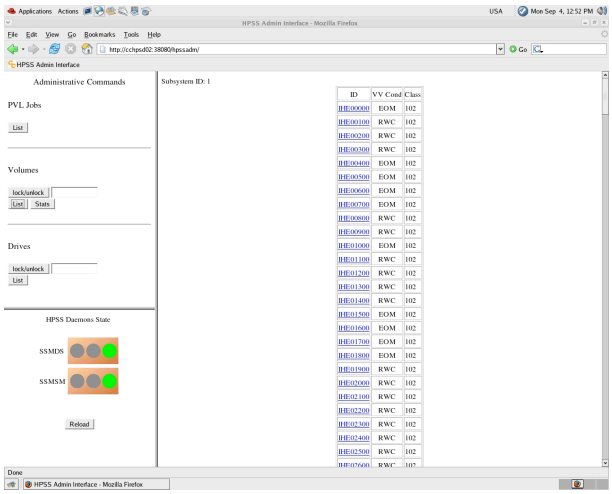


Figure 3: Main window

3.3 Data flow

The whole interface was developed using the following programming tools: perl scripts, shell scripts and java scripts. The data flow of the different programs, to execute a specific action or function of the web interface, is showed using the following diagrams.

When a user press the lock/unlock button in the volume section of the interface, after introduce the volume ID, the script `process_vol1.pl` is called. This executes the script `dump_sspvs.sh` which calls the HPSS command (`dump_sspvs`) in order to find the current state of the volume and validates if the volume exists. Then, parse its output and display another web page (small yellow circle) asking for a confirmation to lock or unlock the volume, depending of the state. If the user answers affirmatively then the `process_vol2.pl` is called, which executes the script `process_vol.sh` and finally execute the right action (lock or unlock the corresponding volume).

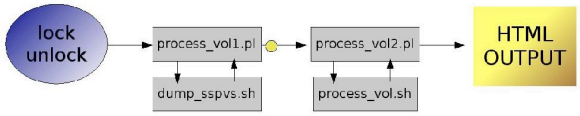


Figure 4: Volumes Management (a)

The procedure described above is very similar for the rest of the functions. Every perl and shell script is well commented inside, and together with these diagrams, any other person can understand how this system works.

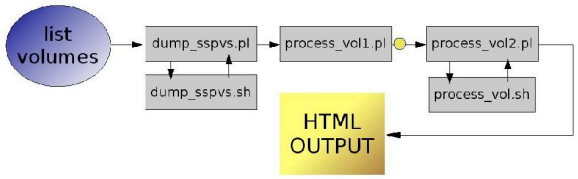


Figure 5: Volumes Management (b)

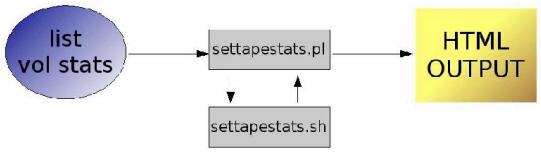


Figure 6: Volumes Management (c)

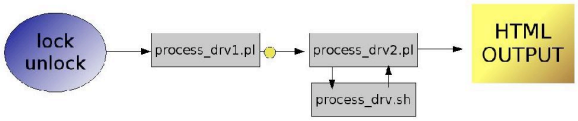


Figure 7: Drivers Management (a)

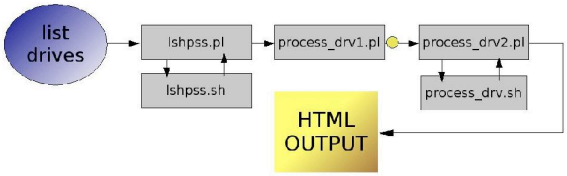


Figure 8: Drivers Management (b)

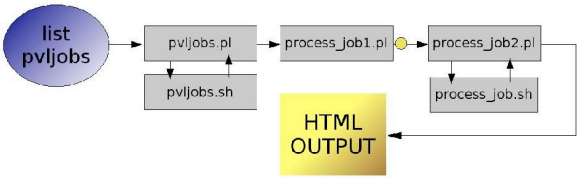


Figure 9: Process Management

4 Conclusions and Recommendations

After finish this work we can say that it is possible to develop a friendly interface for administrate HPSS in remote environments and suitable to execute common daily tasks. During this period only a small prototype was developed, however all the functions selected by the storage team of the CC-IN2P3 were programmed. To create a tool completely functional it is necessary invest more time in this work.

In production system this web interface take long time to display outputs due the big amount of objects. We can develop the lowest layer using another programming language in order to improve the performance instead of reuse the existing commands and increase the usability of the web interface.