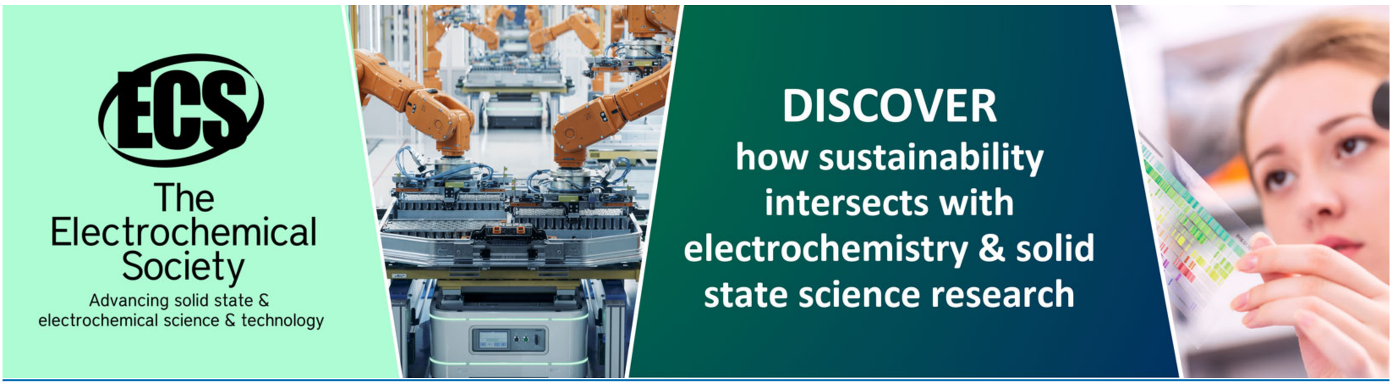
Virtual pools for interactive analysis and software development through an integrated Cloud environment


ECS
The Electrochemical Society
Advancing solid state & electrochemical science & technology
DISCOVER
how sustainability intersects with electrochemistry & solid state science research

Virtual pools for interactive analysis and software development through an integrated Cloud environment

C.Grandi[1]; A.Italiano[2]; D. Salomoni[2]; A.K.Calabrese Melcarne[2]

[1]INFN-Bologna; [2]INFN-CNAF

E-mail: Claudio.Grandi@cern.ch; Alessandro.Italiano@cnaf.infn.it; Davide.Salomoni@cnaf.infn.it; Karen.Calabrese.Melcarne@cnaf.infn.it

Abstract. WNoDeS, an acronym for **W**orker **N**odes **o**n **De**mand **S**ervice, is software developed at CNAF-Tier1, the National Computing Centre of the Italian Institute for Nuclear Physics (INFN) located in Bologna. WNoDeS provides on demand, integrated access to both Grid and Cloud resources through virtualization technologies. Besides the traditional use of computing resources in batch mode, users need to have interactive and local access to a number of systems. WNoDeS can dynamically select these computers instantiating Virtual Machines, according to the requirements (computing, storage and network resources) of users through either the Open Cloud Computing Interface API, or through a web console. An interactive use is usually limited to activities in user space, i.e. where the machine configuration is not modified. In some other instances the activity concerns development and testing of services and thus implies the modification of the system configuration (and, therefore, root-access to the resource). The former use case is a simple extension of the WNoDeS approach, where the resource is provided in interactive mode. The latter implies saving the virtual image at the end of each user session so that it can be presented to the user at subsequent requests. This work describes how the LHC experiments at INFN-Bologna are testing and making use of these dynamically created ad-hoc machines via WNoDeS to support flexible, interactive analysis and software development at the INFN Tier-1 Computing Centre.

1. Introduction

Most sites offering remote access also have local users that need to use the resources interactively. Furthermore they need local access to the batch system and user interfaces to access the distributed infrastructure.

A classic model would be to statically allocate machines for the interactive access. These machines would then have local access to the batch system and user interface software. In the worst case if special configurations are needed for the batch nodes used locally as opposed to those used via remote access, the local batch farm nodes would need to be allocated statically as well.

We describe here a solution based on WNoDeS [1] to dynamically allocate resources to different tasks regardless of the need for specific configurations.

This paper is organized as follows. After an overview of the typical approaches used to handle computing resources in Section 2, we show a new, dynamic way to manage them in Section 3. Then, in Section 4, we describe the main components of WNoDeS, followed by a description of the tests we have been doing in Section 5. Finally, Section 6 presents concluding remarks as well as future directions.

2. Classical approach: static resource allocation model

Computing resources at a typical university or research institute may be logically partitioned in three sections:

- A pool of machines provides the local users with interactive access directly from their desktop or in some cases through bastion hosts. On these machines the users have the environment needed to develop software, to test their programs on small samples of data, to do interactive analysis on reduced data via ROOT [2] or similar programs, and so on. Furthermore they may find the software needed to have access to the remote infrastructure, i.e. the so called Grid User Interface (UI) software. Typically these machines have access to a local shared file system where the experiment software and their home directories are located and to local disk resources where personal or group data may be placed.

- A pool of machines provides the local users with batch access via direct submission from the interactive pool. These machines are used for local purpose production tasks or analysis via facilities such as PROOF [3]. Typically these machines have access to the same file systems of the machines of the interactive pool.
- A pool of machines provides remote batch access via Grid interfaces (Computing Element, CE). Typically these machines do not have access to local shared file systems but only to the local Grid Storage Element (SE).

This architecture, summarized in Figure 1, has the limitation that resources are statically allocated to a pool and idle machines of one pool cannot be easily allocated to another one in case of need. For instance we may expect that during nights or week-ends the interactive pool may be underutilized while the grid pool may continue to be used.

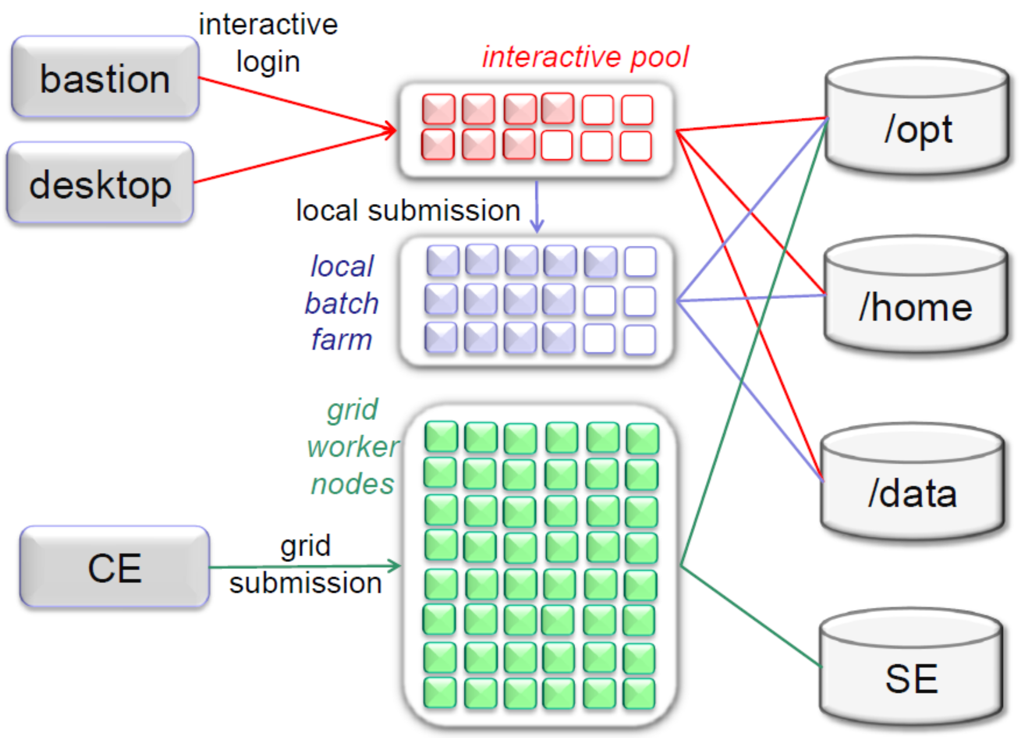


Figure 1: Allocation of resources in a site configured following a static model

3. New approach: dynamic resource allocation model

In a dynamic resource allocation model, described in figure 2, machines are allocated to a pool only at the time they are requested. All idle machines are not pre-configured and may be configured in any of the needed flavours. In the following we describe a technique based on virtualization and batch system features to achieve this result.

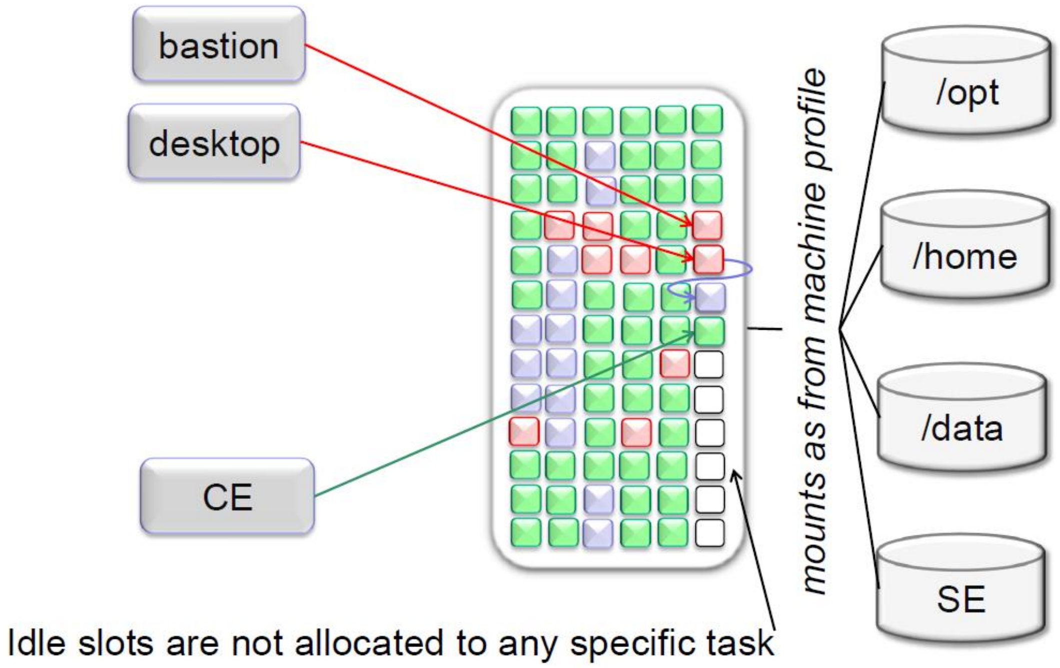

bastion
desktop
CE
mounts as from machine profile
/opt
/home
/data
SE
Idle slots are not allocated to any specific task

Figure 2: Allocation of resources in a site configured following a dynamic model

4. The WNoDeS Project

The WNoDeS Project [1][4] started a few years ago at INFN, the Italian National Institute for Nuclear Physics, with the aim to provide computing environments tailored to the specific user requirements. In fact, the environment required by different Virtual Organizations (VOs) may differ at several levels: for example for what regards the Operating Systems (or version thereof), or the presence (or absence) of specific program or libraries. This issue is normally solved dedicating a subset of the available resources to different sets of users. However this has the important negative side-effect that farm administrators cannot generally optimize the usage of farm resources. Exploiting virtualization techniques and batch system features, INFN has developed an architecture that makes it possible to run batch jobs on virtual worker nodes, i.e. on computing resources functionally identical to traditional servers, but running on Virtual Machines (VM). The following paragraphs briefly describe the type of interfaces provided by WNoDeS.

4.1. Local job execution

VMs are dynamically instantiated at the time of job execution; a job will then run on to one or more of these VMs. The images used to create the VMs can be selected either by the site administrator (maybe statically, differentiating images across users), or directly by users (if authorized to do so). A positive side effect of this VM-based approach is that the execution environment for jobs running on the same physical host is logically separated, thus easing some of the security concerns related to the sharing of resources among many users and VOs.

4.2. Grid integration

Since the execution environment provided as described in Section 4.1 is absolutely identical to the one that a user could get on a real host, it is straightforward to allocate resources to jobs submitted through a grid interface in the same way. Using the gLite CREAM Compute Element [5] it is also possible to specify the characteristics of the required environment (e.g. the Operating System of the virtual machine) in the job requirements using the Job Description Language (JDL) thus providing the users with more flexibility [4]. Authentication and authorization are handled by VOMS [7] and by gLite ARGUS [8] respectively.

4.3. Cloud computing

Requirements for making use of Cloud Computing services are seen with increasing frequency. Europe has largely invested in Grid Computing in the past years, establishing an infrastructure that is now used by several thousands of scientists in full production. Making Cloud services available exploiting the same resources being used to provide Grid services is then absolutely desirable. WNoDeS enables the on-demand instantiation of VMs using the OGF Open Cloud Computing Interface (OCCI) [6] API, either directly or via a Web interface. Authentication is handled through a gateway allowing users to gain access to resources through several authentication methods [9], while authorization is handled via gLite Argus [8], like in section 4.2.

4.4. Virtual Interactive Pool

The WNoDeS Virtual Interactive Pool (VIP) interface, built on top of the WNoDeS cloud computing interface described in Section 4.3, is targeted at local users who can, through provided scripts, autonomously create customized computing resources.

The user requests the instantiation of a VM via the VIP Command Line Interface (CLI). The CLI allows specifying the requirements in terms of RAM, number of CPUs, network bandwidth and file systems mount points. Once the request is made, the user may wait until login to the resource is possible or, alternatively, may choose to get back a request ID that can be used to check the request status; when the resource is ready, he will eventually be able to login on to it.

From an implementation point of view, the request goes through the standard authentication and authorization layers of WNoDeS; a fake batch job, invisible to the user, containing the specified resource requirements is submitted to the local batch system and dispatched to a *bait* host [4]. Similarly to other interfaces supported by WNoDeS, on the bait the batch job execution is immediately stopped and the process to instantiate the VM starts. When the VM is ready, WNoDeS customizes the machine, e.g. creating the requested mount points. The VM can now be used and is therefore made available to the requester. The associated batch job on the bait is resumed and used for accounting purposes; like in Cloud interfaces, only wall clock time is considered for charging resource usage. At logout, WNoDeS automatically kills the fake batch job and destroys the VIP VM.

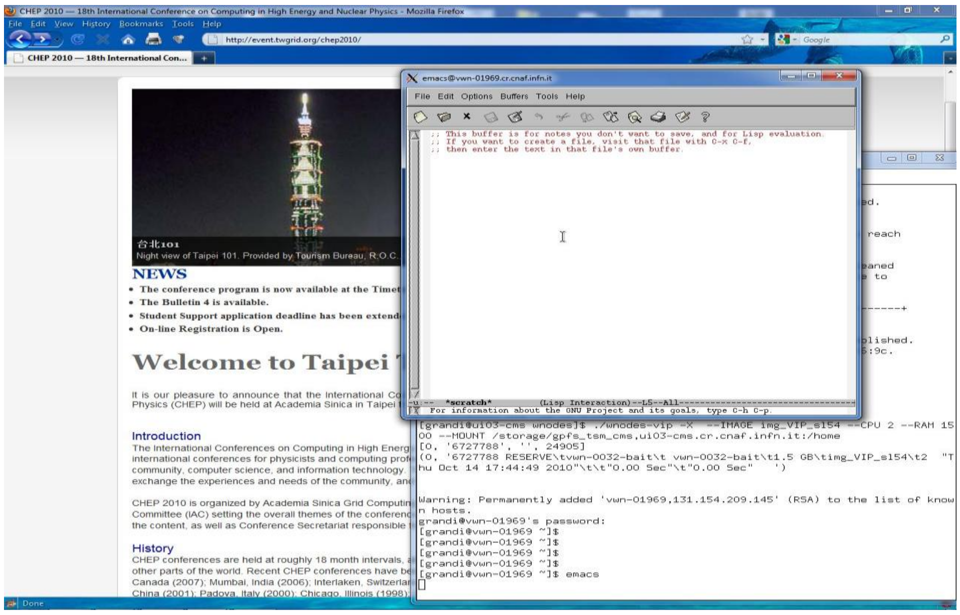
5. First tests on the INFN-Bologna Tier-3 site

VIP is being developed and a prototype is being tested on the INFN-Bologna Tier-3 site. The Bologna Tier-3 is a joint project of INFN and Università degli Studi di Bologna. It supports local research groups including some of the LHC experiments. The Tier-3 resources are fully integrated with the INFN-CNAF Tier-1 infrastructure. The Tier-3 consists of about 50 dual quad-core machines accessible through the INFN Tier-1 LSF batch system, and of 150 TB of disk on the INFN Tier-1 GPFS storage system. The boundaries between the Tier-1 and Tier-3 are virtual. Using WNoDeS each site can expand into the other according to the policies managed by LSF. Only the profile of the VMs define to which site they belong. Computing and Storage Elements, on the other hand, are independent. Differently from the Tier-1 the Tier-3 also offers interactive services to local users, and these are implemented via the VIP prototype.

The first tests on the Tier-3 farm show that the main needed functionalities are available in VIP. Figure 3 shows how a virtual machine is instantiated via the VIP CLI specifying the image name, the number of CPUs, the amount of RAM and the file systems that need to be available on the VM. If requested the CLI also sets up the needed tunnelling to start X11 applications.

Figure 4 shows how the file systems, including the user home directory, have been made available on the VM.

Finally figure 5 shows the characteristics of the CPUs and memory of the VM.


emacs@vwn-01969.cr.cnaf.infn.it
File Edit Options Buffers Tools Help

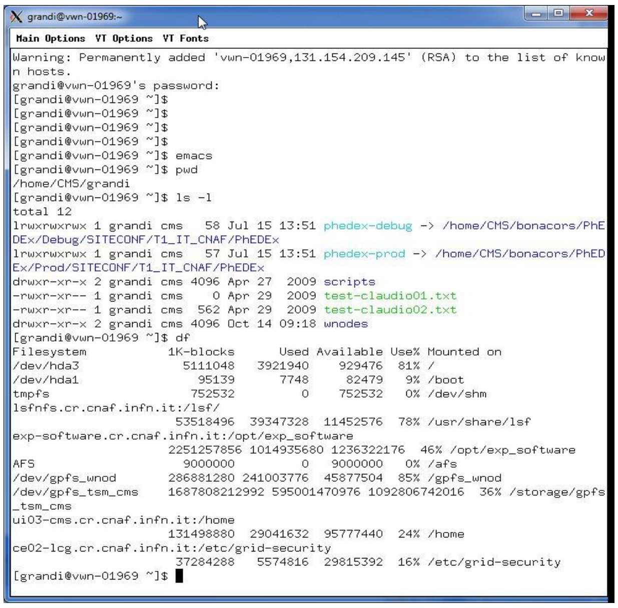
```
;; This buffer is for notes you don't want to save, and for Lisp evaluation.
;; If you want to create a file, visit that file with C-x C-f,
;; then enter the text in that file's own buffer.

-u:--  *scratch*      (Lisp Interaction)--L5--All-------------------------------
For information about the GNU Project and its goals, type C-h C-p.
```

```
[grandi@ui03-cms wnodes]$ ./wnodes-vip -X  --IMAGE img_VIP_sl54 --CPU 2 --RAM 15
00 --MOUNT /storage/gpfs_tsm_cms,ui03-cms.cr.cnaf.infn.it:/home
[0, '6727788', '', 24905]
(0, '6727788 RESERVE\tvwn-0032-bait\t vwn-0032-bait\t1.5 GB\timg_VIP_sl54\t2  "T
hu Oct 14 17:44:49 2010"\t\t"0.00 Sec"\t"0.00 Sec"    ')


Warning: Permanently added 'vwn-01969,131.154.209.145' (RSA) to the list of know
n hosts.
grandi@vwn-01969's password:
[grandi@vwn-01969 ~]$
[grandi@vwn-01969 ~]$
[grandi@vwn-01969 ~]$
[grandi@vwn-01969 ~]$
[grandi@vwn-01969 ~]$ emacs
```

Figure 3: Instantiation of the virtual machine and execution of X-applications

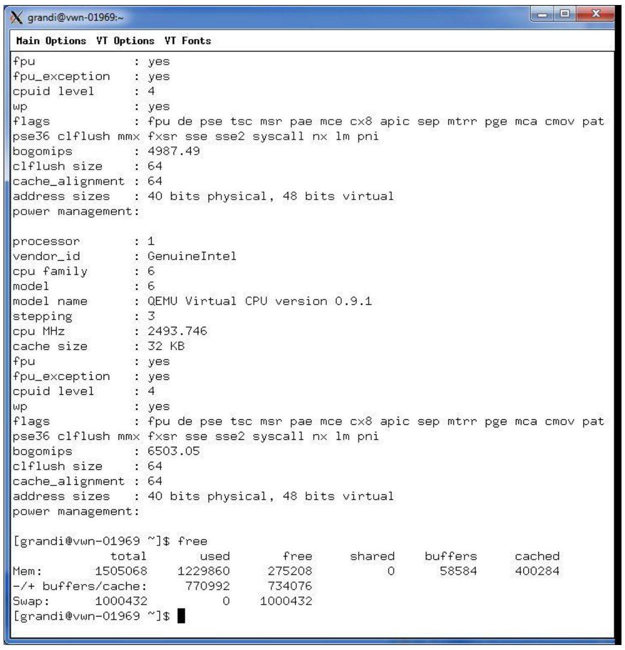


```
Warning: Permanently added 'vwn-01969,131.154.209.145' (RSA) to the list of know
n hosts.
grandi@vwn-01969's password:
[grandi@vwn-01969 ~]$
[grandi@vwn-01969 ~]$
[grandi@vwn-01969 ~]$
[grandi@vwn-01969 ~]$
[grandi@vwn-01969 ~]$ emacs
[grandi@vwn-01969 ~]$ pwd
/home/CMS/grandi
[grandi@vwn-01969 ~]$ ls -l
total 12
lrwxrwxrwx 1 grandi cms   58 Jul 15 13:51 phedex-debug -> /home/CMS/bonacors/PhE
DEx/Debug/SITECONF/T1_IT_CNAF/PhEDEx
lrwxrwxrwx 1 grandi cms   57 Jul 15 13:51 phedex-prod -> /home/CMS/bonacors/PhED
Ex/Prod/SITECONF/T1_IT_CNAF/PhEDEx
drwxr-xr-x 2 grandi cms 4096 Apr 27  2009 scripts
-rwxr-xr-- 1 grandi cms    0 Apr 29  2009 test-claudio01.txt
-rwxr-xr-- 1 grandi cms  562 Apr 29  2009 test-claudio02.txt
drwxr-xr-x 2 grandi cms 4096 Oct 14 09:18 wnodes
[grandi@vwn-01969 ~]$ df
Filesystem           1K-blocks      Used Available Use% Mounted on
/dev/hda3              5111048   3921940    929476  81% /
/dev/hda1                95139      7748     82479   9% /boot
tmpfs                   752532         0    752532   0% /dev/shm
lsfnfs.cr.cnaf.infn.it:/lsf/
                      53518496  39347328  11452576  78% /usr/share/lsf
exp-software.cr.cnaf.infn.it:/opt/exp_software
                     2251257856 1014935680 1236322176  46% /opt/exp_software
AFS                    9000000         0   9000000   0% /afs
/dev/gpfs_wnod       286881280 241003776  45877504  85% /gpfs_wnod
/dev/gpfs_tsm_cms    1687808212992 595001470976 1092806742016  36% /storage/gpfs
_tsm_cms
ui03-cms.cr.cnaf.infn.it:/home
                     131498880  29041632  95777440  24% /home
ce02-lcg.cr.cnaf.infn.it:/etc/grid-security
                      37284288   5574816  29815392  16% /etc/grid-security
[grandi@vwn-01969 ~]$ █
```

Figure 4: File systems available on the virtual machine

6. Summary and outlook

Exploiting the WNoDeS architecture, we developed and tested VIP, a suite that allows users to instantiate VMs on-demand for interactive use out of a common pool of machines used for heterogeneous purposes (local and remote batch access via Grid and Cloud technologies). A prototype has been installed on the INFN-Bologna Tier-3 at INFN-CNAF, the Italian Tier-1 centre. While the basic functionalities are met by the prototype, there are features that are mandatory before such a system may be used in production. The VIP client needs to be properly packaged so that it can be easily installed on users' desktops. Due to the need to allocate resources through the normal batch system resource allocation procedures, the instantiation of the interactive machine out of the standard batch pool currently takes too long for typical interactive usage. We foresee to implement a three-level system for instantiating VMs for this purpose:



```
fpu             : yes
fpu_exception   : yes
cpuid level     : 4
wp              : yes
flags           : fpu de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat
pse36 clflush mmx fxsr sse sse2 syscall nx lm pni
bogomips        : 4987.49
clflush size    : 64
cache_alignment : 64
address sizes   : 40 bits physical, 48 bits virtual
power management:

processor       : 1
vendor_id       : GenuineIntel
cpu family      : 6
model           : 6
model name      : QEMU Virtual CPU version 0.9.1
stepping        : 3
cpu MHz         : 2493.746
cache size      : 32 KB
fpu             : yes
fpu_exception   : yes
cpuid level     : 4
wp              : yes
flags           : fpu de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat
pse36 clflush mmx fxsr sse sse2 syscall nx lm pni
bogomips        : 6503.05
clflush size    : 64
cache_alignment : 64
address sizes   : 40 bits physical, 48 bits virtual
power management:

[grandi@vwn-01969 ~]$ free
             total       used       free     shared    buffers     cached
Mem:       1505068    1229860     275208          0      58584     400284
-/+ buffers/cache:     770992     734076
Swap:      1000432          0    1000432
[grandi@vwn-01969 ~]$ █
```

Figure 5: Characteristics of the virtual machine

1. A few VM for each supported VO with a standard set up are kept active at all times. Access to these machines is fast (comparable to a normal `ssh`).
2. A few more batch slots are kept idle at all times. In case a machine with the standard set up is not available or if a non-standard set-up is requested by the user, a new VM is instantiated out of this pool. Access to these machines takes typically less than a minute.
3. If the reserved batch slots are already taken the user request is queued to the batch system and the machine is made available to the user with the latencies of the batch system, which could be anyway configured and tuned to submit such requests to special, fast-execution queues.

Finally we would like to address the use case of a user needing to save the VM status. Currently each time the user starts an interactive session gets a fresh VM. The only persistent data is on the mounted file systems. The new functionality may be useful e.g. to test services to be installed on the VM not in user space. In general it requires root access. This is a typical Cloud-type service and will be implemented by saving a snapshot of the VM on the storage area before destroying it. The saved image may then be later retrieved via the VIP CLI.

References

[1] WNoDeS web site: http://web.infn.it/wnodes
[2] Root web site: http://root.cern.ch/
[3] G Ganis, J Iwaszkiewicz, F Rademakers "Data Analysis with PROOF", PoS(ACAT08)007, Proceedings of ACAT 2008 Conference, Erice (Italy)
[4] D.Salomoni, A.Italiano, E.Ronchieri, "WNoDeS, a tool for integrated Grid and Cloud access and computing farm virtualization", Proceedings of International Conference on Computing in High Energy and Nuclear Physics - CHEP'10, Taipei, Taiwan (2011)
[5] C.Aiftimiei et al., "Design and Implementation of the gLite CREAM Job Management Service", Future Generation Computer Systems, Volume 26, Issue 4 (2010)
[6] The OGF Open Cloud Computing Interface http://www.occi-wg.org/doku.php
[7] Virtual Organization Membership Service, https://twiki.cnaf.infn.it.twiki/bin/view/VOMS/WebHome
[8] gLite Argus, https://twiki.cern.ch/twiki/bin/view/EGEE/AuthorizationFramework
[9] D.Salomoni, V.Ciaschini, "An Authentication Gateway for Integrated Grid and Cloud Access", Proceedings of International Conference on Computing in High Energy and Nuclear Physics - CHEP'10, Taipei, Taiwan (2011)