INFN SEZIONE DI TORINO
ISTITUTO NAZIONALE DI FISICA NUCLEARE

UNIVERSITÀ DEGLI STUDI DI TORINO

Status of Cloud Toy, a Computing Solution

Belle2 Meeting, Trieste, 4 May 2017

Jacopo Pellegrino

Cloud Toy

- Cloud Computing
- The Aim
- How it Works

- Current Tests
- Running Instances

Overview → Development → Test → Conclusion

- Hardware
- Optimisation
- Improvements

Overview

- Cloud Computing
- The Aim
- How it works

Cloud Computing

The cloud computing paradigm provides access to a shared pool of computational and storage resources.

Setting a cloud infrastructure up may not be trivial by the providers' point of view.

Cloud computing is widely adopted for HEP purposes.

The goal for:

- users is to be able to access computational resources when needed.
- providers is to maximize the efficiency of the cloud infrastructure.

The Aim

The main idea behind this work is to address two common issues of cloud infrastructures:

- Usability: simplifying the setup and installation process.
- Efficiency: making the usage of resources more dynamic, flexible and efficient.

The aim is to improve existing cloud infrastructures and give sites with limited manpower/knowledge easier access to cloud technologies.

The Aim

- Usability: simplifying the setup and installation process:

setting up a OpenNebula hypervisor minimizing the user interaction during the installation process.

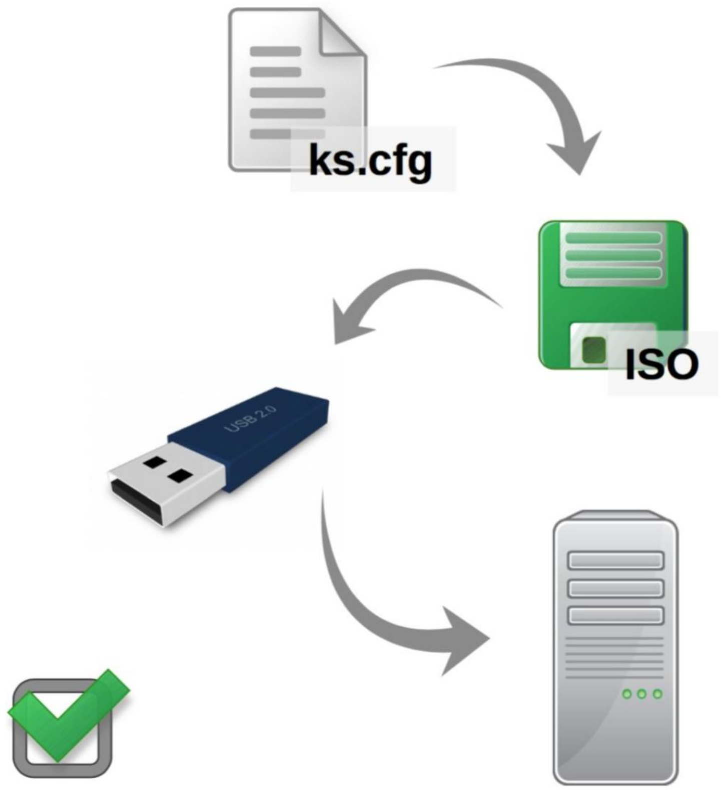
- Efficiency: making the usage of resources more dynamic, flexible and efficient:

allow for a dynamic use of resources according to actual needs of users within a multi-tenants cloud infrastructure.

How it Works - Automatic Installation

Automatization of Installation process:

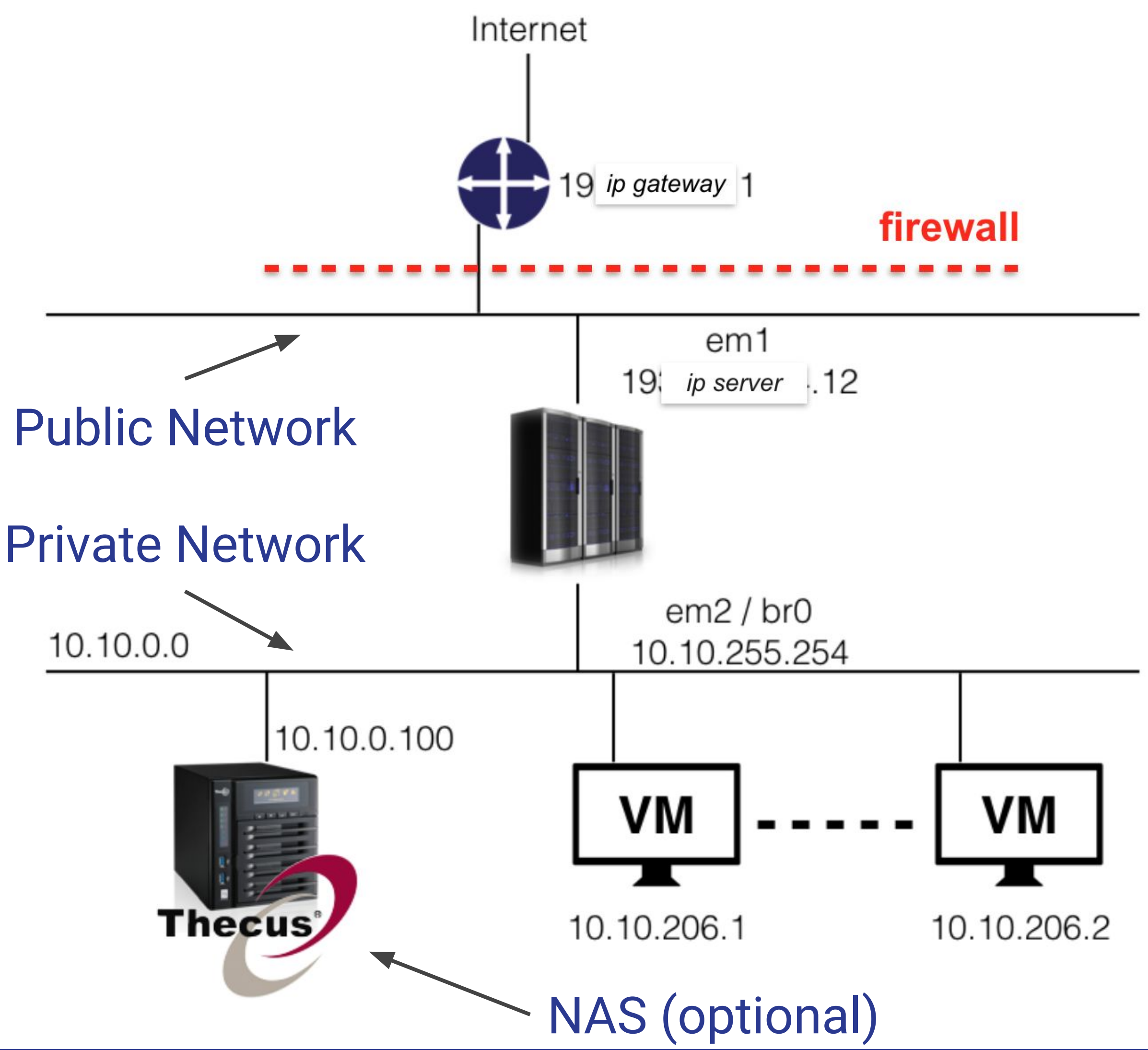
- Creation of kickstart file.
- Preparation of customized ISO.
- Preparation of bootable usb drive.
- Installation on server.
- Test of the installation.



How it Works - Automatic Installation

- Creation of kickstart file: configuration parameters, software packages, disks partitioning, and the like...
- Preparation of customized ISO: start from a standard netinstall iso file, make it look for the kickstart at boot time.
- Preparation of bootable usb drive: burn the iso into a usb drive so that it is possible to boot from it.
- Installation on server: plug the usb drive and reboot.
- Test of the installation: the server is ready in about 1 hour.

How it Works - Automatic Installation



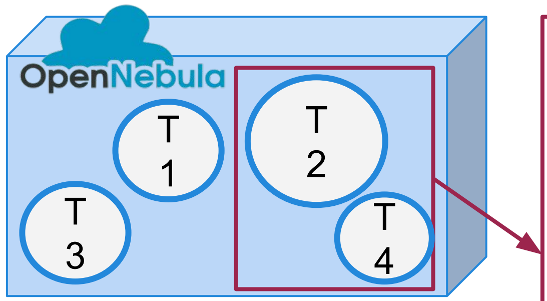
Schematic of the infrastructure resulting from the automatic installation.

Open Ports:
9869 Sunstone
4567 OCCI
11443 rOCCI
29876 proxy VNC

How it Works - Elasticity

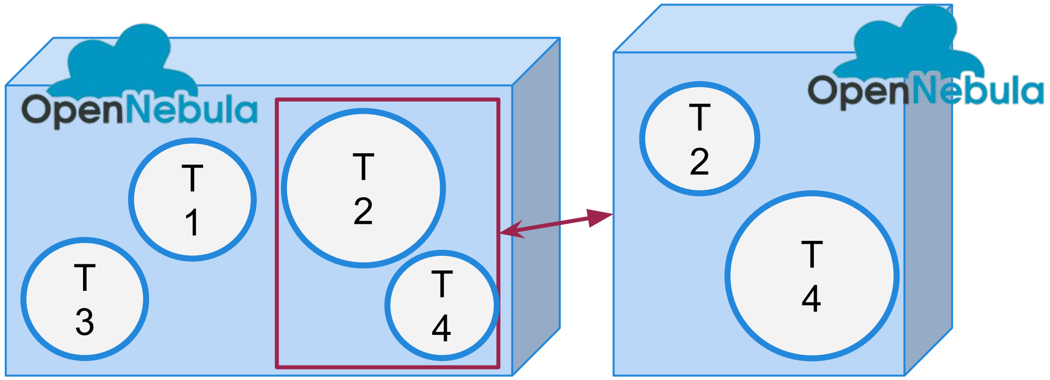
Elasticity:

- An infrastructure used by different stakeholders.
- The aim is to dynamically change the quota of resources.
- A “private” infrastructure is introduced.



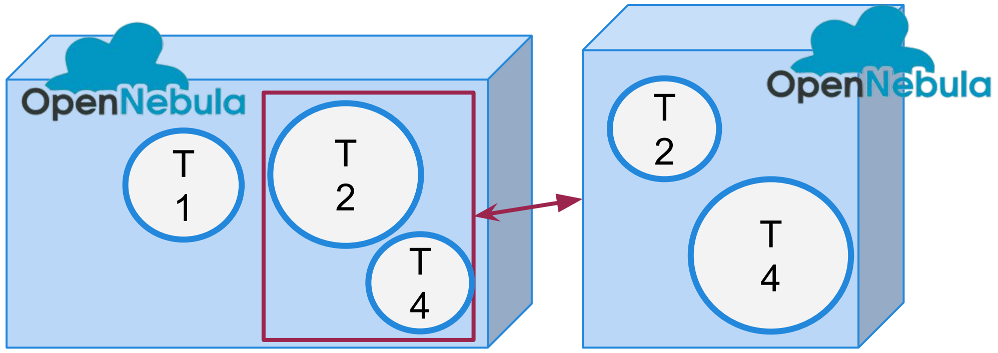
T2 has 100 cores
T4 has 200 cores
Together they have 300

The aim is to manage these 300 cores dynamically for T2 and T4

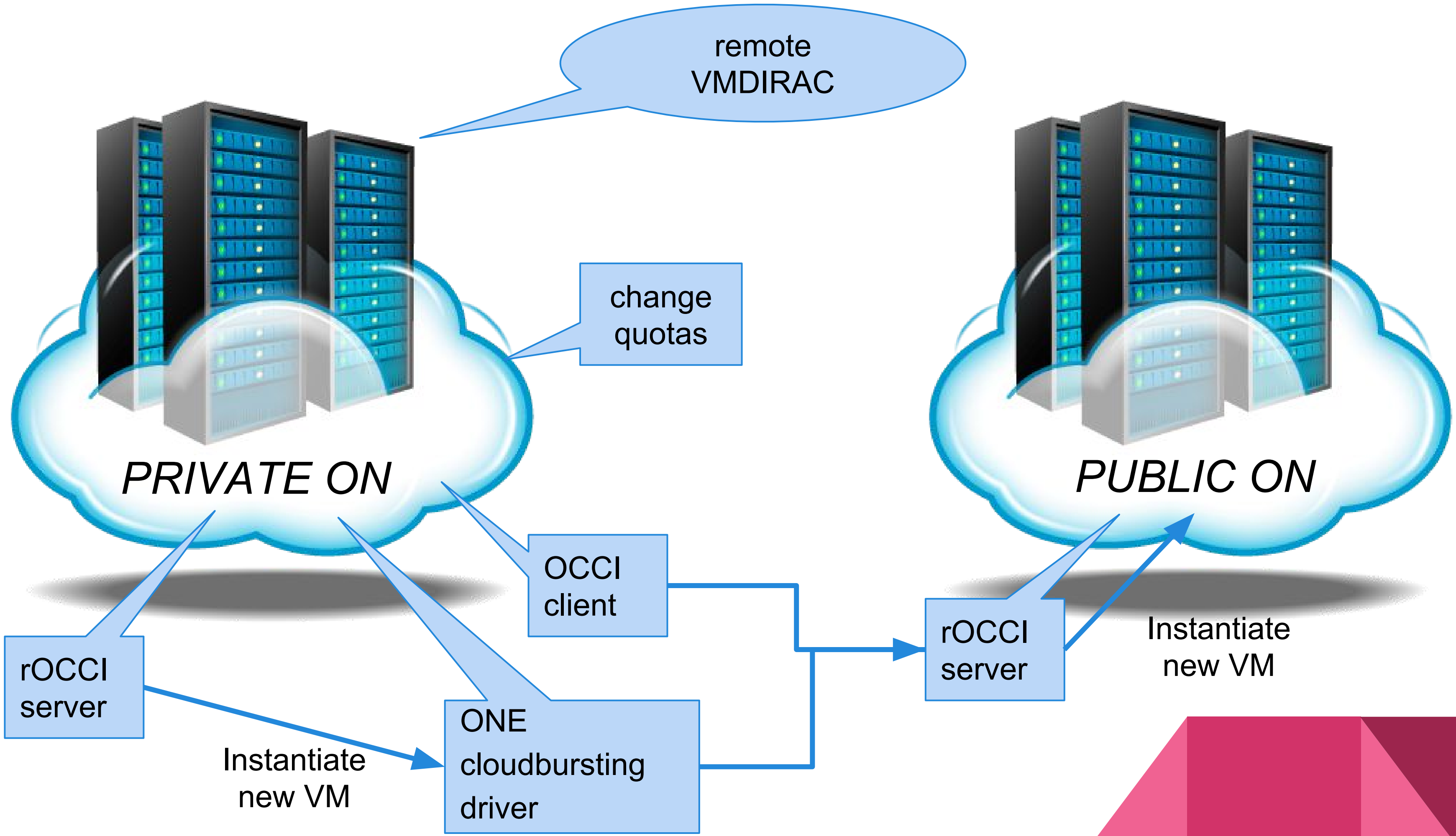


How it Works - Elasticity

- The “public” OpenNebula only knows that T2 and T4 have 300 cores (user-level control).
- The “private” OpenNebula can dynamically change quotas between T2 and T4 according to actual needs (full sys-man control).
- Variations of quotas can be performed via oneadmin user.
- The ONE-cloudbursting-driver is currently under test.
  Aim: start VMs on Public OpenNebula from the Private OpenNebula.



How it Works - Elasticity



Development

- Hardware
- Optimisation
- Improvements

Hardware

Aim: setting up a OpenNebula hypervisor minimizing the user interaction during the installation process.

Installation performed via usb drive.

Server used for test:

Machine	Dell PowerEdge R630
CPU	2 x Intel(r) Xeon(r) E5-2650 v3 @ 2.30 GHz
Cores	20 physical, 40 hyper threading
RAM (GB)	160

Machine	HP ProLiant DL360G7
CPU	2 x Intel(r) Xeon(r) E5-506 v3 @ 2.13 GHz
Cores	4 physical, 8 hyper threading
RAM (GB)	12

Optimisation

Repeating the installation in few months brought to issues concerning missing software packages and ruby gems required by OpenNebula:

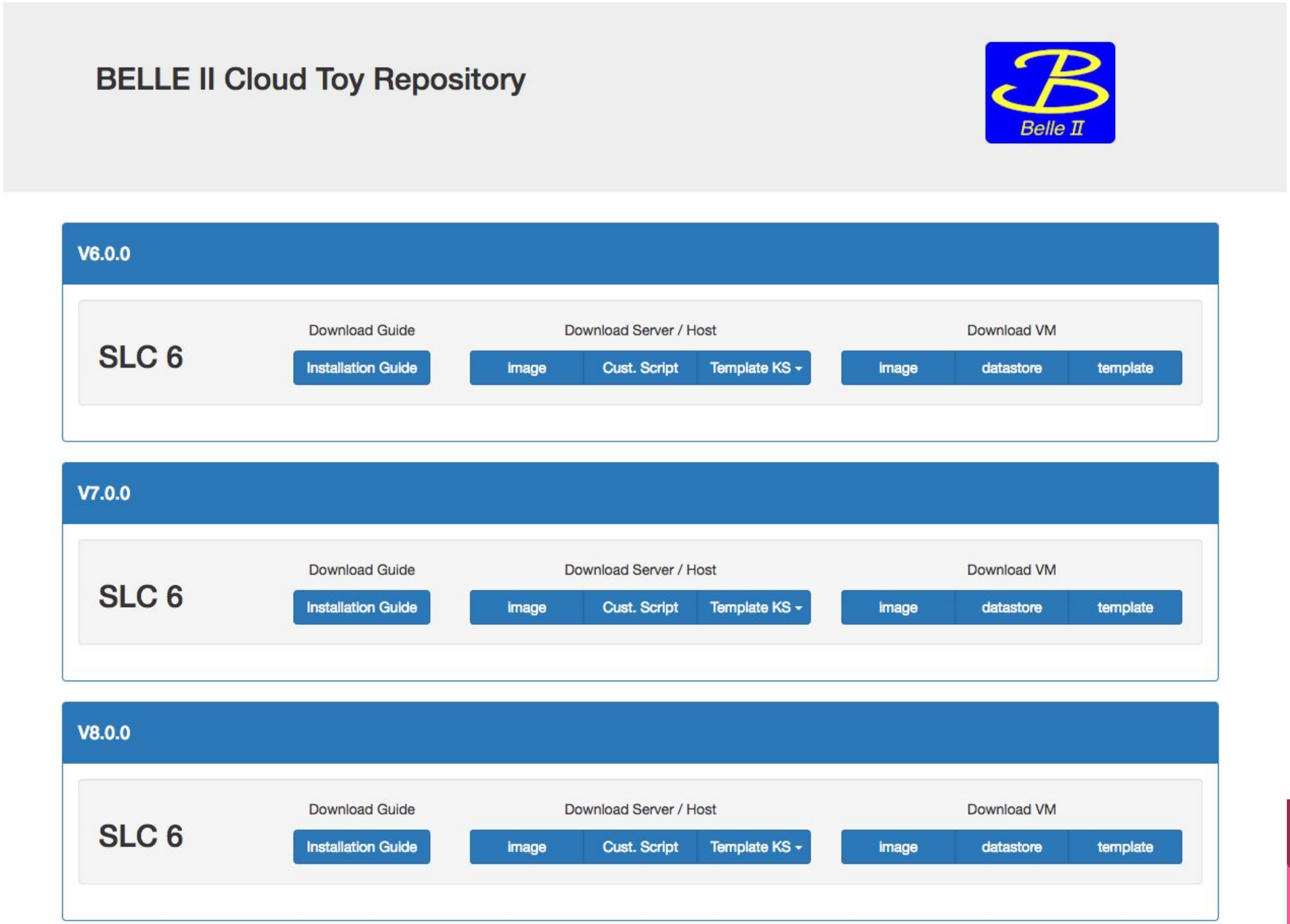
- Installed software packages are downloaded and saved to avoid dependencies issues in the future.
- A dedicated web repository can be created to store all the required software.

Improvements

Some improvements:

- Creation of a Kickstart Template that users can easily customize via a dedicated script.
- Possibility to install a host to be added to an existing infrastructure.
- Creation of a web repository where images, kickstart files and templates are stored. By now is accessible with credentials since under test.
- PBS installation on hypervisor and nodes.
- VMDirac + rOCCI for job submission.

See the Cloud Toy webpage



BELLE II Cloud Toy Repository

V6.0.0

SLC 6

Download Guide	Download Server / Host			Download VM		
Installation Guide	image	Cust. Script	Template KS	image	datastore	template

V7.0.0

SLC 6

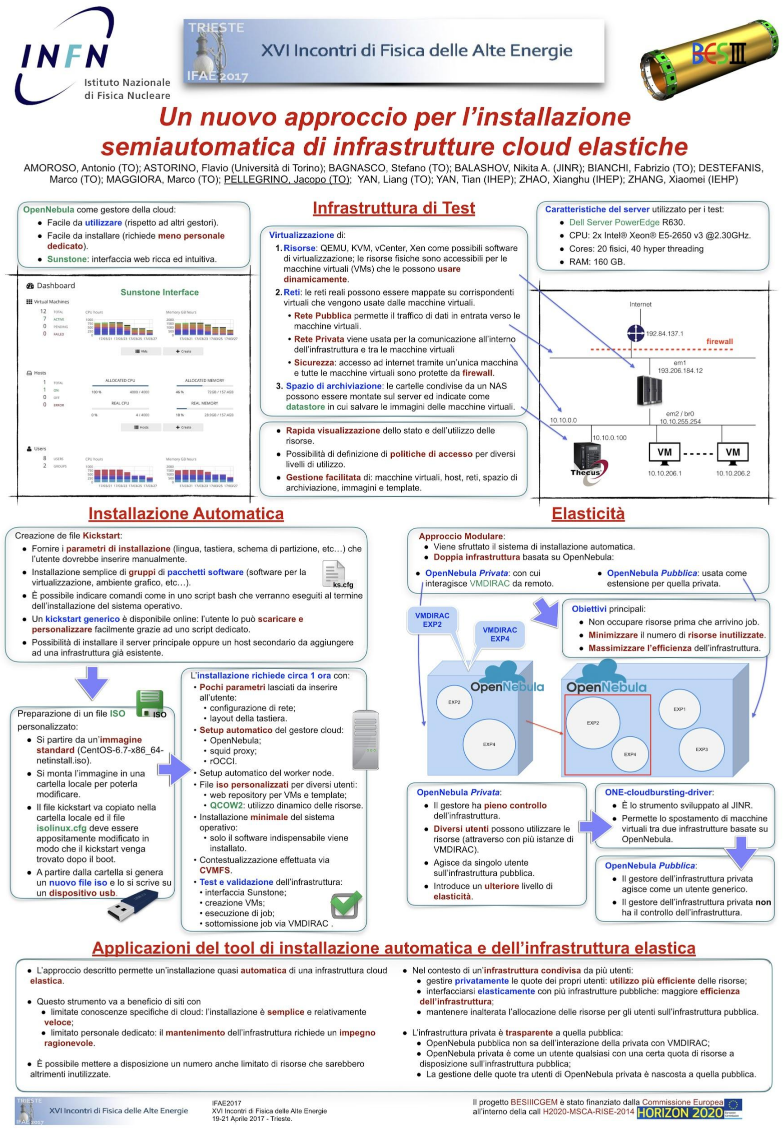
Download Guide	Download Server / Host			Download VM		
Installation Guide	image	Cust. Script	Template KS	image	datastore	template

V8.0.0

SLC 6

Download Guide	Download Server / Host			Download VM		
Installation Guide	image	Cust. Script	Template KS	image	datastore	template

Cloud Toy



XVI Incontri di Fisica delle Alte Energie

TRIESTE IFAE 2017

INFN Istituto Nazionale di Fisica Nucleare

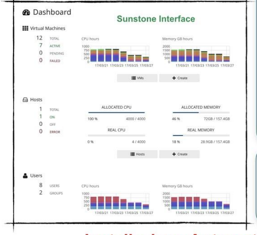
BESIII

Un nuovo approccio per l'installazione semiautomatica di infrastrutture cloud elastiche

AMOROSO, Antonio (TO); ASTORINO, Flavio (Università di Torino); BAGNASCO, Stefano (TO); BALASHOV, Nikita A. (JINR); BIANCHI, Fabrizio (TO); DESTEFANIS, Marco (TO); MAGGIORA, Marco (TO); PELLEGRINO, Jacopo (TO); YAN, Liang (TO); YAN, Tian (IHEP); ZHAO, Xianghu (IHEP); ZHANG, Xiaomei (IEHP)

OpenNebula come gestore della cloud:

- Facile da **utilizzare** (rispetto ad altri gestori).
- Facile da installare (richiede **meno personale dedicato**).
- **Sunstone**: interfaccia web ricca ed intuitiva.



Infrastruttura di Test

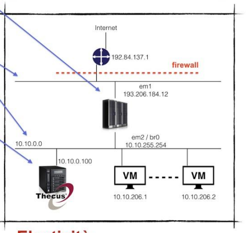
Virtualizzazione di:

1. **Risorse**: QEMU, KVM, vCenter, Xen come possibili software di virtualizzazione; le risorse fisiche sono accessibili per le macchine virtuali (VMs) che le possono **usare dinamicamente**.
2. **Reti**: le reti reali possono essere mappate su corrispondenti virtuali che vengono usate dalle macchine virtuali.
   - **Rete Pubblica** permette il traffico di dati in entrata verso le macchine virtuali.
   - **Rete Privata** viene usata per la comunicazione all'interno dell'infrastruttura e tra le macchine virtuali
   - **Sicurezza**: accesso ad internet tramite un'unica macchina e tutte le macchine virtuali sono protette da **firewall**.
3. **Spazio di archiviazione**: le cartelle condivise da un NAS possono essere montate sul server ed indicate come **datastore** in cui salvare le immagini delle macchine virtuali.

- **Rapida visualizzazione** dello stato e dell'utilizzo delle risorse.
- Possibilità di definizione di **politiche di accesso** per diversi livelli di utilizzo.
- **Gestione facilitata** di: macchine virtuali, host, reti, spazio di archiviazione, immagini e template.

Caratteristiche del server utilizzato per i test:

- Dell Server PowerEdge R630.
- CPU: 2x Intel® Xeon® E5-2650 v3 @2.30GHz.
- Cores: 20 fisici, 40 hyper threading
- RAM: 160 GB.



Installazione Automatica

Creazione de file **Kickstart**:

- Fornire i **parametri di installazione** (lingua, tastiera, schema di partizione, etc…) che l'utente dovrebbe inserire manualmente.
- Installazione semplice di **gruppi** di **pacchetti software** (software per la virtualizzazione, ambiente grafico, etc…).
- È possibile indicare comandi come in uno script bash che verranno eseguiti al termine dell'installazione del sistema operativo.
- Un **kickstart generico** è disponibile online: l'utente lo può **scaricare e personalizzare** facilmente grazie ad uno script dedicato.
- Possibilità di installare il server principale oppure un host secondario da aggiungere ad una infrastruttura già esistente.

ks.cfg

Preparazione di un file **ISO** personalizzato:

- Si partire da un'**immagine standard** (CentOS-6.7-x86_64-netinstall.iso).
- Si monta l'immagine in una cartella locale per poterla modificare.
- Il file kickstart va copiato nella cartella locale ed il file **isolinux.cfg** deve essere appositamente modificato in modo che il kickstart venga trovato dopo il boot.
- A partire dalla cartella si genera un **nuovo file iso** e lo si scrive su un **dispositivo usb**.

ISO

L'**installazione richiede circa 1 ora** con:

- **Pochi parametri** lasciati da inserire all'utente:
  - configurazione di rete;
  - layout della tastiera.
- **Setup automatico** del gestore cloud:
  - OpenNebula;
  - squid proxy;
  - rOCCI.
- Setup automatico del worker node.
- File **iso personalizzati** per diversi utenti:
  - web repository per VMs e template;
  - **QCOW2**: utilizzo dinamico delle risorse.
- Installazione **minimale** del sistema operativo:
  - solo il software indispensabile viene installato.
- Contestualizzazione effettuata via **CVMFS**.
- **Test e validazione** dell'infrastruttura:
  - interfaccia Sunstone;
  - creazione VMs;
  - esecuzione di job;
  - sottomissione job via VMDIRAC .

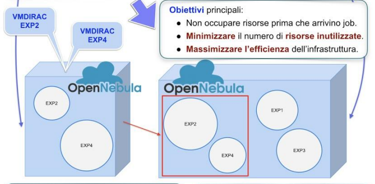
Elasticità

Approccio Modulare:

- Viene sfruttato il sistema di installazione automatica.
- **Doppia infrastruttura** basata su OpenNebula:
- **OpenNebula *Privata***: con cui interagisce VMDIRAC da remoto.
- **OpenNebula *Pubblica***: usata come estensione per quella privata.

Obiettivi principali:

- Non occupare risorse prima che arrivino job.
- **Minimizzare** il numero di **risorse inutilizzate**.
- **Massimizzare l'efficienza** dell'infrastruttura.



OpenNebula *Privata*:

- Il gestore ha **pieno controllo** dell'infrastruttura.
- **Diversi utenti** possono utilizzare le risorse (attraverso con più istanze di VMDIRAC).
- Agisce da singolo utente sull'infrastruttura pubblica.
- Introduce un **ulteriore** livello di **elasticità**.

ONE-cloudbursting-driver:

- È lo strumento sviluppato al JINR.
- Permette lo spostamento di macchine virtuali tra due infrastrutture basate su OpenNebula.

OpenNebula *Pubblica*:

- Il gestore dell'infrastruttura privata agisce come un utente generico.
- Il gestore dell'infrastruttura privata **non** ha il controllo dell'infrastruttura.

Applicazioni del tool di installazione automatica e dell'infrastruttura elastica

- L'approccio descritto permette un'installazione quasi **automatica** di una infrastruttura cloud **elastica**.
- Questo strumento va a beneficio di siti con
  - limitate conoscenze specifiche di cloud: l'installazione è **semplice** e relativamente **veloce**;
  - limitato personale dedicato: il **mantenimento** dell'infrastruttura richiede un **impegno ragionevole**.
- È possibile mettere a disposizione un numero anche limitato di risorse che sarebbero altrimenti inutilizzate.
- Nel contesto di un'**infrastruttura condivisa** da più utenti:
  - gestire **privatamente** le quote dei propri utenti: **utilizzo più efficiente** delle risorse;
  - interfacciarsi **elasticamente** con più infrastrutture pubbliche: maggiore **efficienza dell'infrastruttura**;
  - mantenere inalterata l'allocazione delle risorse per gli utenti sull'infrastruttura pubblica.
- L'infrastruttura privata è **trasparente** a quella pubblica:
  - OpenNebula pubblica non sa dell'interazione della privata con VMDIRAC;
  - OpenNebula privata è come un utente qualsiasi con una certa quota di risorse a disposizione sull'infrastruttura pubblica;
  - La gestione delle quote tra utenti di OpenNebula privata è nascosta a quella pubblica.

TRIESTE XVI Incontri di Fisica delle Alte Energie IFAE 2017

IFAE2017
XVI Incontri di Fisica delle Alte Energie
19-21 Aprile 2017 - Trieste.

Il progetto BESIIICGEM è stato finanziato dalla Commissione Europea all'interno della call H2020-MSCA-RISE-2014 HORIZON 2020

The tool has been presented in a Poster at IFAE2017.

The contribution and the full-size pdf of the poster can be found here.

Special thanks to Marco, Flavio, Antonio and Marco Maggiora for the support during the development of the tool.

Test

- Current Tests
- Running Instances

Test

An installation and operation guide has been published on the Cloud Toy web page.

- The tool has been tested several times in Torino.
- Belle II groups are currently performing beta testing as well.
- The submission via VMDirac is under test on the Turin infrastructure.
- BesIII groups may contribute as testers in the future to test elasticity.

Test

The tool is adopted in Torino to quickly set up small cloud infrastructures to provide computational power for students:

- Three master students are currently working with several VMs hosted by two Cloud Toy infrastructures.
- This kind of test helps for spotting any installation error or misconfiguration.

See the cloud section of my webpage

Conclusion

Conclusion

The automatic installation tool is ready and is under test.

To Do:

- implement elasticity;
- create and test the software repository;
- add BesIII groups as testers.

… and keep testing the tool in order to improve it.

Question Time

Automatic set-up - kickstart.cfg

A **kickstart** file contains the information needed to perform the installation in order to **avoid the user providing them**.

Some parameters may be left to user input (network parameters, keyboard layout, ...).

It is possible to specify which **packages or additional software** have to be installed.

Software installed via kickstart: OpenNebula, squid proxy, and rOCCI.

Automatic set-up - customized ISO

Starting from a kickstart and a standard iso it is possible to modify the iso so that it will look for the given kickstart at installation time.

The standard iso adopted during the test are:

- ➔ **CentOS- 6.7**-x86_64-netinstall.iso
- ➔ **CentOS-7**-x86_64-NetInstall-1511.iso

Pay attention to the location of the kickstart: a good idea is to refer to the usb drive using its “label”.

Automatic set-up - bootable usb drive

Format (FAT32, mbr) a usb drive (2GB is enough).

It's important to give it the **name** indicated in the customized iso creation process. Otherwise the kickstart will not be found when the installation begins.

Make the usb bootable either via command line or using an application such as Unetbootin.

Automatic set-up - install on server

Start or reboot the machine and plug in the bootable usb drive, choose the **boot from usb drive** option.

The installation will ask the user to provide the parameters not specified within the kickstart file, then the installation will proceed autonomously. For instance, the network parameters have been left to the user input.

The machine will reboot when the installation is over. Remove the usb drive and let the machine boot normally.

Automatic set-up - test the installation

Once the machine is up and running, users may control that the parameters have been properly set (network interface is up, disks are mounted, ...).

To verify that OpenNebula is working check the hypervisor status via the Sunstone interface.

The entire installation process requires about 1 hour (on the hardware used for the test).

Automatic set-up - test the installation

Test the installation running a customized VM to check that all the installed software run properly.

Create a **new image** in OpenNebula: a file .one will be provided with the proper path.

Create a **new template** in OpenNebula: a file .txt will be provided, insert image and network number.

Instantiate a new VM: the machine is ready in less than 1 minute and can be used, software mounted via CVMFS used in combination with the squid proxy.

Automatic set-up - about VMs

There are no constraints on OS: SL5, SL6, Ubuntu 14, ...

Strict requirements on **storage space**: the VM image has to be copied over the net each time a new VM is instantiated. It could be a bottleneck.

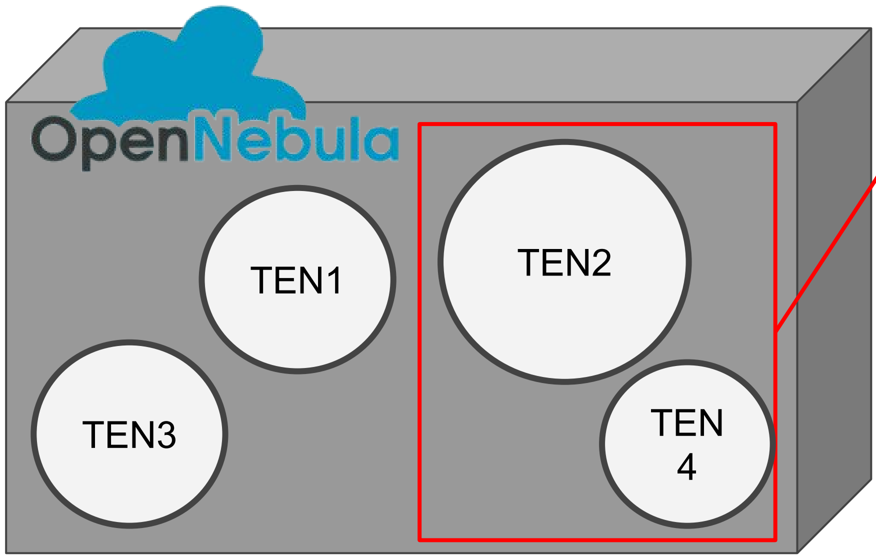
- **QCOW2** image format: Dynamic increase of the storage
- **Minimal OS** installation: Only the required software is installed

Elasticity of the infrastructure

Elasticity of the infrastructure

Aim: provide inter-experiment elasticity.

The CI is used by different stakeholders, the aim is to dynamically change quotas



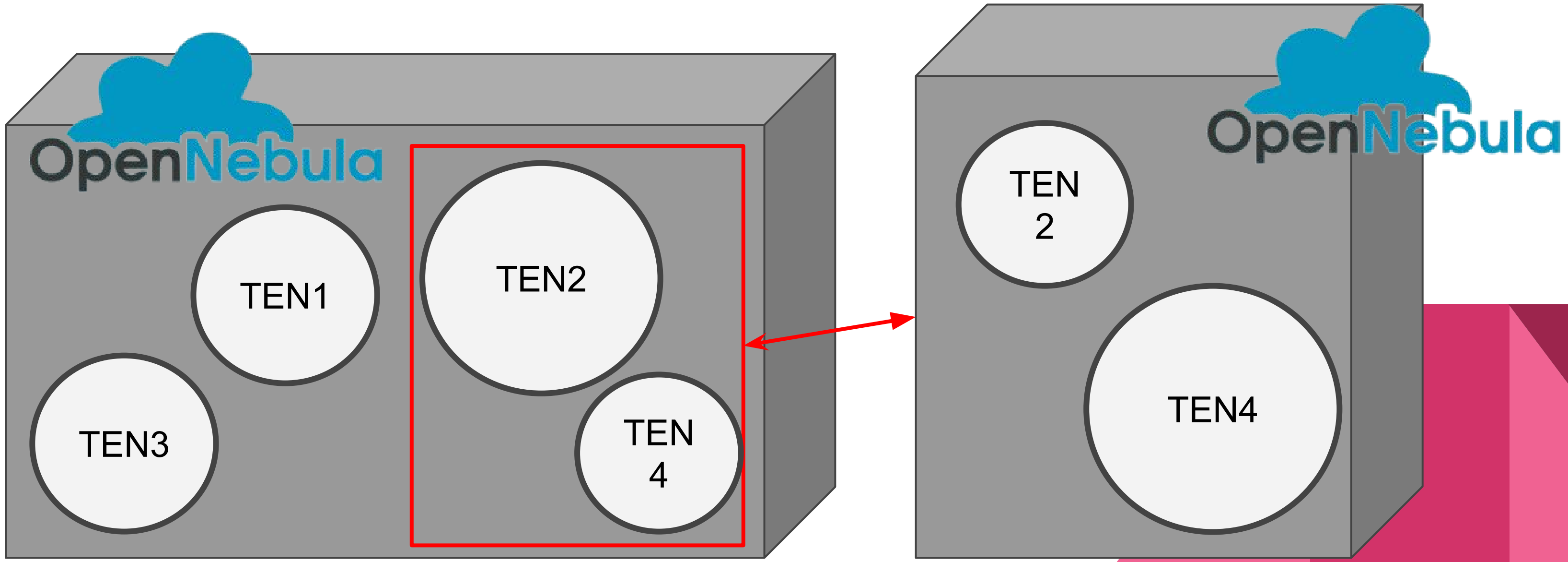
TEN2 has 100 cores
TEN4 has 200 cores
Together they have 300

The aim is to manage these 300 cores dynamically for TEN2 and TEN4

Elasticity of the infrastructure

Directly change quotas may not be safe (different stakeholders are involved)

A “private” OpenNebula is introduced



Elasticity of the infrastructure

The “public” OpenNebula only knows that TEN2 and TEN4 have 300 cores (user-level control).

The “private” OpenNebula can dynamically change quotas between TEN2 and TEN4 according to actual needs (full sys-man control).

Variations of quotas can be performed via **oneadmin.**

Elasticity of the infrastructure

A script has been written to perform the quota variation

It takes as arguments:

- $1 the user
- $2 the new value of CPU for it

```
#!/bin/bash
echo "Changing CPU Quotas of user: $1. New value: $2"
#1)
sleep 100

#2)
touch ./tempQuotas

#3)
cat > tempQuotas << EOF
VM = [
 VMS           = "-1",
 MEMORY        = "-1",
 CPU           = "$2",
 VOLATILE_SIZE = "-1"
]
EOF

cat tempQuotas

#4)
oneuser quota $1 ./tempQuotas
rm -f tempQuotas
echo "done.”
```

Elasticity of the infrastructure

#1) wait for 100ms

#2) create a new temporary file

#3) writes the proper commands into the file and prints them

#4) perform the changes

```
#!/bin/bash
echo "Changing CPU Quotas of user: $1. New value: $2"
#1)
sleep 100

#2)
touch ./tempQuotas

#3)
cat > tempQuotas << EOF
VM = [
 VMS           = "-1",
 MEMORY        = "-1",
 CPU           = "$2",
 VOLATILE_SIZE = "-1"
]
EOF

cat tempQuotas

#4)
oneuser quota $1 ./tempQuotas
rm -f tempQuotas
echo "done.”
```

Elasticity of the infrastructure

So far both the “private” and “public” OpenNebula have been set-up for testing purposes.

Interaction between them is a work-in-progress:

> The **ONE-cloudbursting-driver** provided by N. Balashov (JINR), is currently under test.

Elasticity of the infrastructure

The **ONE-cloudbursting-driver** is currently under test.

It enables OpenNebula-based cloud to "burst" Virtual Machines (VM) to external OpenNebula clouds using built-in OpenNebula XML-RPC and OCCI interfaces.

Aim: **start VMs on Public OpenNebula from the Private OpenNebula**