THE ITALIAN RESEARCH & EDUCATION NETWORK

Consortium GARR

OpenStack automation at GARR using MAAS and Juju (plus some off-topics)

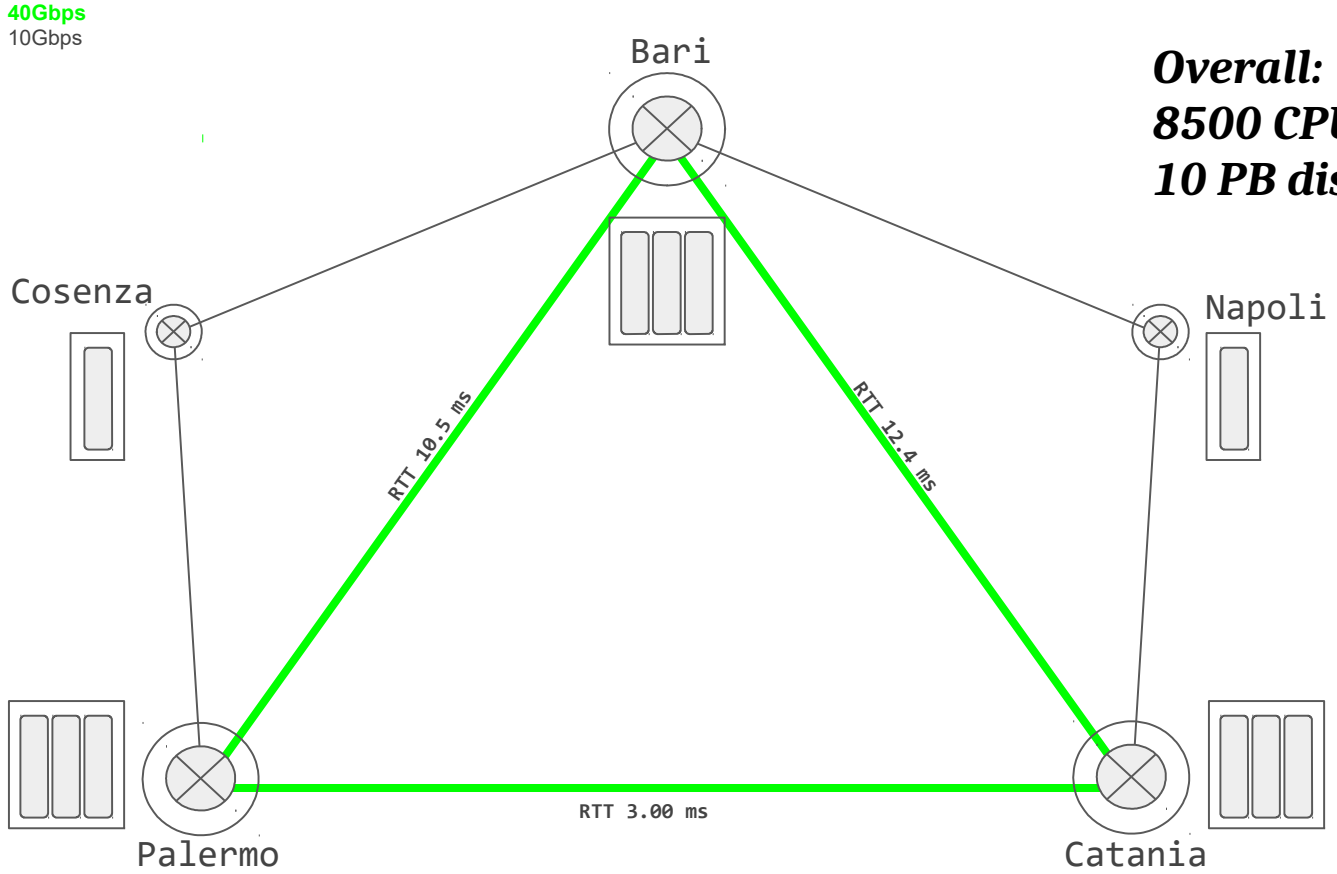
Amsterdam 2017-09-26
Fulvio Galeazzi (fulvio.galeazzi@garr.it, GARR dipartimento CSD)

Outline

- GARR computing infrastructure
- Our role, goals, requirements
- Our solution for automation
  - MAAS
  - Juju
  - More on charms and bundles
- Garr cloud status
  - Federation
- Off-topic 1: Virtual Data Centre
- Off-topic 2: federated Authentication and Authorization

GARR computing infrastructure (1)

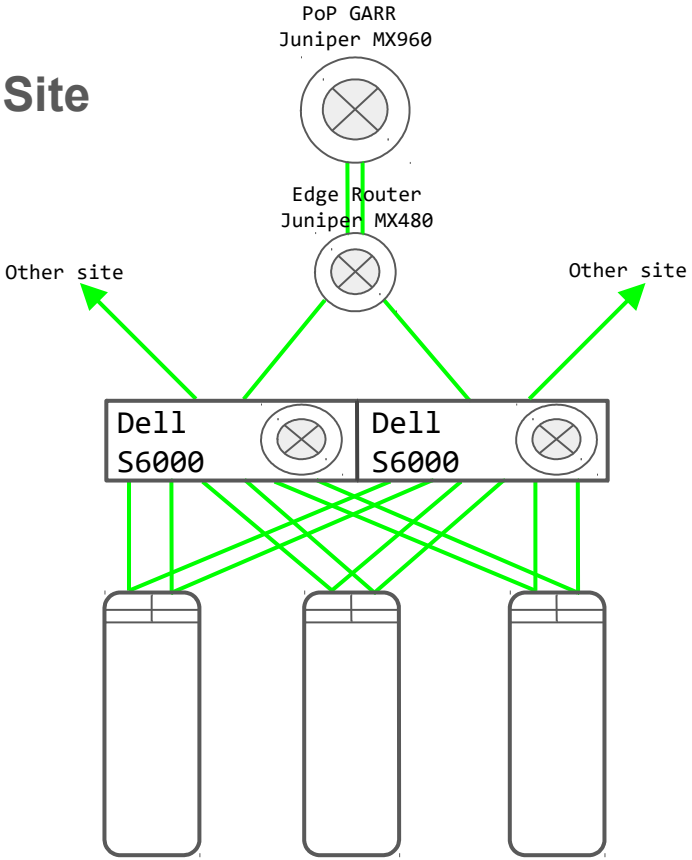
All sites



Overall:
8500 CPU cores
10 PB disk

GARR computing infrastructure (2)

Site



Data Network 40 Gbps

- 2 Switch ToR Dell MXL in each module-CSD
- 2 star Router/Switch Dell S6000
  - 32 ports x 40 Gbps

Managment network (“ILO”) separated from data

- 2 Switch management ToR Dell S55 each module-CSD
- 2 Switch management star Dell S4810
  - 48 ports x 1 Gbps

GARR computing infrastructure (3)

Chassis Blade **Dell M1000e:**

- 16 server (blades) Dell Poweredge M620
- 2 integrated Ethernet (Dell MXL)
  - 2x16 porte 10 Gbps -> blades
  - 4 uplink 40 Gbps -> main site switch;
- 2 Fibre Channel switches (Brocade M6505)
  - 16 ports (16 Gbps) to blades
  - 8 uplink (16 Gbps) to storage controller;

2 Storage Array **MD3860f FC:**

- SAS disks 116x4TB + 4xSSD 1.6TB
- FiberChannel Brocade controller 2x16 Gbps (2x4 ports)

“CSD Module”

OpenStack: “simplified” view

- Lots of components
- Lots of in-depth expertise needed
- Need to automate as much as possible
  - It WOULD still be painful, we knew!

Dashboard: **Horizon** or CLI

Nova

Nova API
Scheduler
Conductor
Queue
Nova db
Nova compute
Hypervisor
VM
Network

Keystone

Keystone API
Keystone db

Cinder

Queue
Cinder db
Cinder Vol
Cinder API
Scheduler
Cinder bck

Neutron

Queue
Neutron db
Neutron API
Scheduler
Plugin/agent

Glance

Glance API
Glance db
Glance Registry

Block Storage

storage

Network Node

DHCP/IPAM
Router/GW

Ceilometer

Ceilometer API
Agent
Collector

Swift

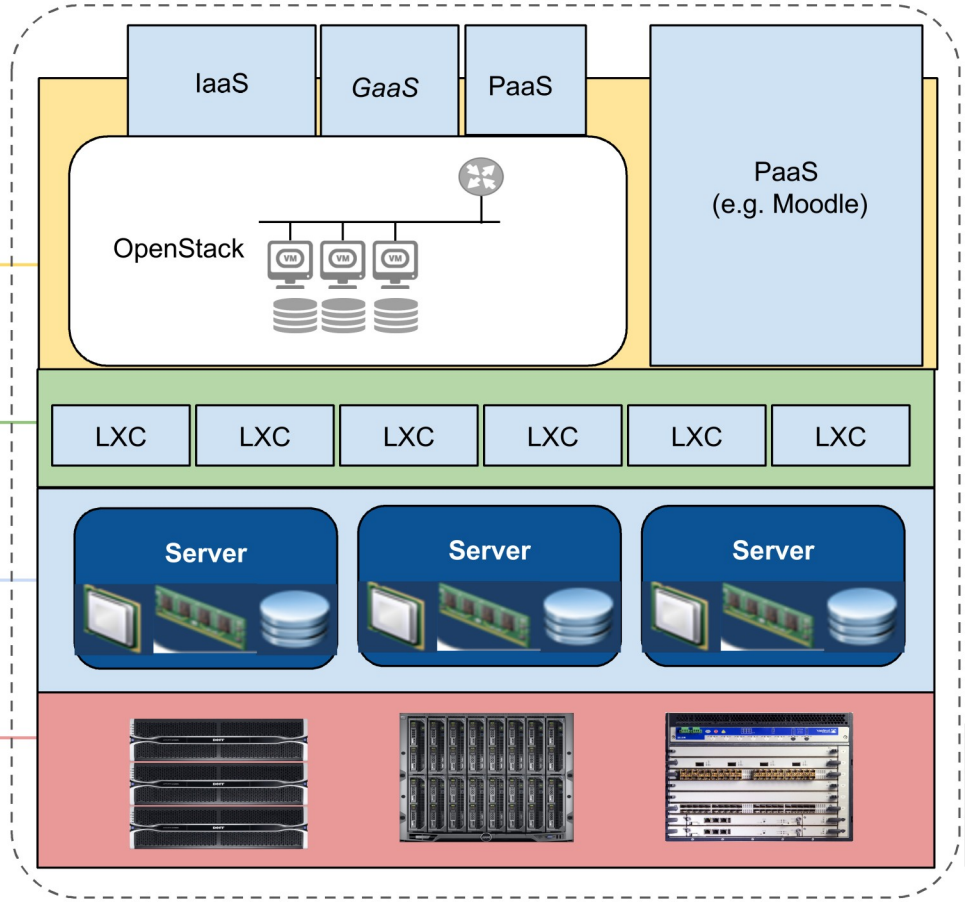
Proxy server
Object Store

Our role, goals and requirements

- Role:
  - Act both as a resource aggregator (federation) and as a provider of computing resources (“long tail of science”)
- Goals
  - simplify provisioning of storage and computing services
  - serve different organizations
  - unified access (SSO)
  - Empower users with something more than a PAAS and something easier than a IAAS
- Requirements
  - open-source
  - reduced manpower *efforts*
  - sharing resources
  - always on
  - replicable and scalable
  - *self* deploying and *self* healing
  - elastic
  - separation / flexible security policies

4 Layers

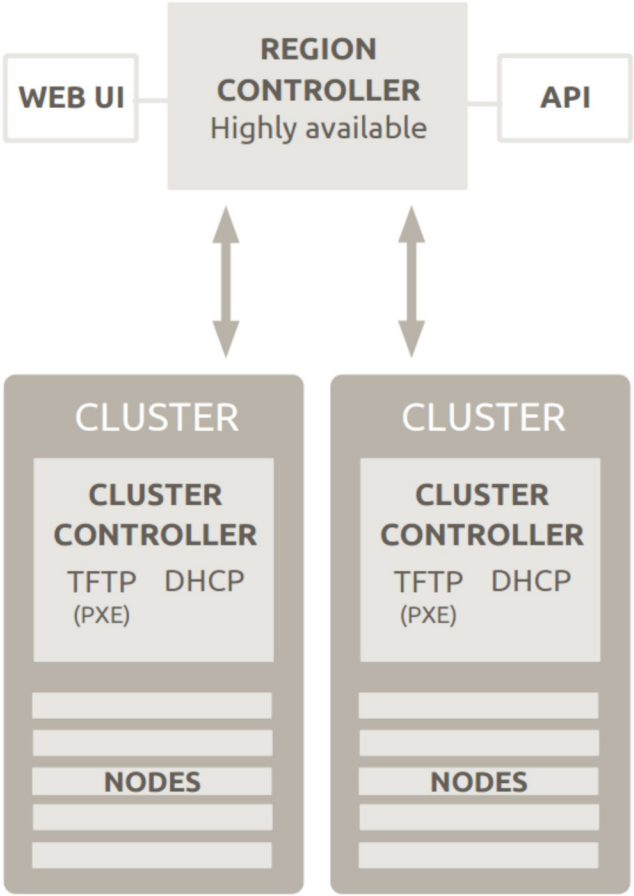
1. Application Services
2. Infrastructure *Virtualization*
3. Operating System
4. Physical resources



Can we automate them all?

Metal As A Service

- Discover, commission and deploy **physical servers**
- **Allocate** physical resources to match workload requirements.
- **Retire servers when they are no longer needed** and make them available for new workloads as required.
- **Cross datacenters** provisioning



Rapid provisioning at cloud scale

Images Courtesy of CANONICAL

3-step provisioning process

1

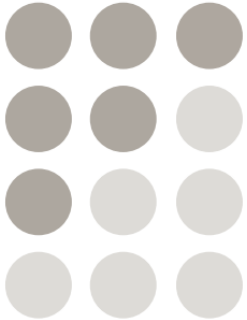
Install MAAS on first server

2

Discover Nodes

Automatically discover nodes
Enlist nodes via PXE boot
or manually enter MAC addresses

3

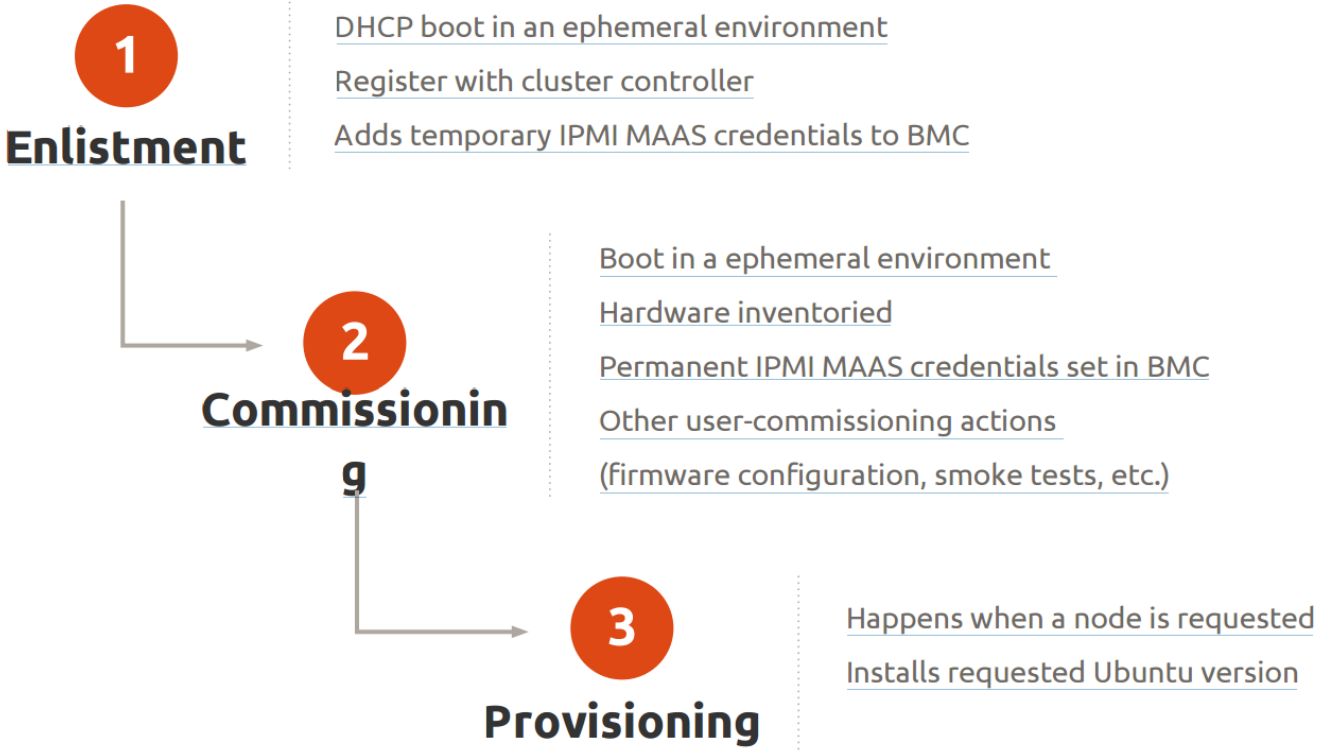


Power on Nodes

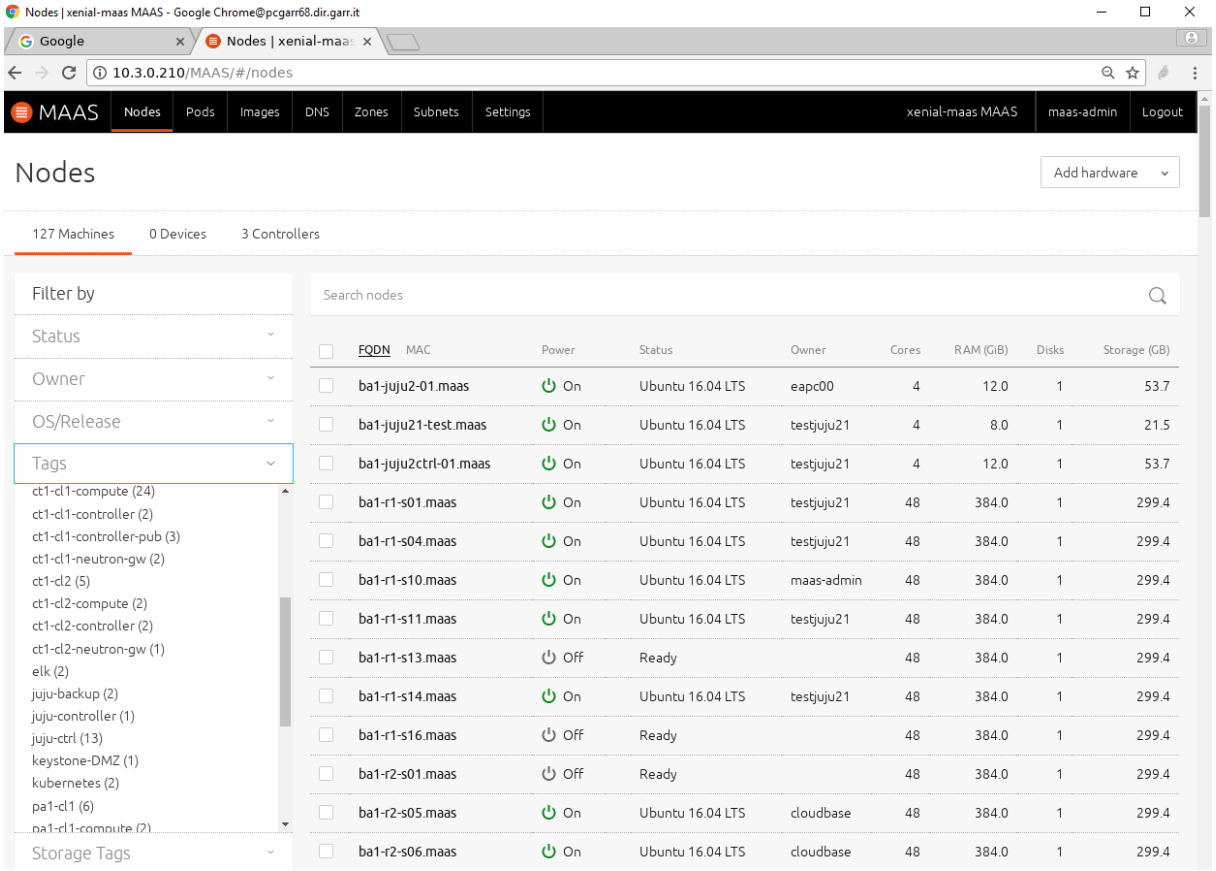
Hypervisor or OS
provisioned automatically

Hardware provisioning workflow

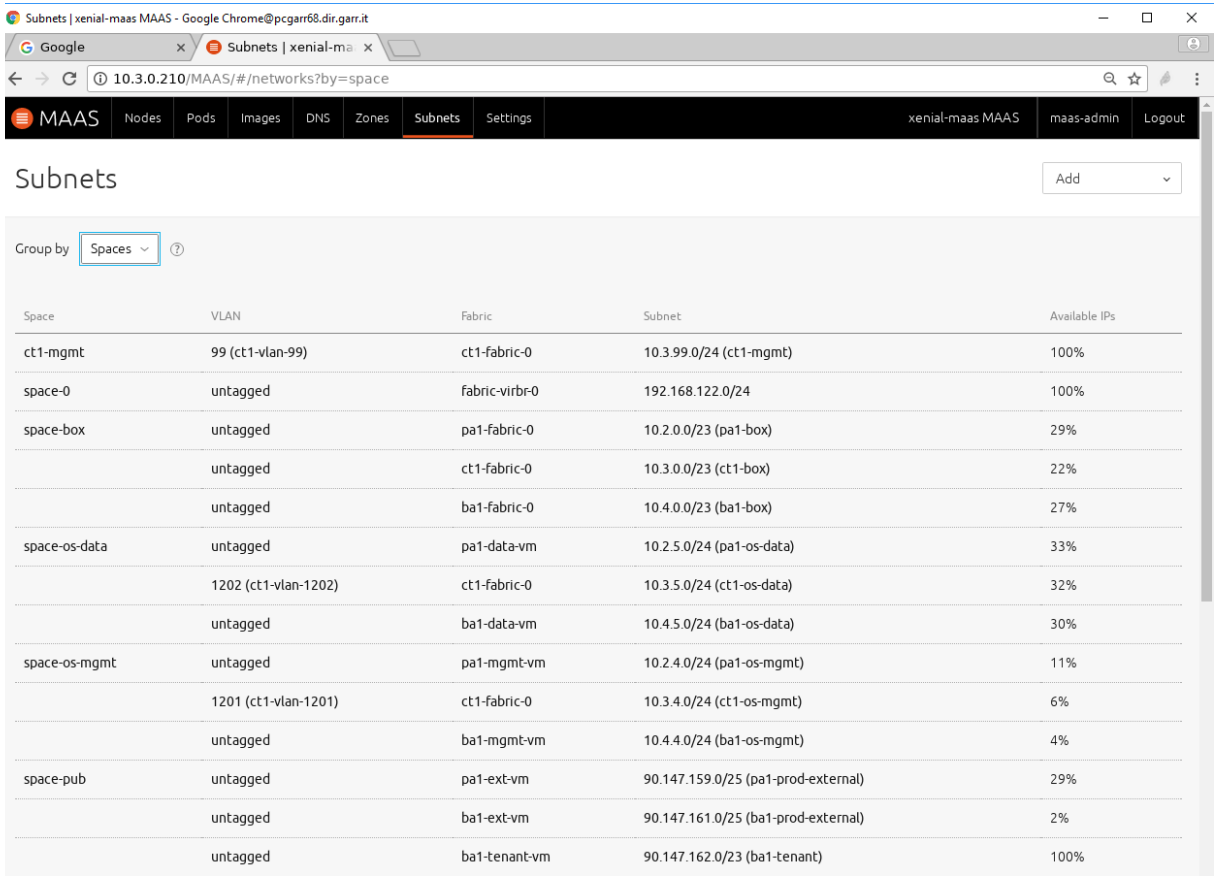
Courtesy of CANONICAL



MAAS host inventory

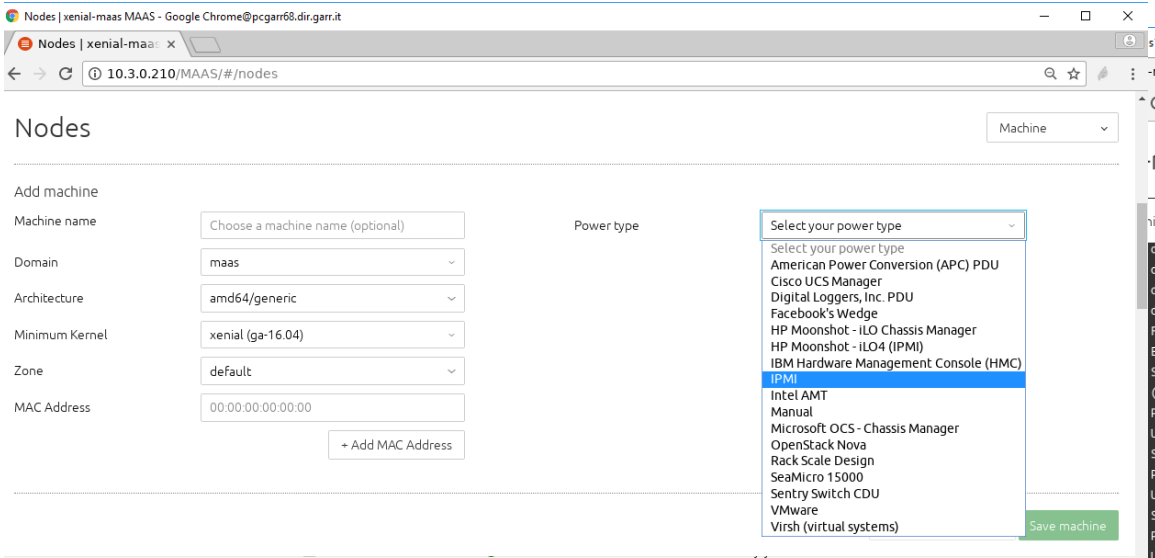


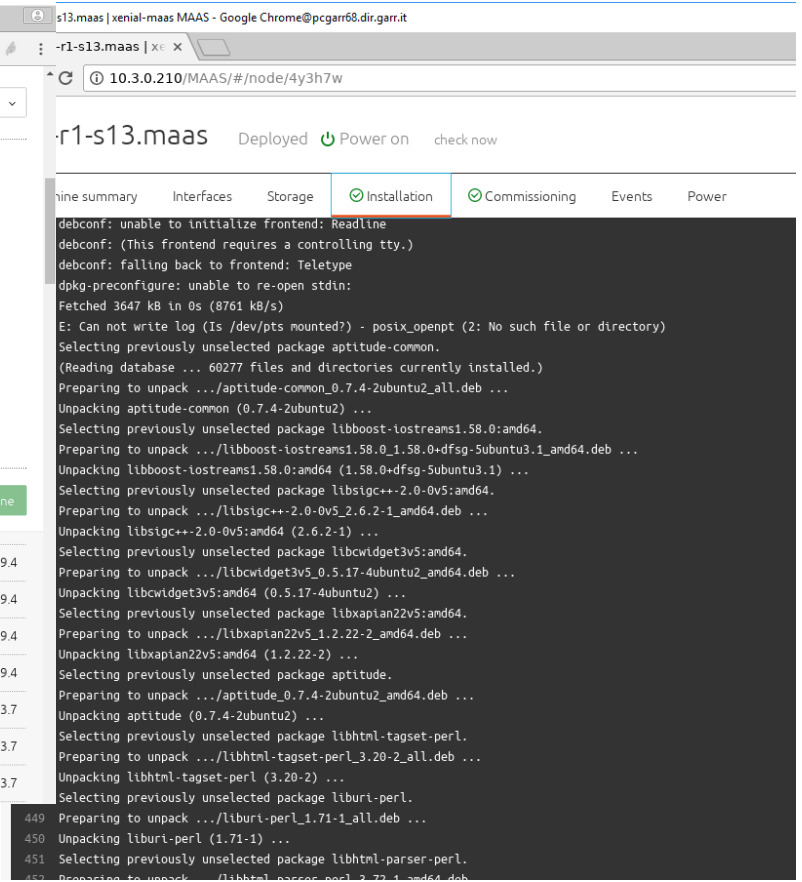
MAAS network inventory



Space	VLAN	Fabric	Subnet	Available IPs
ct1-mgmt	99 (ct1-vlan-99)	ct1-fabric-0	10.3.99.0/24 (ct1-mgmt)	100%
space-0	untagged	fabric-virbr-0	192.168.122.0/24	100%
space-box	untagged	pa1-fabric-0	10.2.0.0/23 (pa1-box)	29%
	untagged	ct1-fabric-0	10.3.0.0/23 (ct1-box)	22%
	untagged	ba1-fabric-0	10.4.0.0/23 (ba1-box)	27%
space-os-data	untagged	pa1-data-vm	10.2.5.0/24 (pa1-os-data)	33%
	1202 (ct1-vlan-1202)	ct1-fabric-0	10.3.5.0/24 (ct1-os-data)	32%
	untagged	ba1-data-vm	10.4.5.0/24 (ba1-os-data)	30%
space-os-mgmt	untagged	pa1-mgmt-vm	10.2.4.0/24 (pa1-os-mgmt)	11%
	1201 (ct1-vlan-1201)	ct1-fabric-0	10.3.4.0/24 (ct1-os-mgmt)	6%
	untagged	ba1-mgmt-vm	10.4.4.0/24 (ba1-os-mgmt)	4%
space-pub	untagged	pa1-ext-vm	90.147.159.0/25 (pa1-prod-external)	29%
	untagged	ba1-ext-vm	90.147.161.0/25 (ba1-prod-external)	2%
	untagged	ba1-tenant-vm	90.147.162.0/23 (ba1-tenant)	100%

MAAS new node and node check





Juju

Juju allows **configuring, managing, maintaining, deploying and scaling** cloud services (workloads) quickly and efficiently on multiple providers:

- private or public clouds
- bare metal, **leveraging MAAS to control the hardware.**

Juju uses descriptions of services called **Charms** which specify how to deploy a service, how to interact with other charms ("relations") and how to react to changes (e.g., configuration parameters).

Juju can manage and scale models consisting of many charms, creating complex architectures, like an OpenStack cluster.

Juju can be controlled via a **web GUI, the command line, or API.**

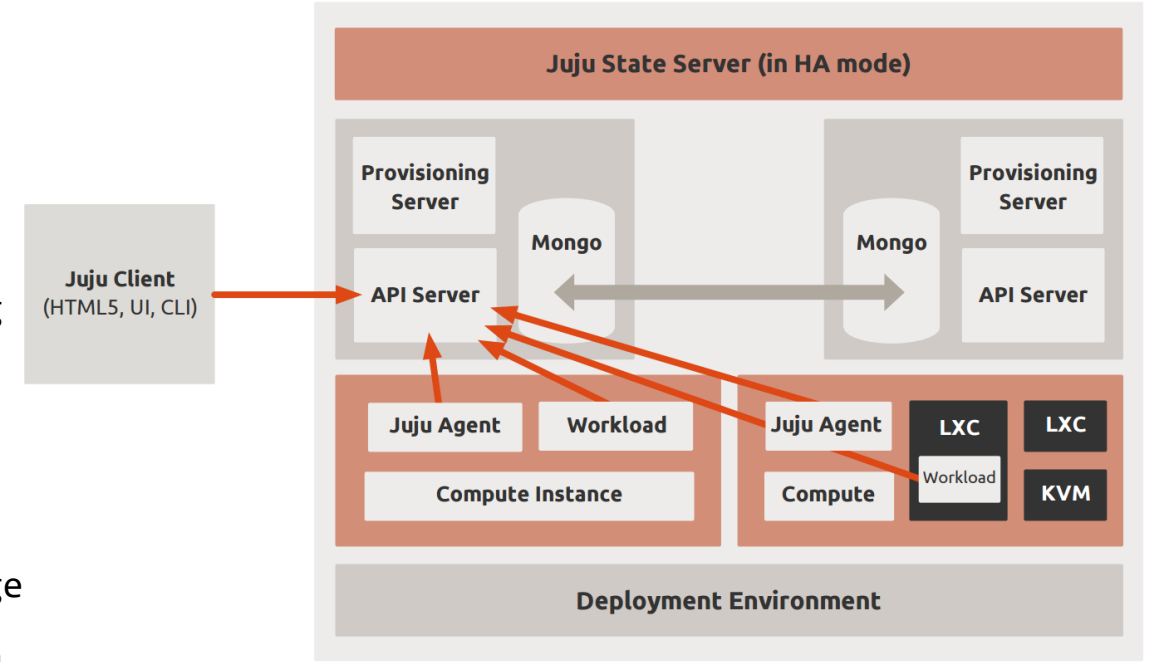
Controller: the management node of a Juju cloud environment; manages the database and the API server; can host multiple models.

Model: an environment associated with a controller; it is associated to specific credentials for the cloud environment.

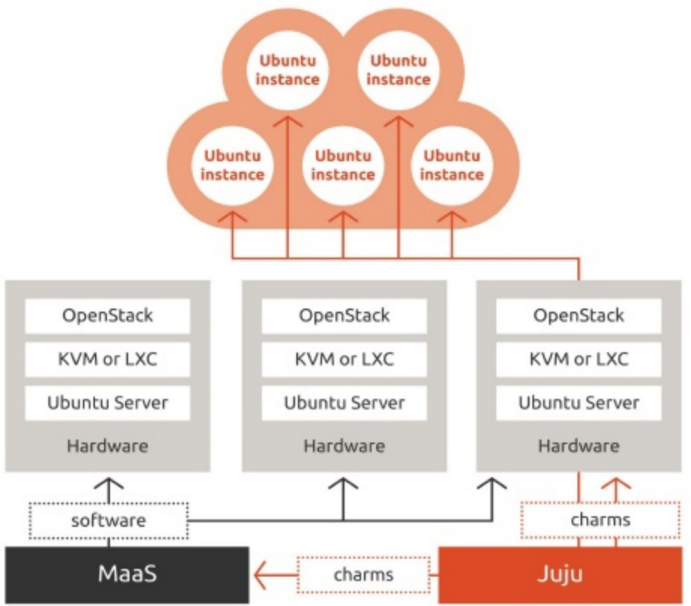
User: this is a Juju user, with some rights on the model; will be mapped to a Unix user on the Juju-CLI machine.

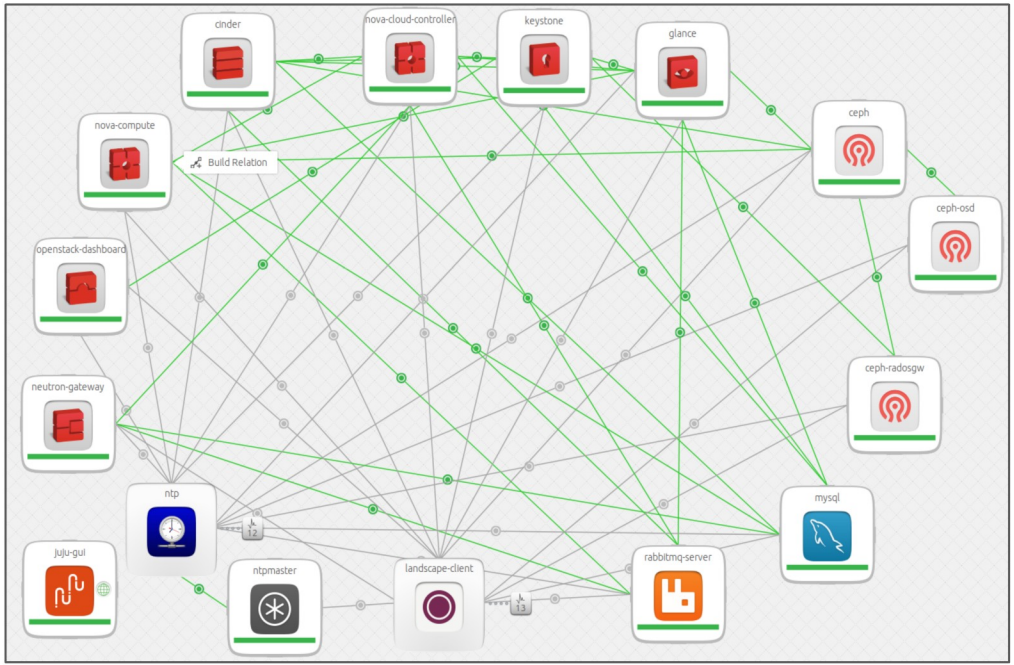
Juju architecture

- **Ease of provisioning:** from local machines to large clouds
- **Event-based** Reacts to changes in environment, self configuring
- **Scalable Templates** designed to scale by adding more units
- **Language independence** Hooks can be written in any language
- **In our env**: one controller on MAAS cloud to deploy O~S, and one controller on O~S cloud to be available as-a-service



OpenStack as (one) orchestrated service





More about charms & bundles (1)

Example add Nagios to running OpenStack cluster:

```
juju deploy nagios nagios-server-ct1-cl2 --constraints spaces=space-box,space-os-mgmt,space-pub
--bind 'space-os-mgmt website=space-pub' --to lxd:182
juju config nagios-server-ct1-cl2 password=********
...
juju deploy nrpe nrpe-cinder-ct1-cl2
juju add-relation cinder-ct1-cl2 nrpe-cinder-ct1-cl2
juju add-relation cinder-hacluster-ct1-cl2 nrpe-cinder-ct1-cl2
juju add-relation nrpe-cinder-ct1-cl2:monitors nagios-server-ct1-cl2:monitors
juju config nrpe-cinder-ct1-cl2 swap='-w 40% -c 25% -n ok'
juju config nrpe-cinder-ct1-cl2...
```

More about charms & bundles (2)

More than 200 “official” charms currently available in the Juju charm store at http://jujucharms.org/

Don’t like what a charm is doing, or cannot find what you’re looking for?

- `charm pull <charm_name>` and examine it,
  - Submit a patch
  - Fork and modify
- Write your own
  - Templates and documentation are available
  - Can be as easy as a Bash script, or more complex (Python with hook handlers)

Completeness and usefulness of the catalogue depends on reaching a “Critical mass” of users… anyone interested, here?

GARR OpenStack cluster (and federation)

(Almost) entirely built on Juju/MAAS:

- With the exception of Ceph cluster, although charms for Ceph do exist, but:
  - Wanted to keep OpenStack and Ceph development cycles independent
  - Have the same Ceph cluster serve prod/test/dev
- **Openstack**
  - Release Mitaka, upgrading to Newton and soon to Ocata

Base services:

- **Global** (**3 sites replica** - common to whole federation)
  - Identity service / Keystone
  - Image service / Glance
  - Object Storage / rados gw
- **Local** (**3 racks replica (GARR region)** - specific for the region)
  - Controller service / Nova
  - Network service / Neutron
  - Block Storage / Ceph

Each site is an Openstack **Region**: currently 3 GARR regions, with 2 “external” ones joining

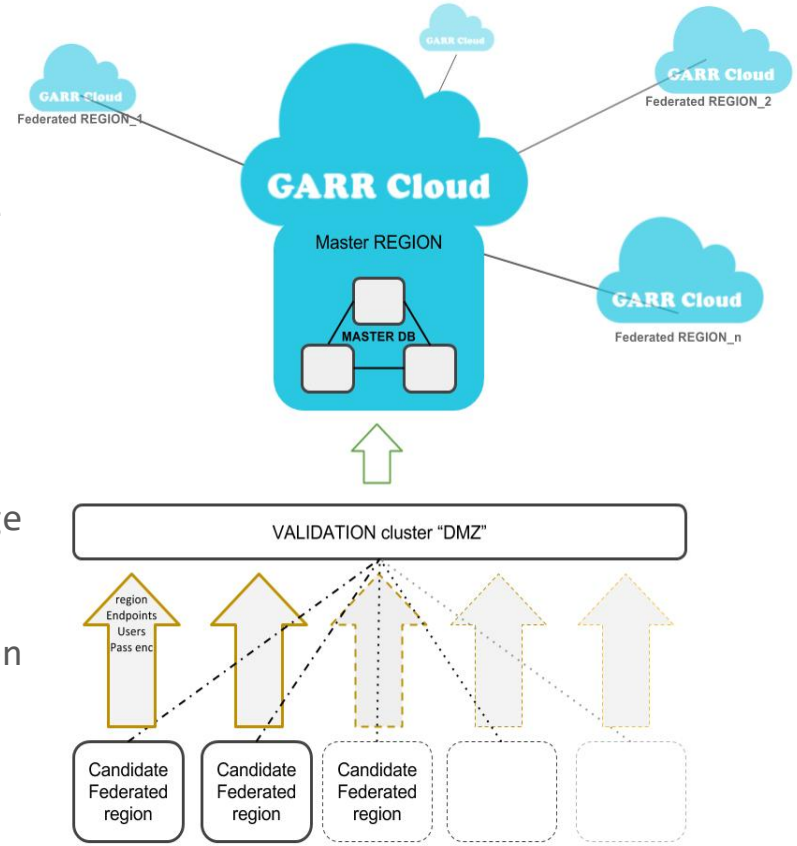
Joining the Federation

Procedure of inclusion of a Juju-managed cluster

- Bundle O~S attaches to validation cluster
- Validation in “DMZ” cluster
- No cleartext credentials exchange

Different contribution options:

1. You own HW, but have no manpower/knowledge
2. You already have an O~S deployment
3. None of the previous, but you have (wo)men



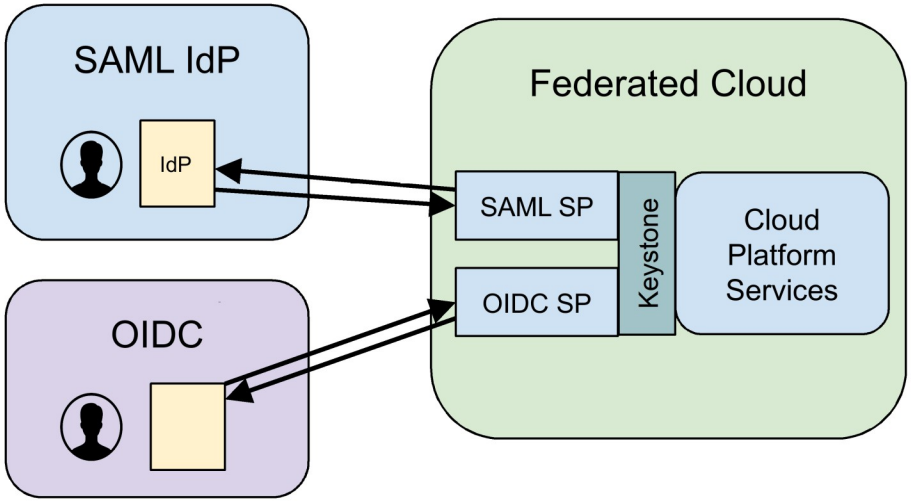
Off-topic 1:
Virtual Data Center

Aim: delegate administration workload to vDC admins

- **Cloud admins** create “parent” project with agreed total resources (CPU, RAM, storage…)
- **vDC admins**
  - **create “child”** projects (limited by the quotas set on parent)
  - **assign users** to child projects
  - can **delegate administration** of parts of the project tree
- Required tweaking of policy.json files

Off-topic 2:
Federated authentication/authorization

1. **Separation of roles**: cloud administrator and the domain administrators.
2. The federated Identity providers are **delegated** only for **authentication**
3. **No authorization information stored outside of keystone**, in order to avoid:
    a. Having to check reliability and consistency of such information
    b. Having to map it to internal keystone entities
    c. Force users to act on an IdP not under their personal control
4. **Users can be granted rights on any project** of the federation, irrespective of their affiliation and under the sole control of the administrator for that project
5. Deploy the simplest solution, relying **as much as possible on native OpenStack** capabilities avoiding any extra non necessary component.



Available in Juju charm store:
~cs:csd-garr/keystone-fed