Sync & Share on the INFN Corporate Cloud

Marica Antonacci (INFN-BARI)

on behalf of the *INFN-CC Team*

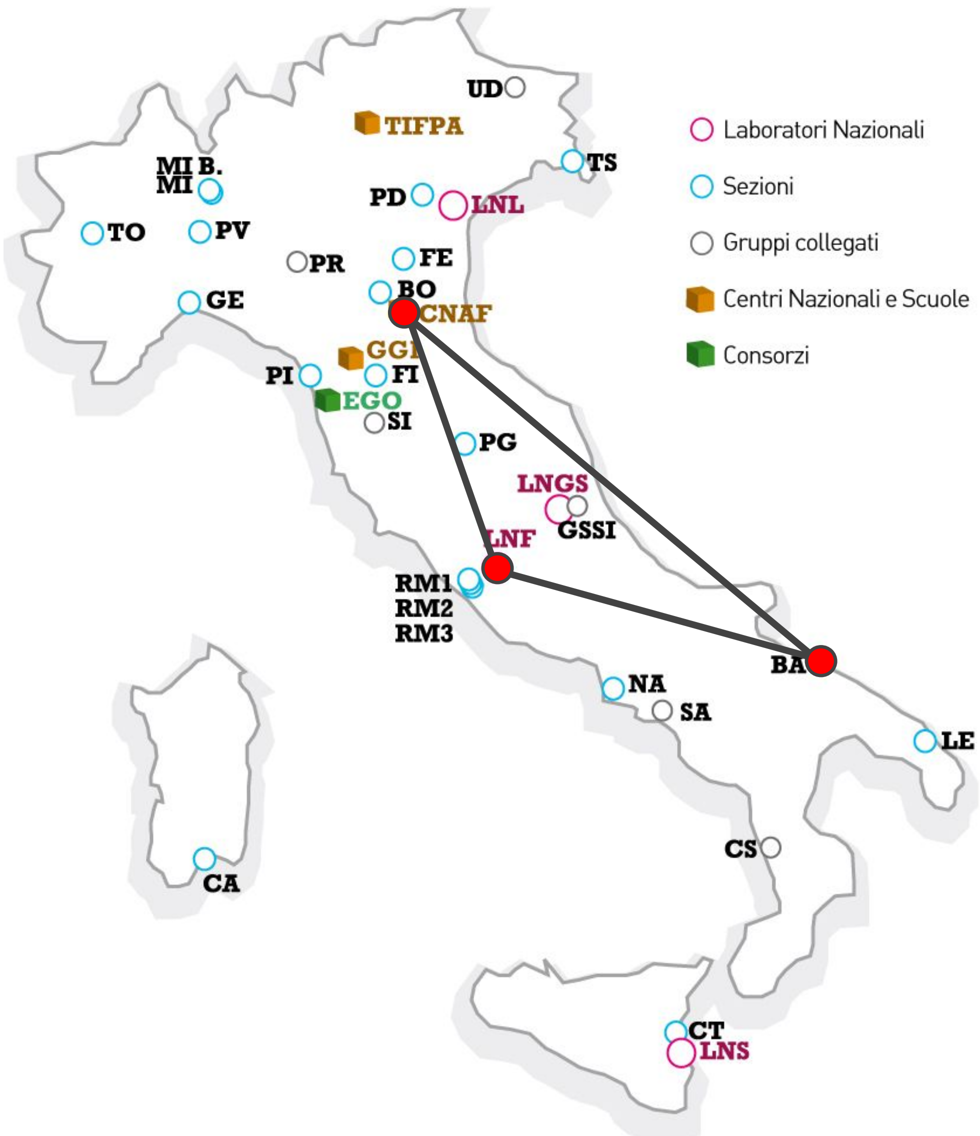
Outline

- INFN Corporate Cloud (CC) Overview
- INFN CC Architecture
- Storage solutions for:
  - Sharing virtual images
  - Synchronizing virtual devices
- Experimenting solutions for reducing the storage costs
- Sync&Share platform as a Service

INFN CC

An OpenStack-based geographically distributed cloud platform for the INFN community

- Key features:
  - Fully automated configuration and management (Foreman+Puppet)
  - IaaS/PaaS services
  - Backup & Disaster recovery
- A good solution for
  - running mission-critical workloads
  - promoting collaboration among INFN departments



Platform Architecture

Horizon Dashboard

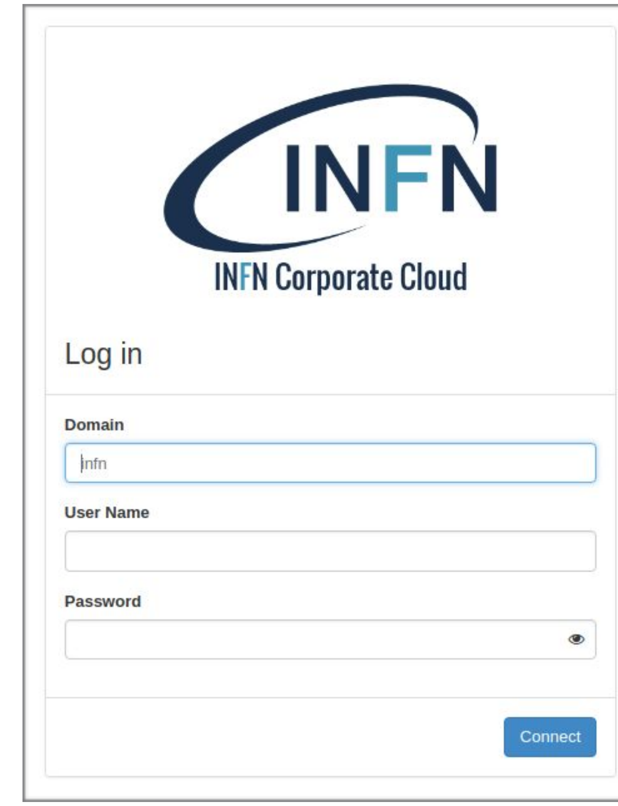
Keystone Identity Service

Glance Image Repository Service

Swift Object Storage Service

REGION: BARI	REGION: LNF	REGION: CNAF
Nova	Nova	Nova
Cinder	Cinder	Cinder
Neutron	Neutron	Neutron
Heat	Heat	Heat

Percona XtraDB Cluster



A dedicated WAN for management and storage

INFN

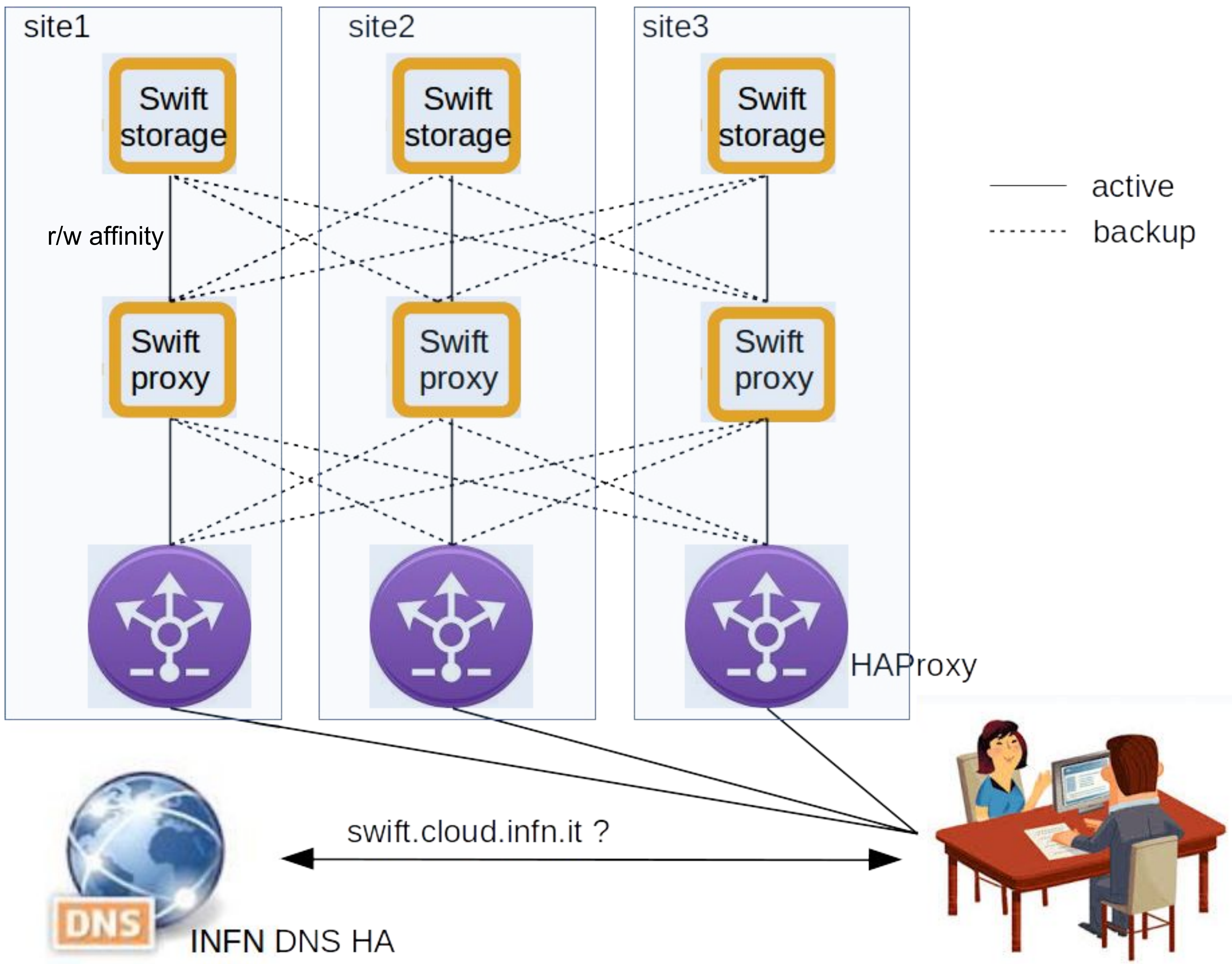
Load balancing and high-availability

- Each site exposes the public services endpoints through a LoadBalancer (HAProxy) that also provides SSL termination
- For the global services the LB redirects the traffic to the local services or, in case of failure, to the remote sites (backup)
- A **distributed DNS** is used to dynamically modify the DNS records so that the global names (e.g. swift.cloud.infn.it) point to the healthy endpoints
- It is based on **PowerDNS**: a monitoring probe updates the DNS entries using PowerDNS REST APIs

Sharing virtual appliances (and more..)

- **Swift Object Storage with 3 Regions**
  - 1 proxy node, 2 storage nodes (currently) per site.
  - RW affinity configured on the proxies in order to “prefer” the local storage servers
- Mixed SSDs and HDDs
  - *account* and *container* databases use SSD for better performance
- **Swift is used as backend for the Virtual Images repository in order to share the same images on all the sites**
  - VM snapshots are replicated as well
    - In case of a site failure, the remote snapshot replica can be used to start the VM on another site —> ***“cold”* migration**

Swift Object Storage



Ceph Distributed Storage setup

- **3 Clusters** - 1 for each site
  - Network latency is critical
- **Luminous** version (August 2017) with ***Bluestore backend*** with

  WAL and DB devices on SSD

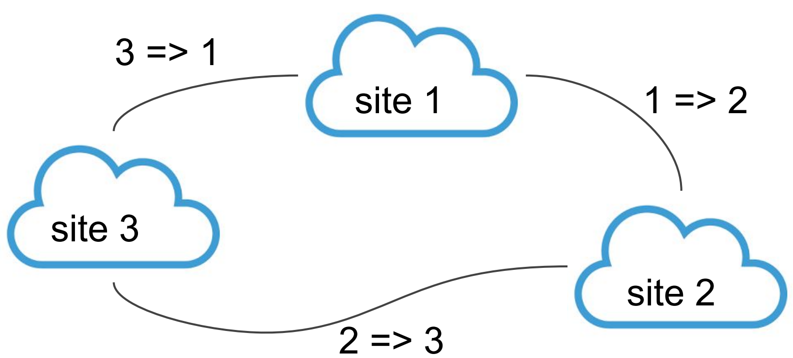
Hardware Characteristics of the storage nodes:
SGG- 6048 / File Server Supermicro RackMount 4U 24+2 Bay:
- 2 CPUs, Intel Xeon E5-2609v4
- 64Gb RAM DDR4-1200
- 12 x Toshiba 6TB SATA 6Gb/s 7.2Krpm 128M
- 2 x Intel S3520 150GB, SATA 6Gb/s, 3D MLC 2.5" (for the OS)
- 6 x Intel S3520 480GB SATA 6Gb/s, 3D MLC 2.5"
- Rear 2 x 2.5" HDD Kit

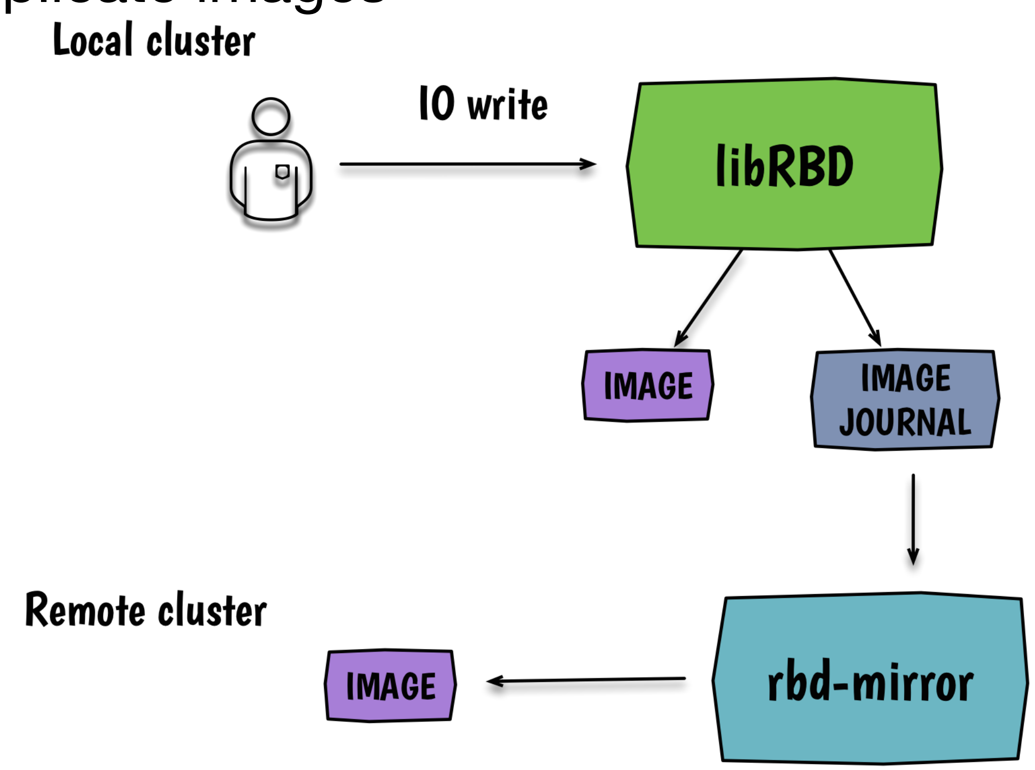
- **RBD** is used to:
  - host the **running Virtual Machine disks**
    - Ensuring the live-migration
  - provision **virtual disk devices**

STRONG CONSISTENCY REQUIRED!

Data replication

- **RBD asynch replication** (available since Jewel) relies on two new image features:
    - *journaling*: enables journaling for every transaction on the image
    - *mirroring*: tells the rbd-mirror daemon to replicate images
  - Can be enabled per pool or image
  - Our configuration:





- **Use case: Disaster recovery**
  - In case of failure, any site can recover its data from another site

Reducing the storage costs

- Replica 3 is a typical configuration
  - ~33% of the raw capacity is usable!
- Ceph **Erasure Coding** allows to achieve greater usable capacity

```
n = k + m  where ,
k  =  The number of chunks original data divided into.
m =  The extra codes added to original data chunks to provide data protection.
n  =  The total number of chunks created after erasure coding process.
```

- 4+2 configuration ensures 66% usable capacity and allows for 2 OSD failures
- We are evaluating different configuration options (erasure coding vs replication) comparing performances, data durability and availability

INFN

Volume quality of service

- **Geo-replicated**
  - HDD pool, mirrored
- **High Performance and replicated**
  - SSD pool, mirrored
  - E.g. for DB for mission critical application
- **Locally replicated**
  - Data Loss in case of site disaster
- **Erasure coded**
  - Experimental - under test

cinder.conf

```
[rbd-default]
volume_backend_name=rbd-default
volume_driver=cinder.volume.drivers.rbd.RBDDriver
rbd_user=cinder
rbd_pool=volumes

[rbd-ssd]
volume_backend_name=rbd-ssd
volume_driver=cinder.volume.drivers.rbd.RBDDriver
rbd_user=cinder
rbd_pool=volumes-ssd

[rbd-nomirror]
volume_backend_name=rbd-nomirror
volume_driver=cinder.volume.drivers.rbd.RBDDriver
rbd_user=cinder
rbd_pool=volumes-nomirror

[rbd-erasure]
volume_backend_name=rbd-erasure
volume_driver=cinder.volume.drivers.rbd.RBDDriver
rbd_user=cinder-erasure
rbd_pool=erasure-metadata
```

Preliminary test results

- Comparison of different types of disks (SSD/HDD) and different types of pools (replicated vs erasure coded)
- Use of different tools and procedures:
    - cinder volume attached to a VM; *dd* run inside the VM
    - rbd image mapped to a device on an hypervisor; run *fio + rbd engine*
    - Patterns: seq write/read, random write/read
    - Size: 4KB —> 1MB
- As expected, SSD pools show better read/write performances
- In general, writes are slower than reads
- Erasure coded pools do not present significant overhead
- Testing data durability: write 2TB of data (split in 2GB files) and check the md5 sum after some months

Building sync&share services

- Experimenting with **Nextcloud** setup on top of resources (VMs and block devices) deployed on the INFN CC
- Developing **Heat/TOSCA templates**:
  - Allow users to deploy their own file share platform in an easy fully automated way
    - Using Openstack Orchestrator or INDIGO-DataCloud PaaS
  - Support for **disk encryption** (LUKS)
    - Critical for some use cases: sensitive data, GDPR compliance

Thank you!