

oceanos

Building a real-world public cloud
from the ground up



Vangelis Koukis
vkoukis@grnet.gr
Technical Coordinator, ~oceanos Project

Outline

- ◆ ~oceanos ?
- ◆ Rationale
- ◆ Design – Platform - Features
- ◆ Unity - Automation
- ◆ Opensource – Upcoming



What is ~oceanos?



What is ~oceanos?

'oceanos' is Greek for 'ocean'.



What is ~oceanos?

'oceanos' is Greek for 'ocean'.

Oceans capture, store and deliver energy, oxygen and life around the planet.



Simplicity



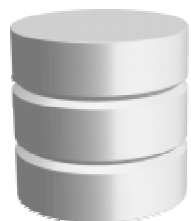




Compute



Network



Storage



Security



Virtual Machines



Virtual Ethernets



Virtual Disks

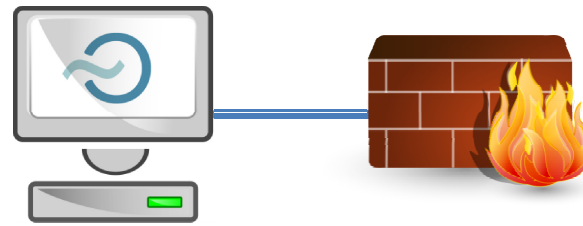


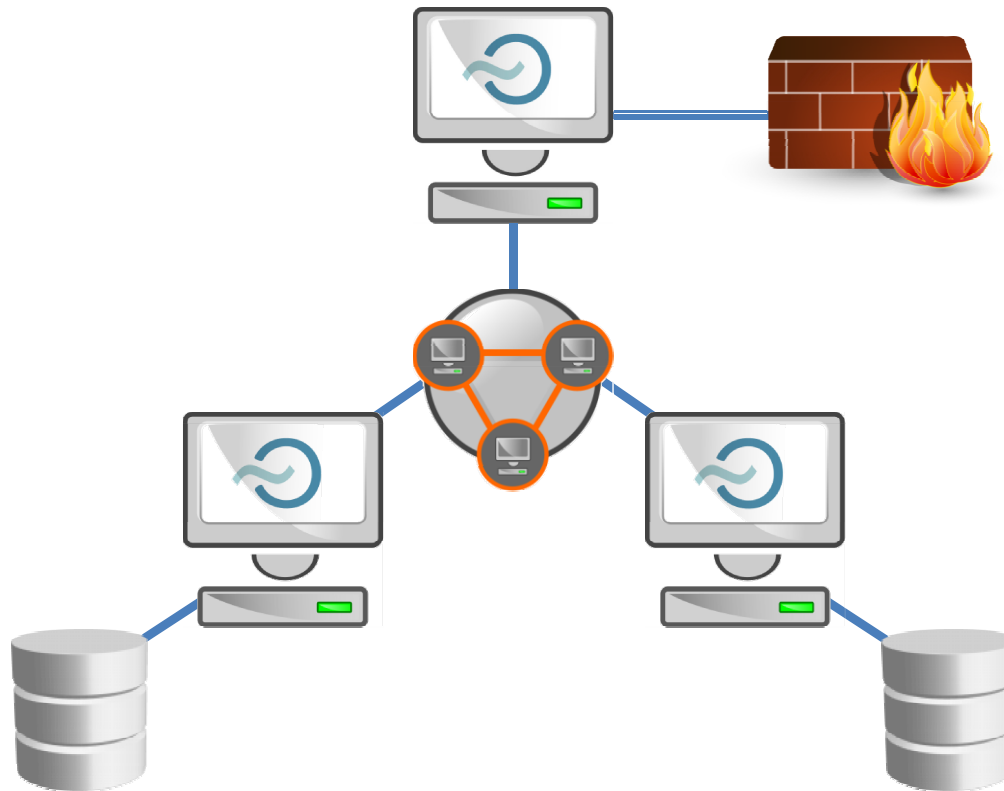
Virtual Firewalls

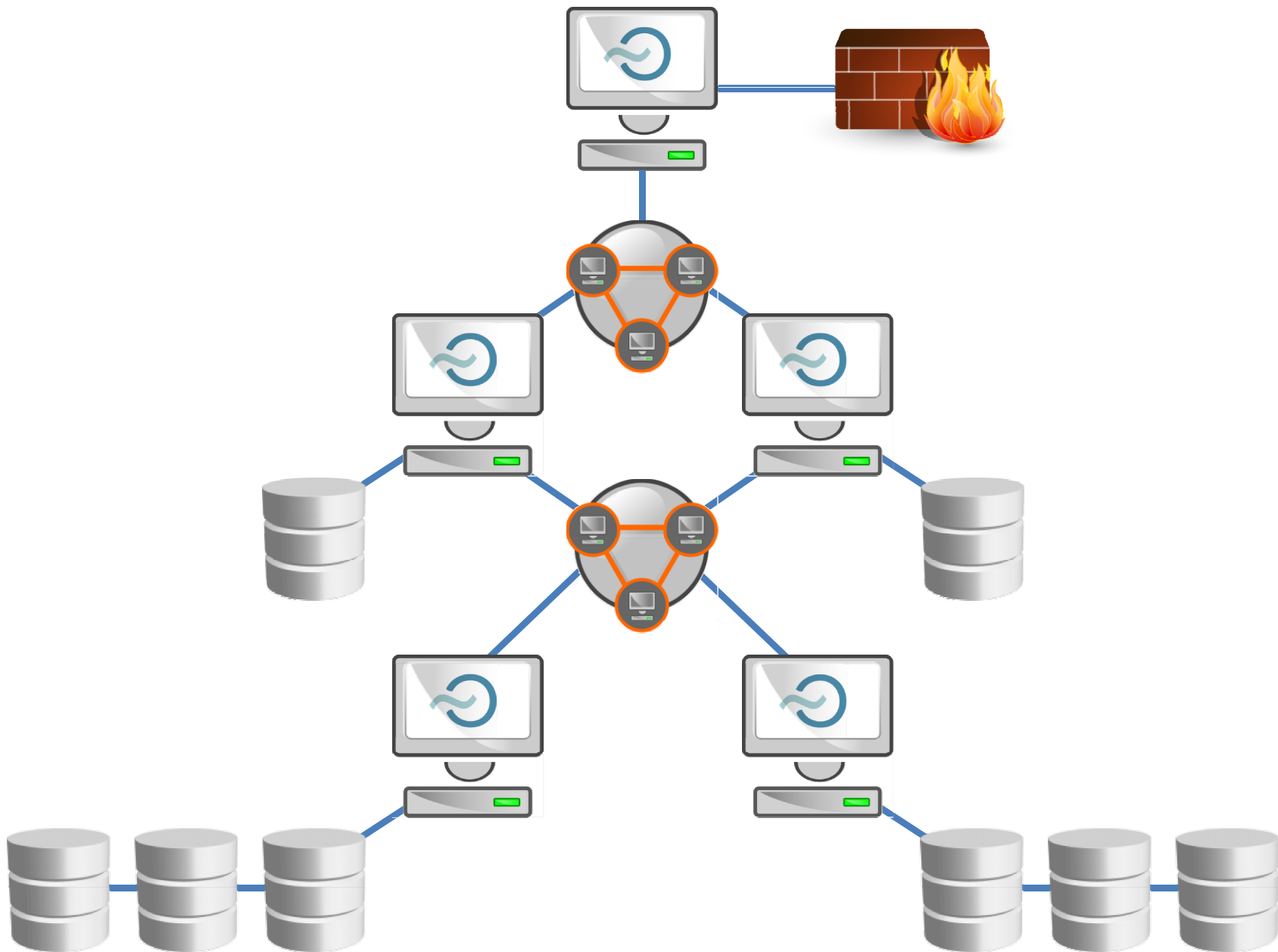
Flexibility







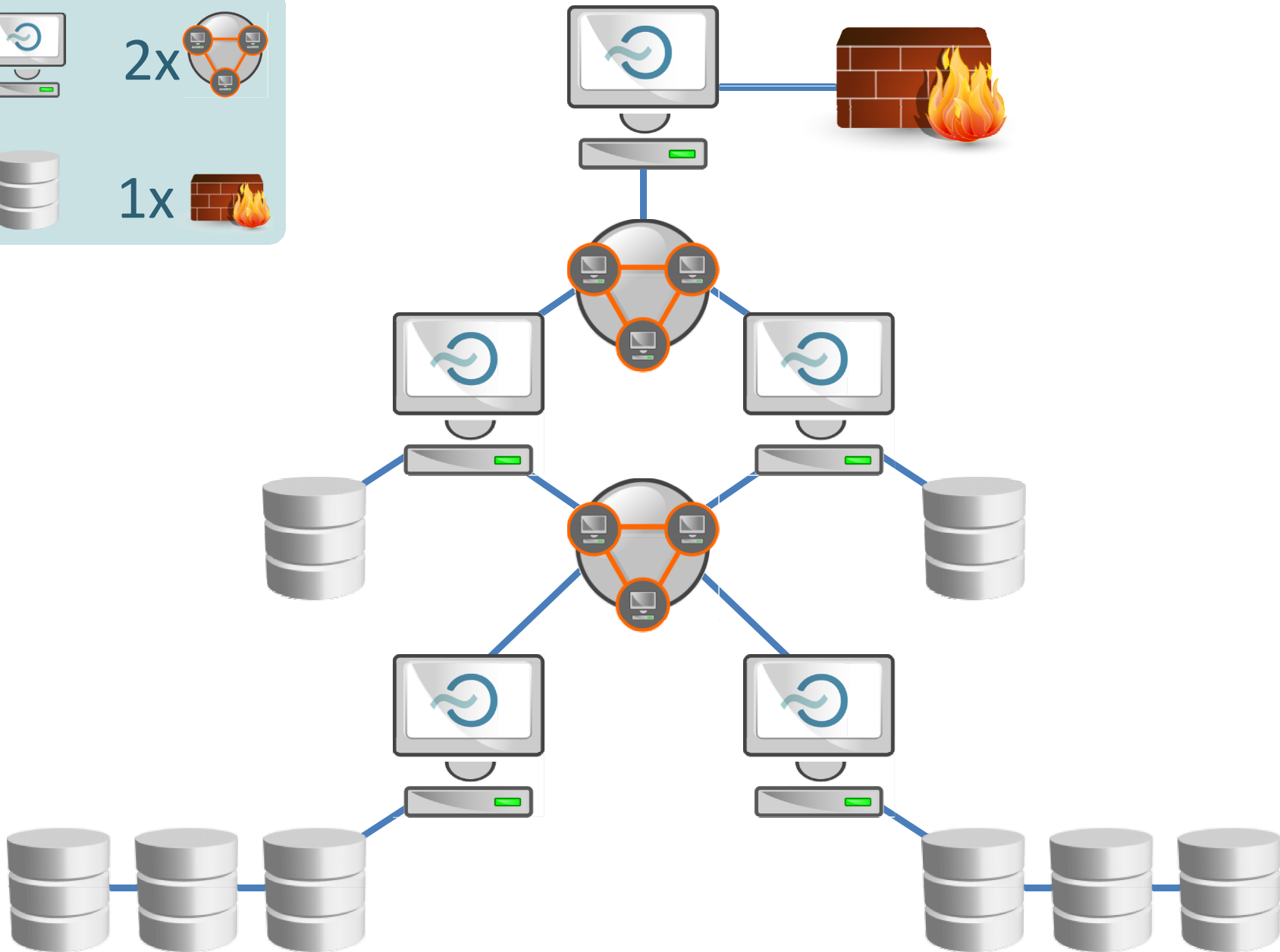








5x  2x 
8x  1x 



~okeanos service

- ◆ Goal: Production-quality IaaS
- ◆ Beta in Dec, current Alpha: >1600 VMs / >1000 users
- ◆ Target group: GRNET's customers
 - ➔ direct: IT depts of connected institutions
 - ➔ indirect: university students, researchers in academia
- ◆ Users manage resources over
 - ➔ a simple, elegant UI, or
 - ➔ a REST API, for full programmatic control



~okeanos features

- ◆ **Compute/Network Service:** Cyclades
- ◆ **File Storage Service:** Pithos+
- ◆ **Image Service:** Plankton
- ◆ **Identity Service:** Astakos

- ◆ **Volume Service:** Archipelago



Rationale

How it all started



How it all started

- ◆ Need for easy, secure access to GRNET's datacenters
 - ➔ User friendliness, simplicity
- ◆ Scalable to the thousands
 - ➔ #VMs, TBs, users (Pithos: ~10k)
- ◆ running within GRNET's AAI Federation
- ◆ Resell or build your own?
 - ➔ IaaS cloud provider, vendor, or own infrastructure?
 - ➔ It all depends on your needs



Build on commercial IaaS?

◆ Commercial IaaS

- ➔ Amazon EC2 not an end-user service
- ➔ Need to develop custom UI, AAI layers
- ➔ Vendor lock-in
- ➔ Unsuitable for IT depts
 - persistent, long-term servers
 - custom networking requirements

◆ GRNET has invested heavily in its core network

- ➔ > 8000km of dark fiber



Bring vendor into datacenter?

- ◆ Hypervisor lock-in
- ◆ Is a turn-key solution suitable for a public cloud?
- ◆ Building public clouds is an ongoing process
 - ➔ Manageable by GRNET's operation
 - ➔ Integrated into the rest of the infrastructure
 - ➔ Scaling to thousands of users
- ◆ Build on existing know-how
- ◆ Gain know-how, build own IaaS → reuse for own services



What about opensource?

- ◆ OpenStack, Eucalyptus, OpenNebula
- ◆ Need a mature opensource core to *build* around
- ◆ Maturity, production-readiness?
 - proven in production environments, predictable
- ◆ Extensibility?
- ◆ Flexibility?
- ◆ Upgradeability, maintainability?

Design

~okeanos design decisions

- ◆ Reuse existing components
- ◆ Build on Google Ganeti
- ◆ target commodity hardware
- ◆ release to the community as opensource



~okeanos design principles

- ◆ No need to make the world
- ◆ No need to support *everything*
 - ➔ Service developed and maintained by ~10-15 people
- ◆ Start from the architecture...
 - ➔ ...then discover, combine, reuse the right components
- ◆ And for everything that's not already available
 - ➔ Do it yourself!





Jigsaw puzzle

- ◆ Synnefo
 - ➔ custom cloud management software to power ~oceanos
- ◆ Google Ganeti backend
 - ➔ VM cluster management: physical nodes, VMs, migrations
- ◆ OpenStack APIs: Compute API v1.1, Object Storage API
 - ➔ with custom extensions whenever necessary
- ◆ Then everything comes together
 - ➔ UI, Networking, Images, Storage, Monitoring, Identity management, Accounting, Billing, Clients, Helpdesk

grnet

Why Ganeti?

- ◆ No need to reinvent the wheel
- ◆ Scalable, proven software infrastructure
 - ➔ Built with reliability and redundancy in mind
 - ➔ Combines open components (KVM, LVM, DRBD)
 - ➔ Well-maintained, readable code
- ◆ VM cluster management in production is serious business
 - ➔ reliable VM control, VM migrations, resource allocation
 - ➔ handling node downtime, software upgrades



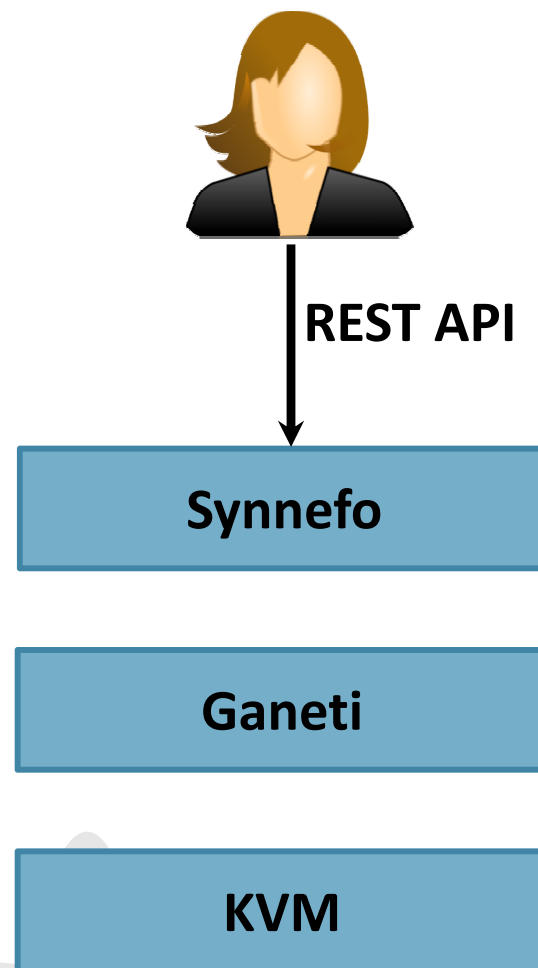
Why Ganeti?

- ◆ GRNET already had long experience with Ganeti
 - ➔ provides ~280 VMs to NOCs through the ViMa service
 - ➔ involved in development, contributing patches upstream
- ◆ Build on existing know-how for ~oceanos
 - ➔ Common backend, common fixes
 - ➔ reuse of experience and operational procedures
 - ➔ simplified, less error-prone deployment

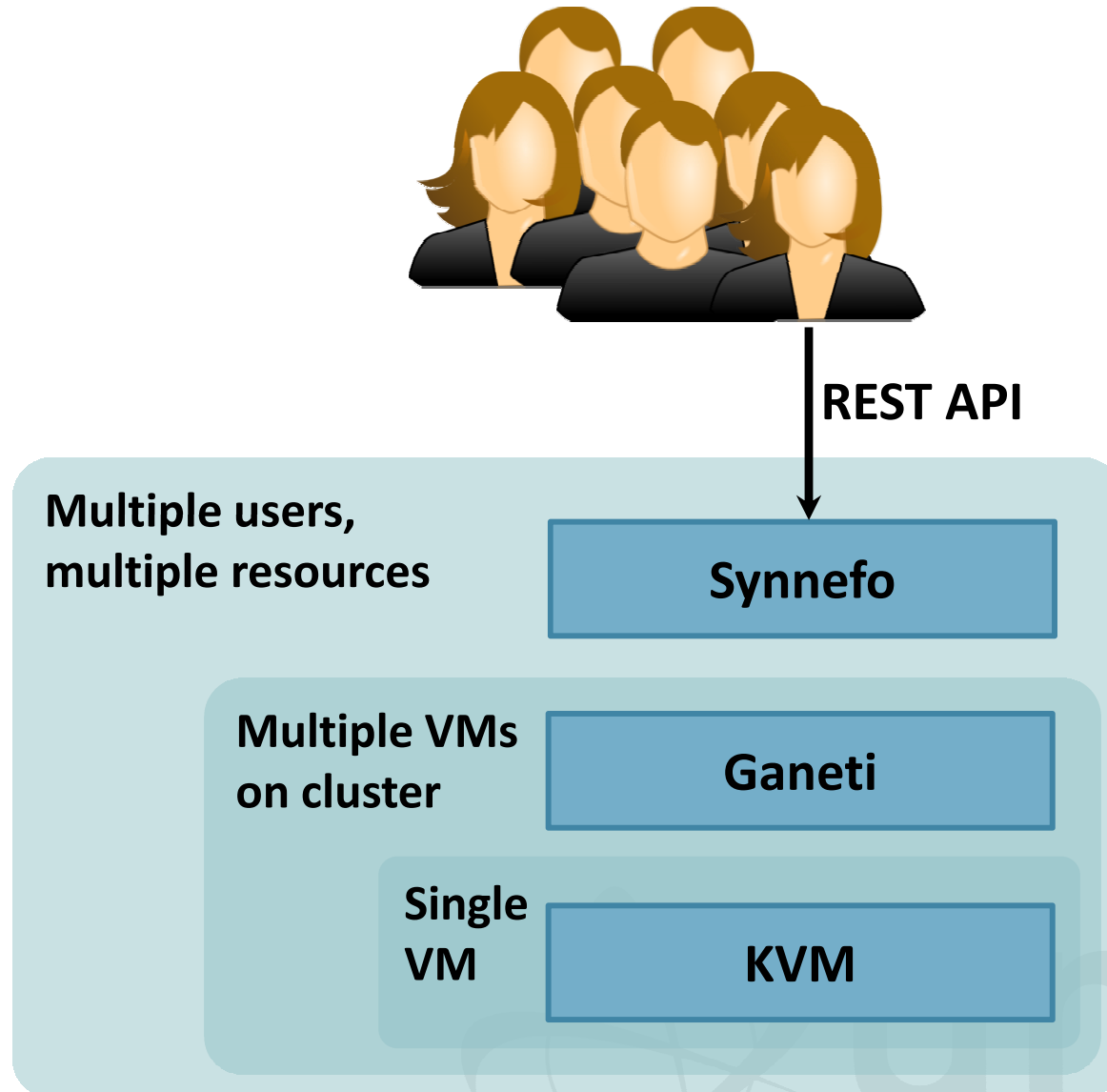


Platform

Software Stack



Software Stack

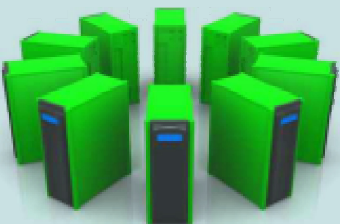


Platform Design

user@home

admin@home

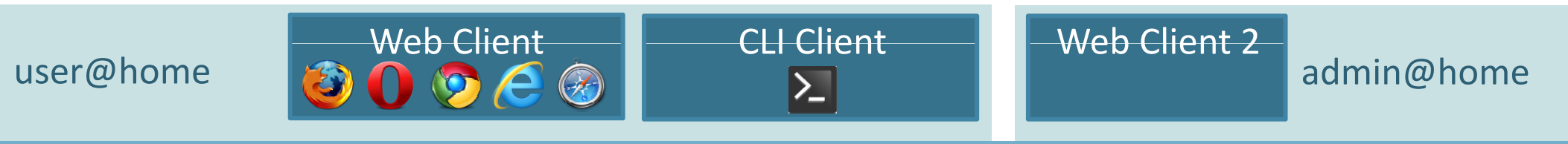
GRNET
datacenter



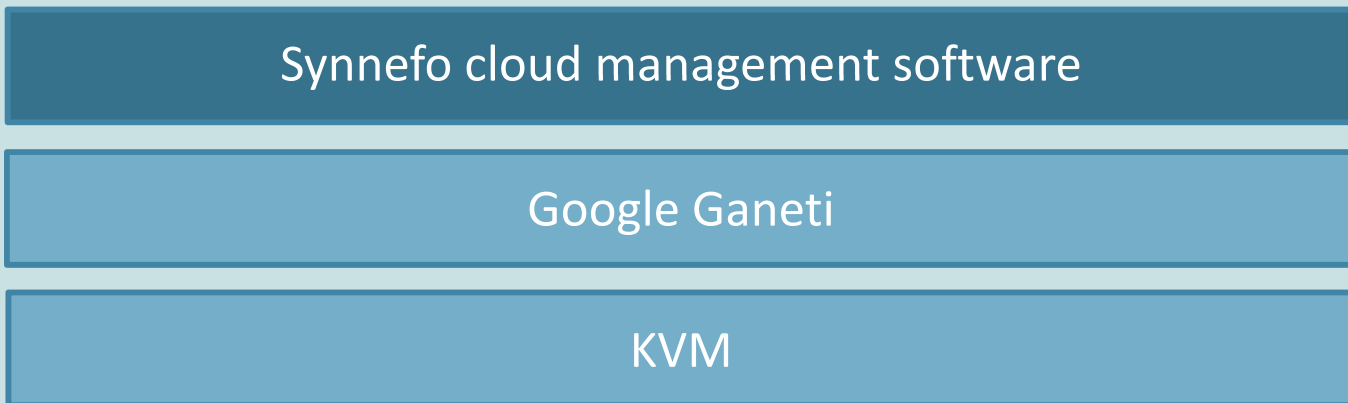
Virtual
Hardware



Platform Design



GRNET datacenter

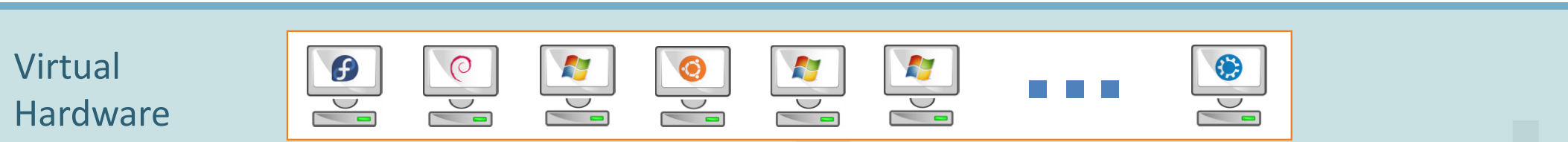
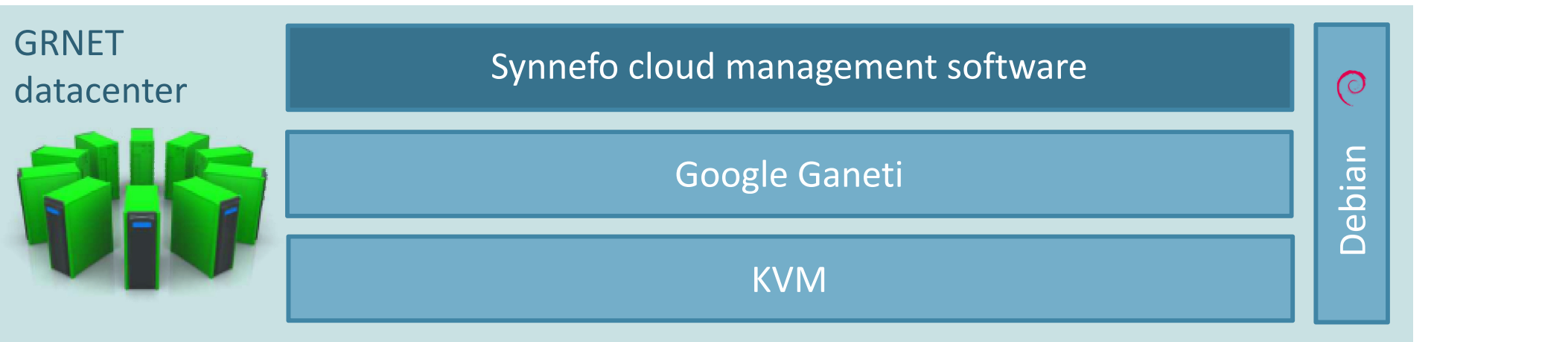


Debian

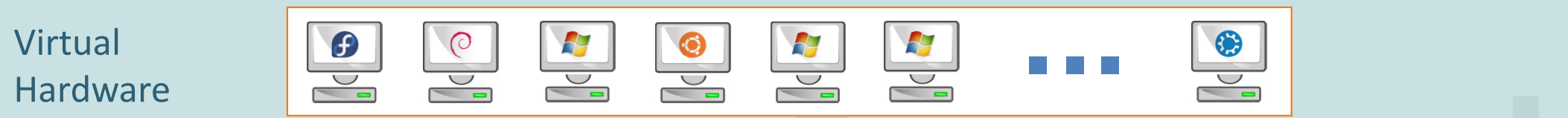
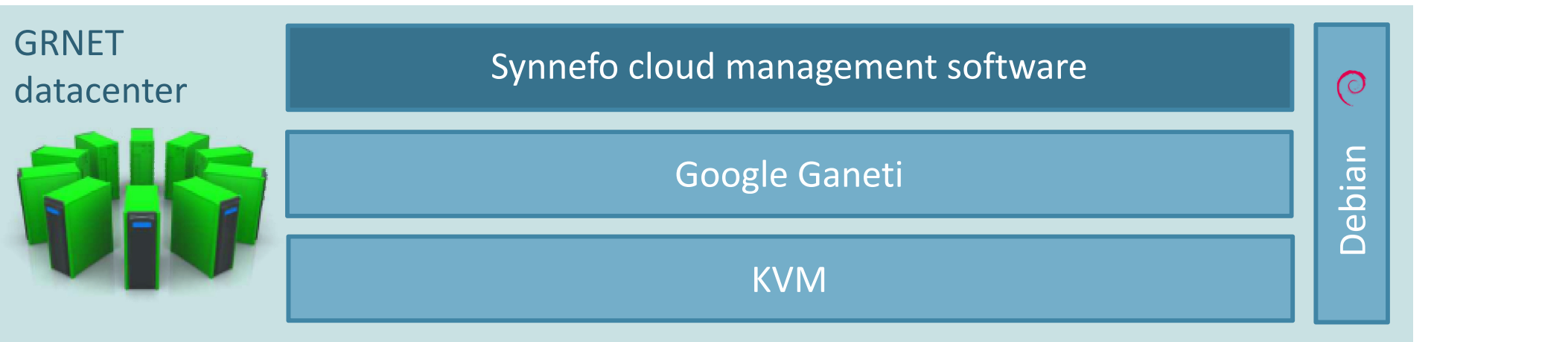
Virtual Hardware



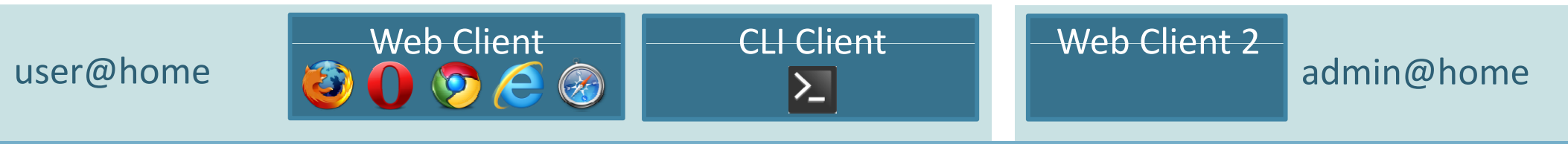
Platform Design



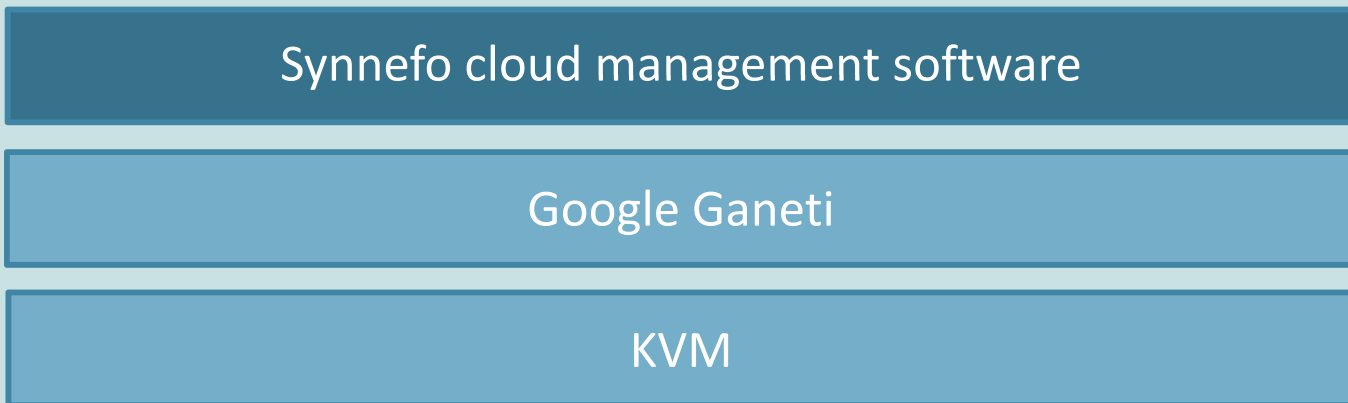
Platform Design



Platform Design

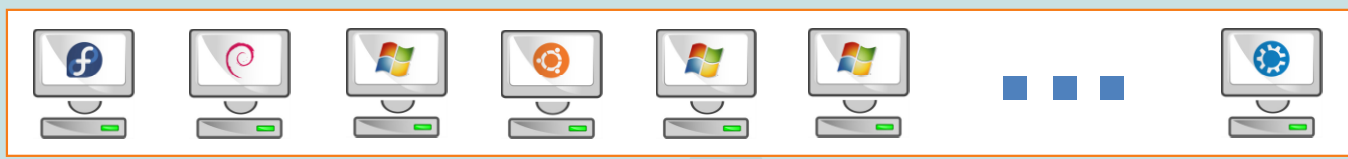


GRNET
datacenter

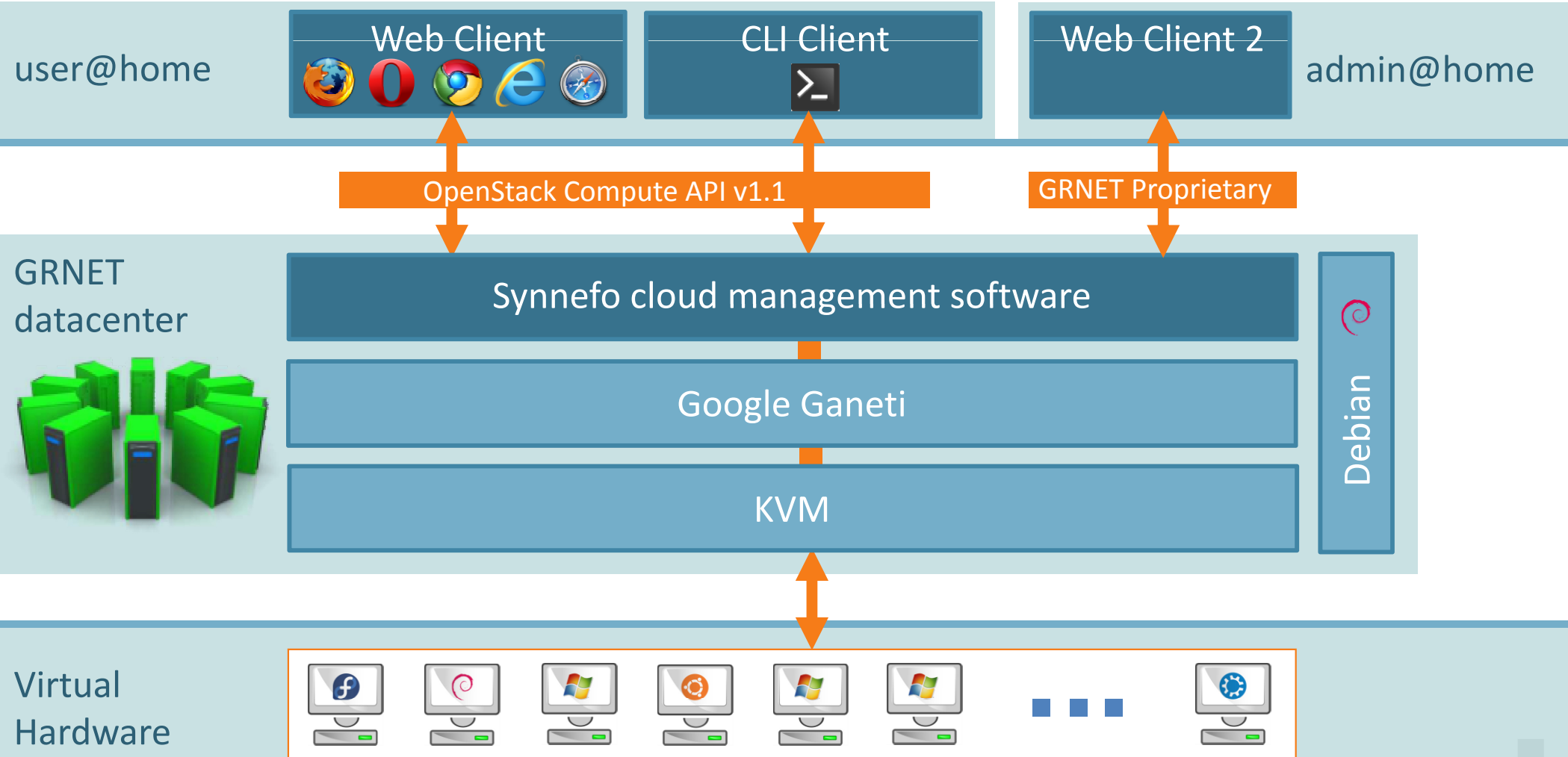


Debian

Virtual
Hardware



Platform Design



Features

Virtual Machine Actions



My_Windows_desktop

Virtual Machine Actions



My_Windows_desktop



Start



Reboot



Shutdown



Virtual Machine Actions



My_Windows_desktop



Start



Console



Reboot



Shutdown



Destroy



IaaS – Compute (1)

◆ Virtual Machines

- ➔ powered by KVM
 - Linux and Windows guests, on Debian hosts
- ➔ Google Ganeti for VM cluster management
- ➔ accessible by the end-user over the Web or programmatically (OpenStack Compute v1.1)

IaaS – Compute (2)

◆ User has full control over own VMs

➔ Create

- Select # CPUs, RAM, System Disk
- OS selection from pre-defined or *custom* Images
- popular Linux distros (Fedora, Debian, Ubuntu)
- Windows Server 2008 R2

➔ Start, Shutdown, Reboot, Destroy

➔ Out-of-Band console over VNC for troubleshooting



IaaS – Compute (3)

- ◆ REST API for VM management
 - ➔ OpenStack Compute v1.1 compatible
 - ➔ 3rd party tools and client libraries
 - ➔ custom extensions for yet-unsupported functionality
 - ➔ Python & Django implementation
- ◆ Full-featured UI in JS/jQuery
 - ➔ UI is just another API client
 - ➔ All UI operations happen over the API

IaaS – Network (Virtual Ethernets)



Internet



Private Network 1

IaaS – Network (Virtual Ethernets)



Internet



Private Network 1



IaaS – Network (Virtual Ethernets)



Internet



Private Network 1



IaaS – Network (Virtual Ethernets)



Internet



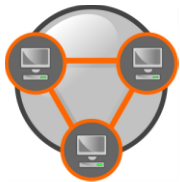
Private Network 1



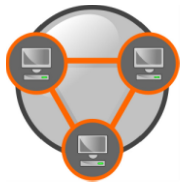
IaaS – Network (Virtual Ethernets)



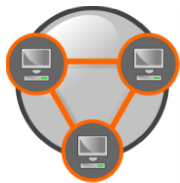
Internet



Private Network 1



Private Network 2



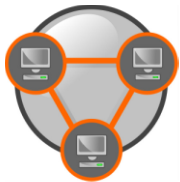
Private Network 3



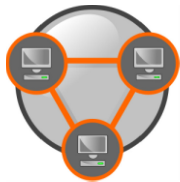
IaaS – Network (Virtual Ethernets)



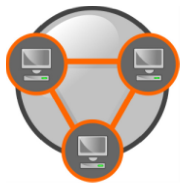
Internet



Private Network 1



Private Network 2



Private Network 3



IaaS – Network - Functionality

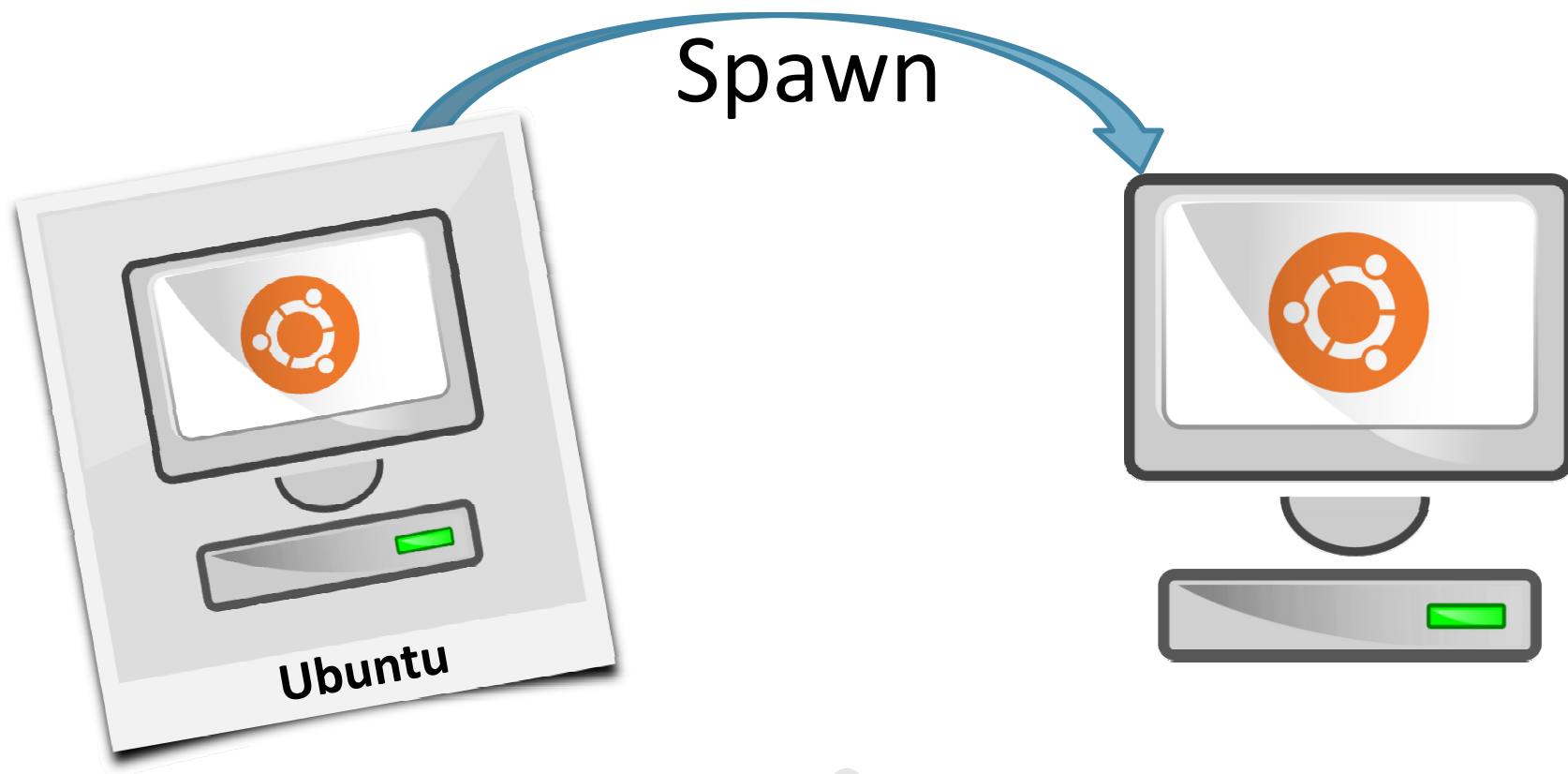
- ◆ Dual IPv4/IPv6 connectivity for each VM
- ◆ Easy, platform-provided firewalling
 - ➔ Array of pre-configured firewall profiles
 - ➔ Or roll-your-own firewall inside VM
- ◆ Multiple private, virtual L2 networks
- ◆ Construct arbitrary network topologies
 - ➔ e.g., deploy VMs in multi-tier configurations
- ◆ Exported all the way to the API and the UI

Unity

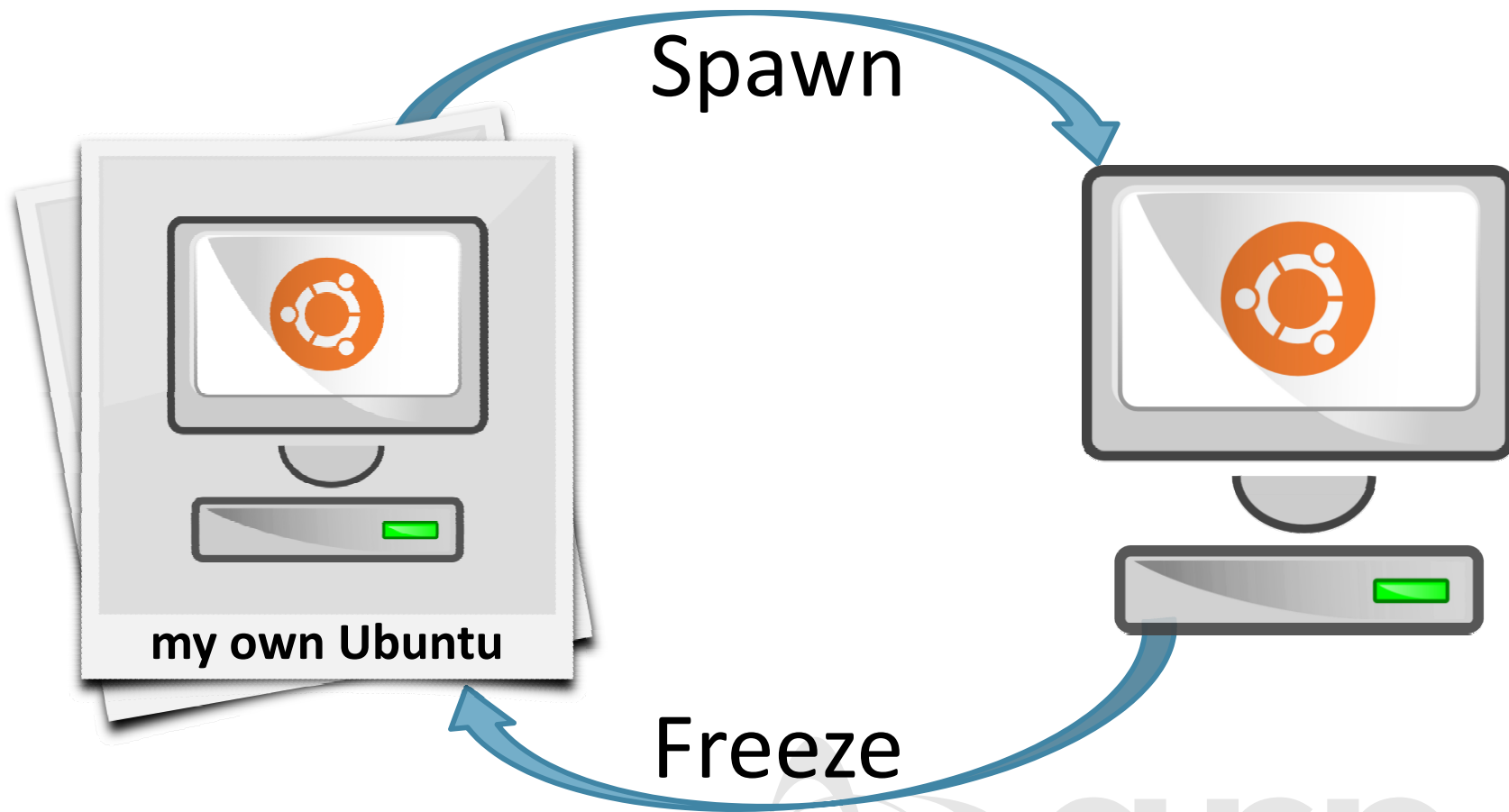
Images



Images



Images



grnet

Custom Images: snf-image

◆ *Untrusted* images

- ➔ Host cannot touch user-provided data
- ➔ Resize fs, change hostname, change passwords, inject files

◆ Split design

- ➔ snf-image-host
- ➔ snf-image-helper

◆ All customization in helper VM





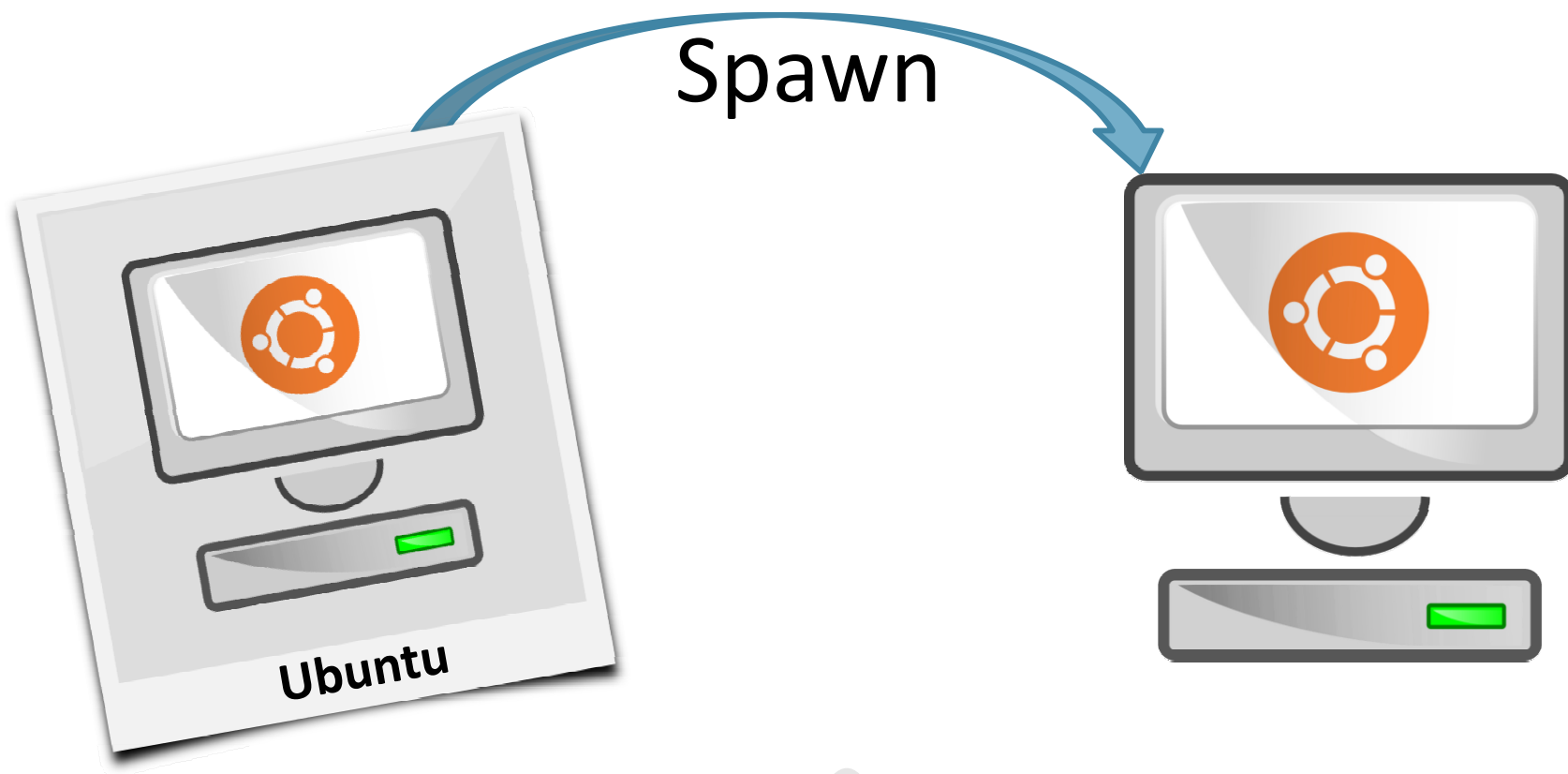
- ◆ OpenStack Object Storage API
- ◆ Block storage
- ◆ Content-based addressing for blocks
- ◆ Every file is a collection of blocks
- ◆ Web-based, command-line, and native clients
- ◆ Synchronization, deduplication
- ◆ An integral part of ~oceanos
 - ➔ User files, Image registry for VM Images
 - ➔ Goal: use common backend with Archipelago

grnet

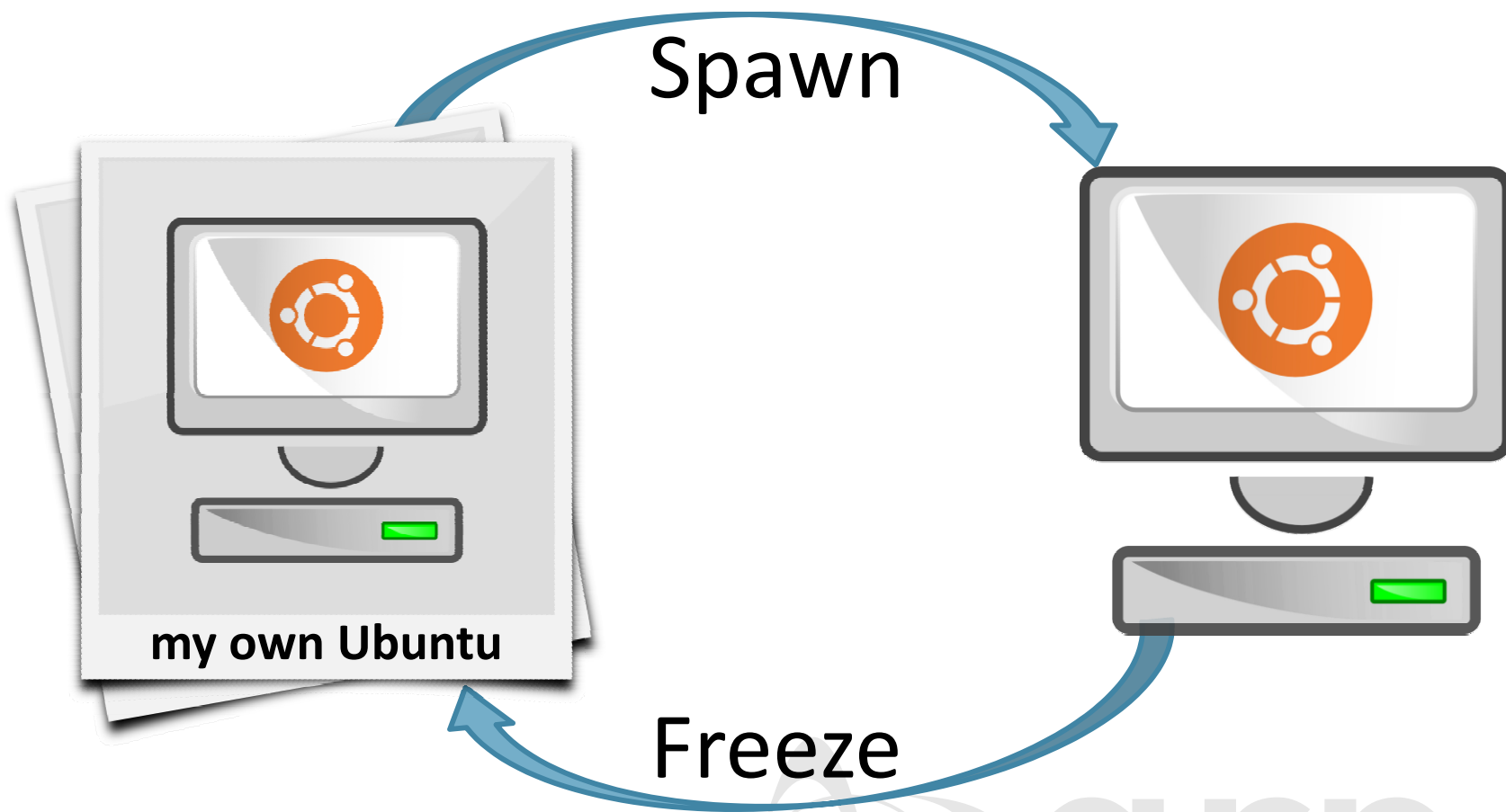
Images



Images

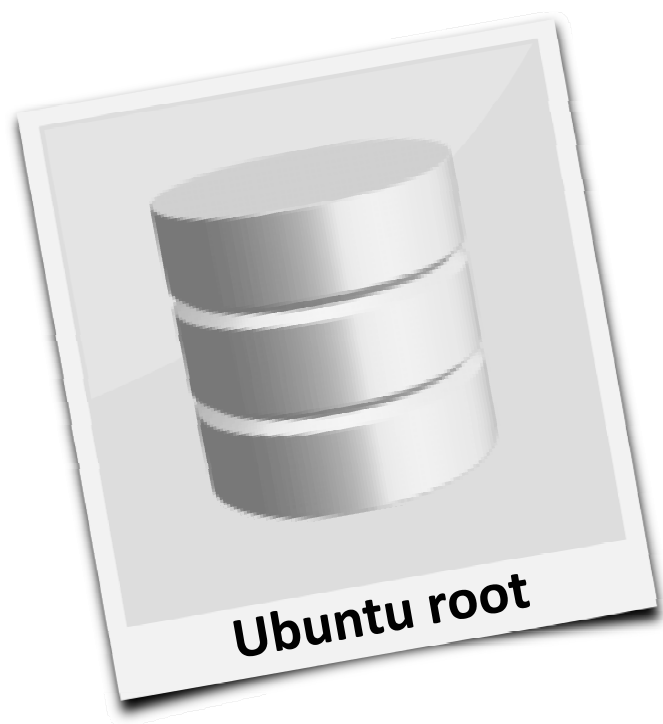


Images

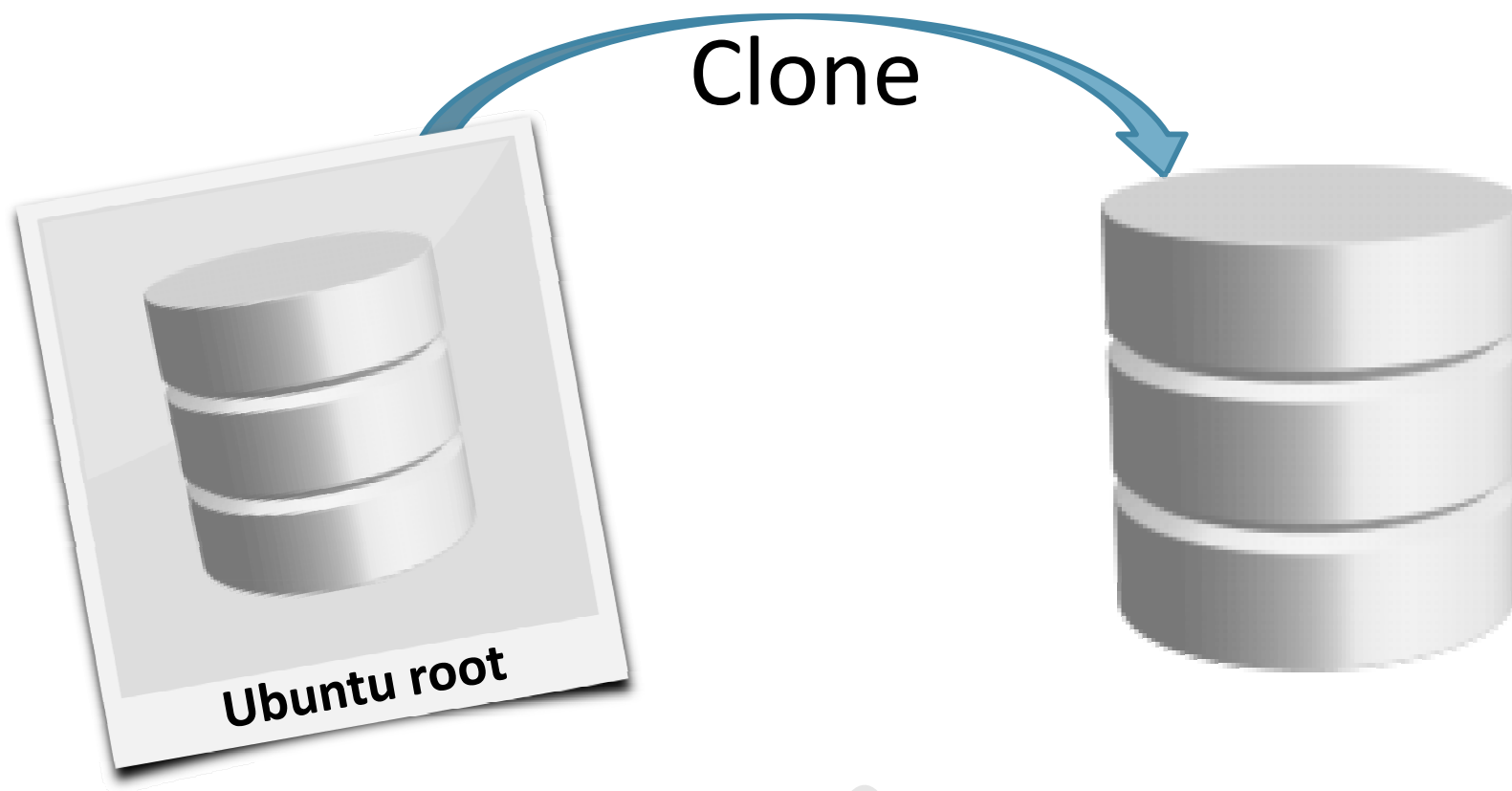


grnet

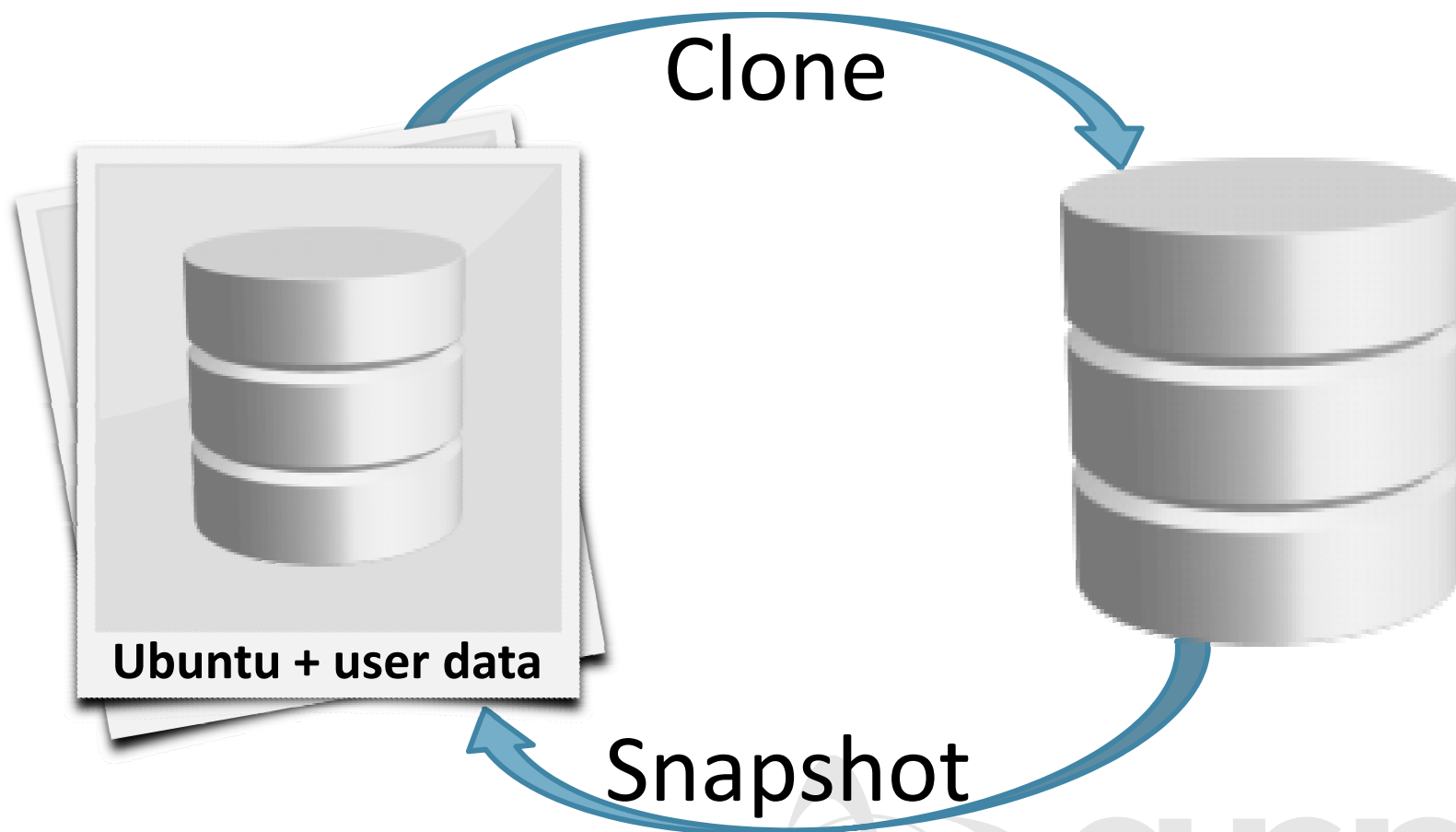
Images ↔ Storage



Images ↔ Storage



Images ↔ Storage

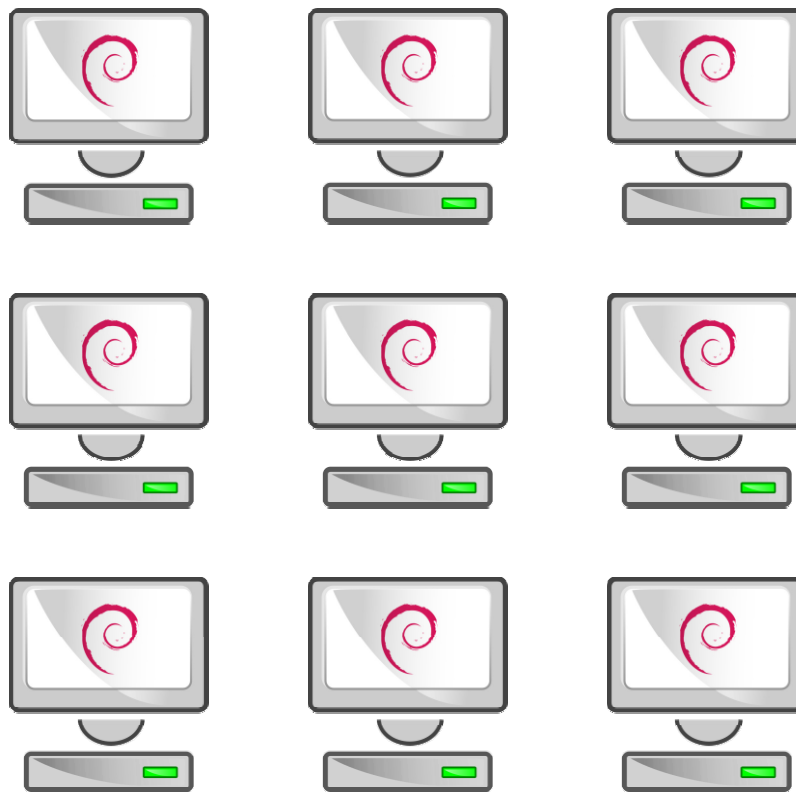


grnet

Images – Golden Image



Images – Golden Image



IaaS – Storage



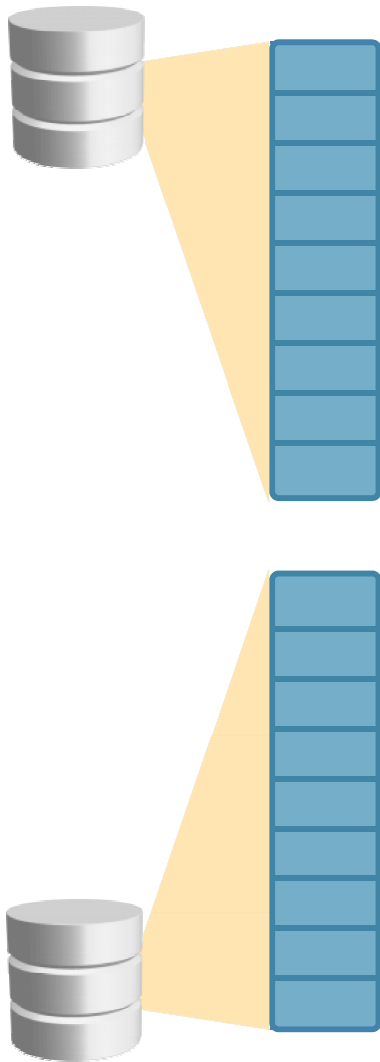
IaaS – Storage



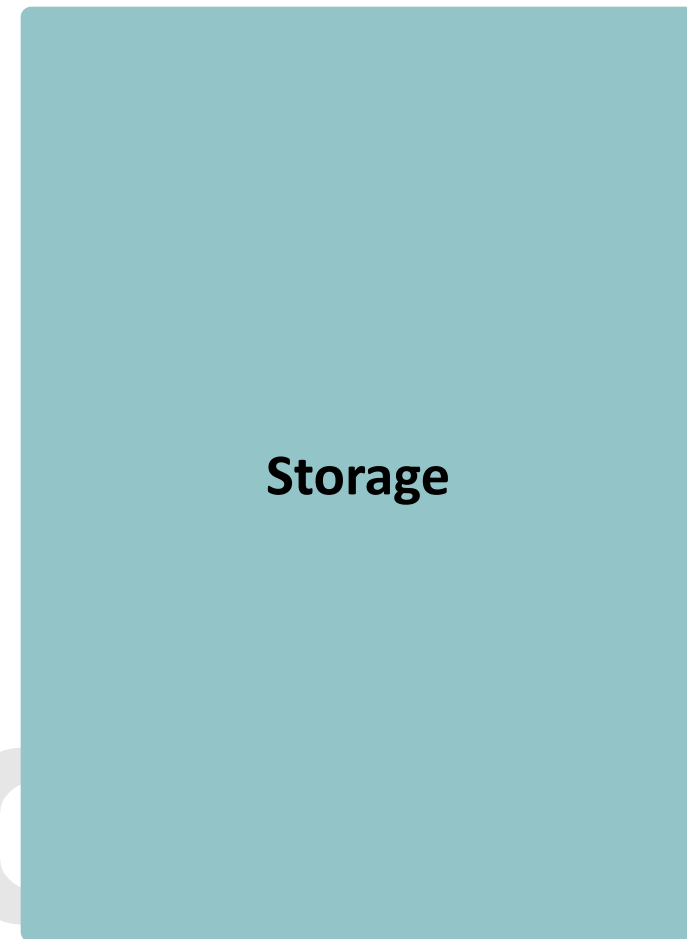
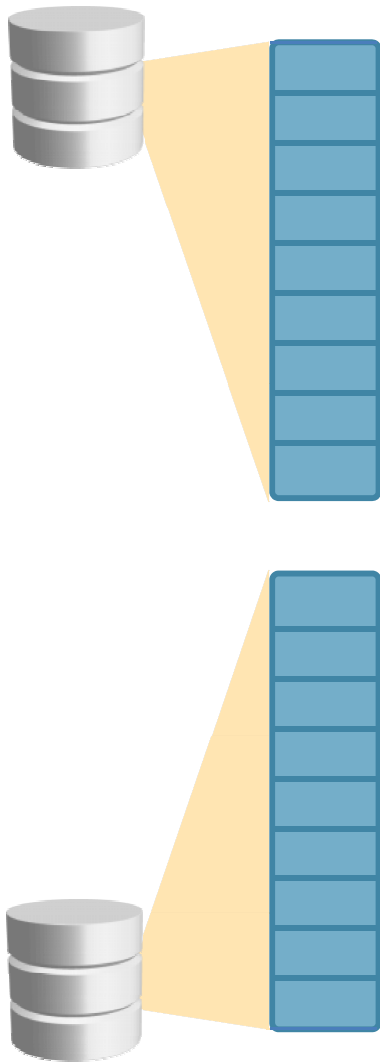
IaaS – Storage



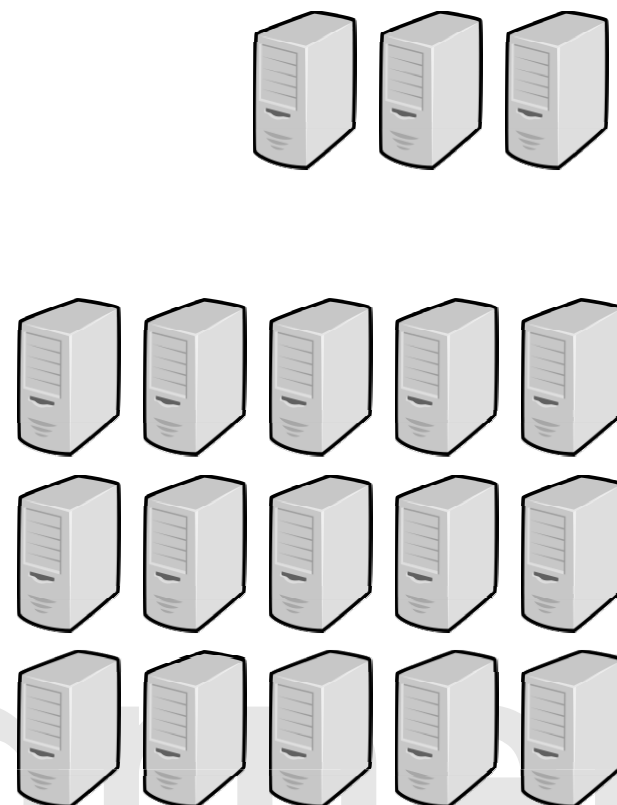
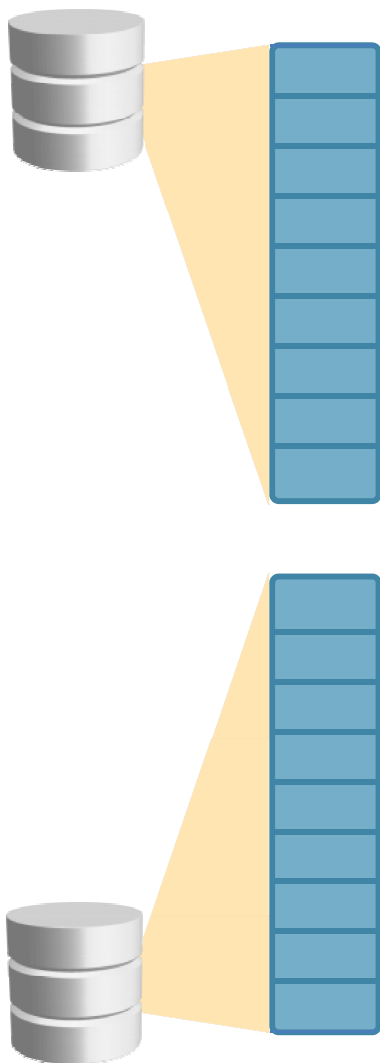
IaaS – Storage



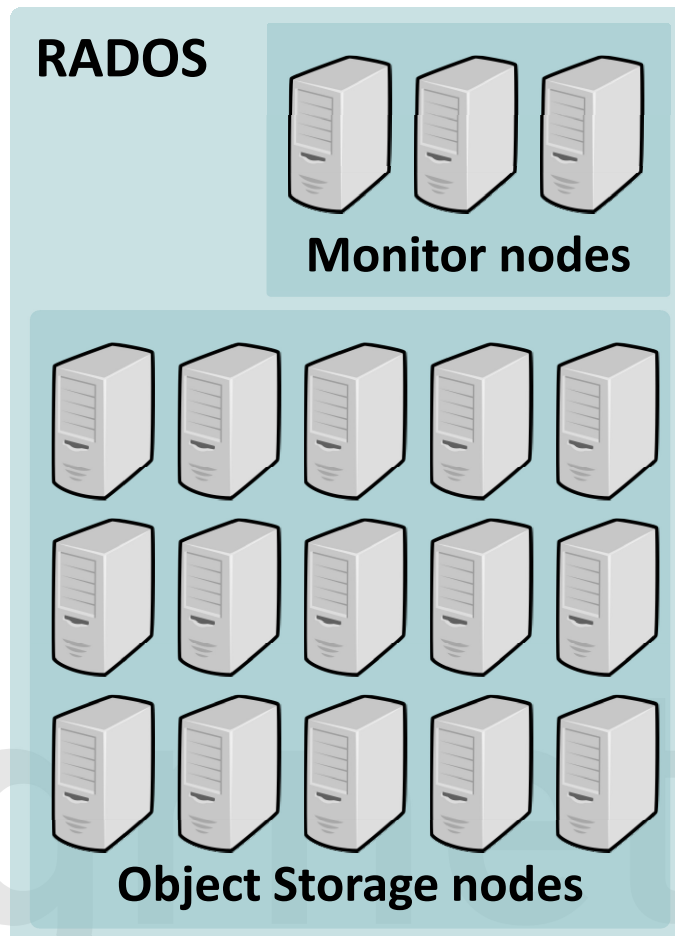
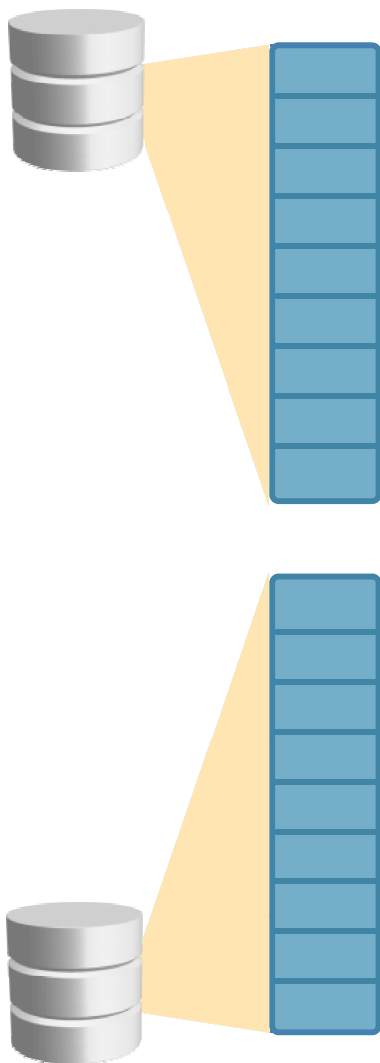
IaaS – Storage



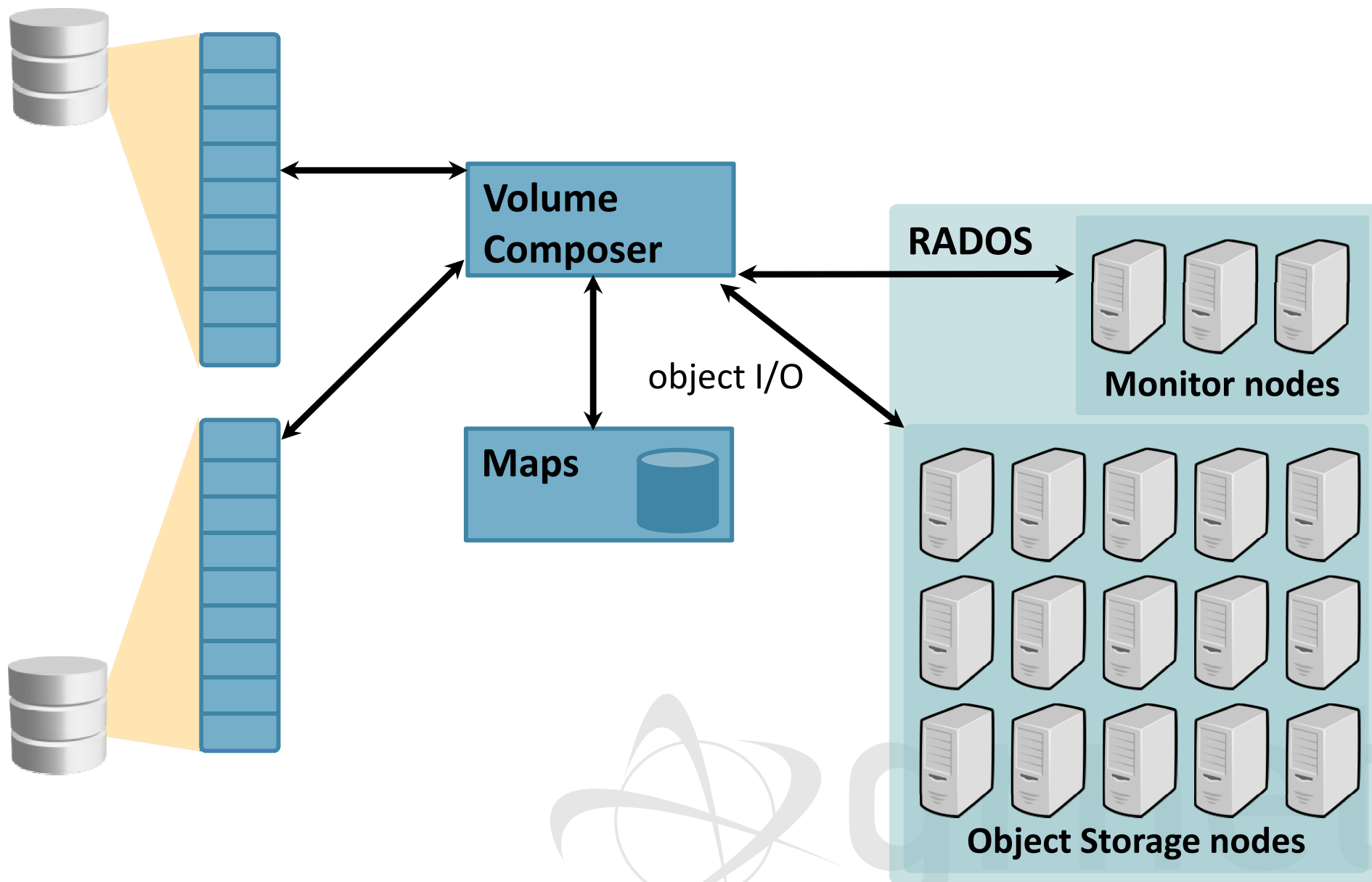
IaaS – Storage



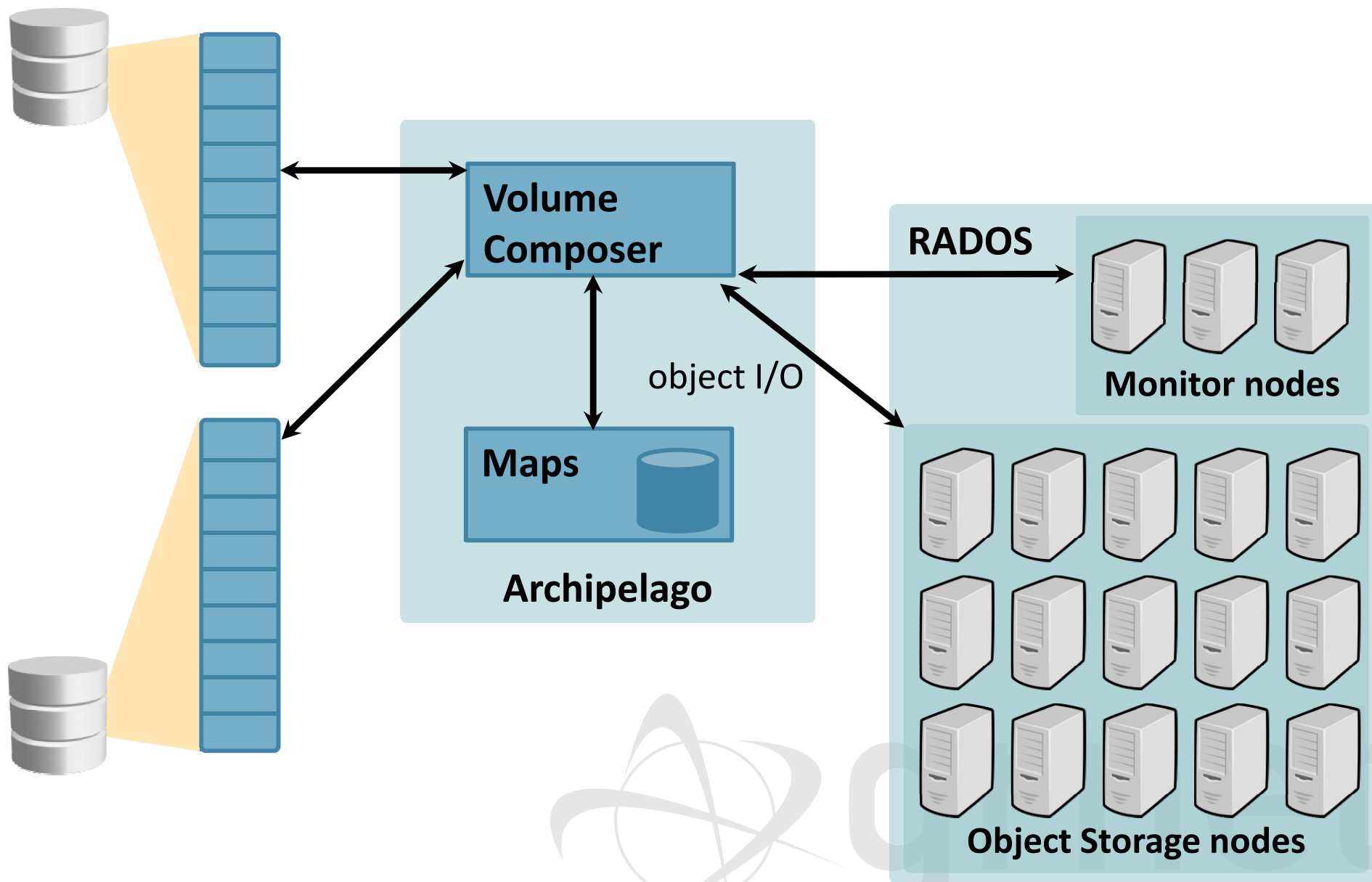
IaaS – Storage



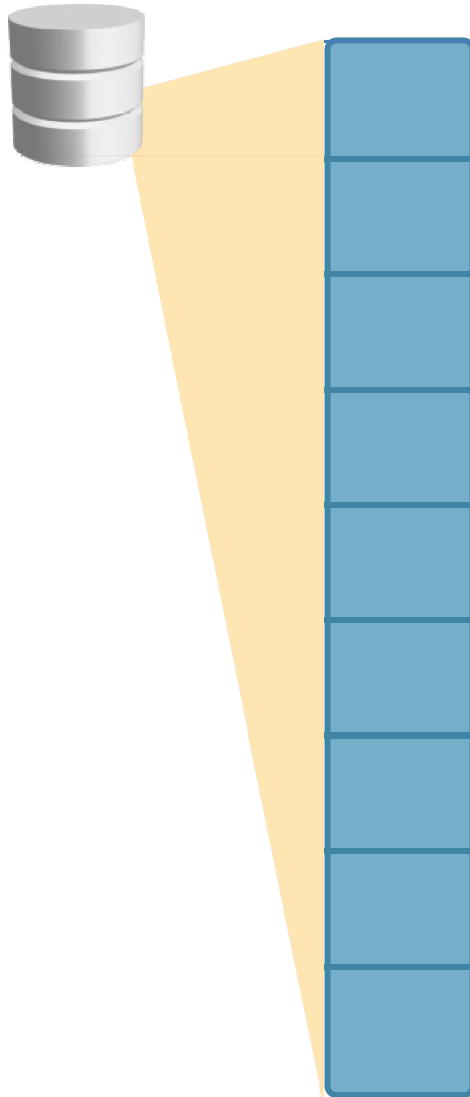
IaaS – Storage



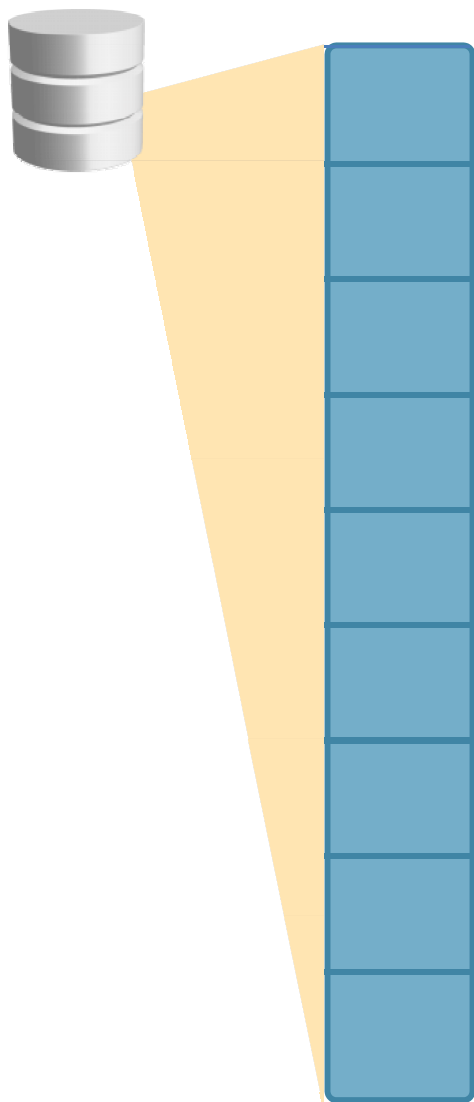
IaaS – Storage



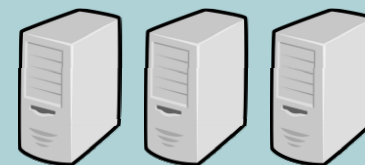
IaaS – Storage



IaaS – Storage



RADOS



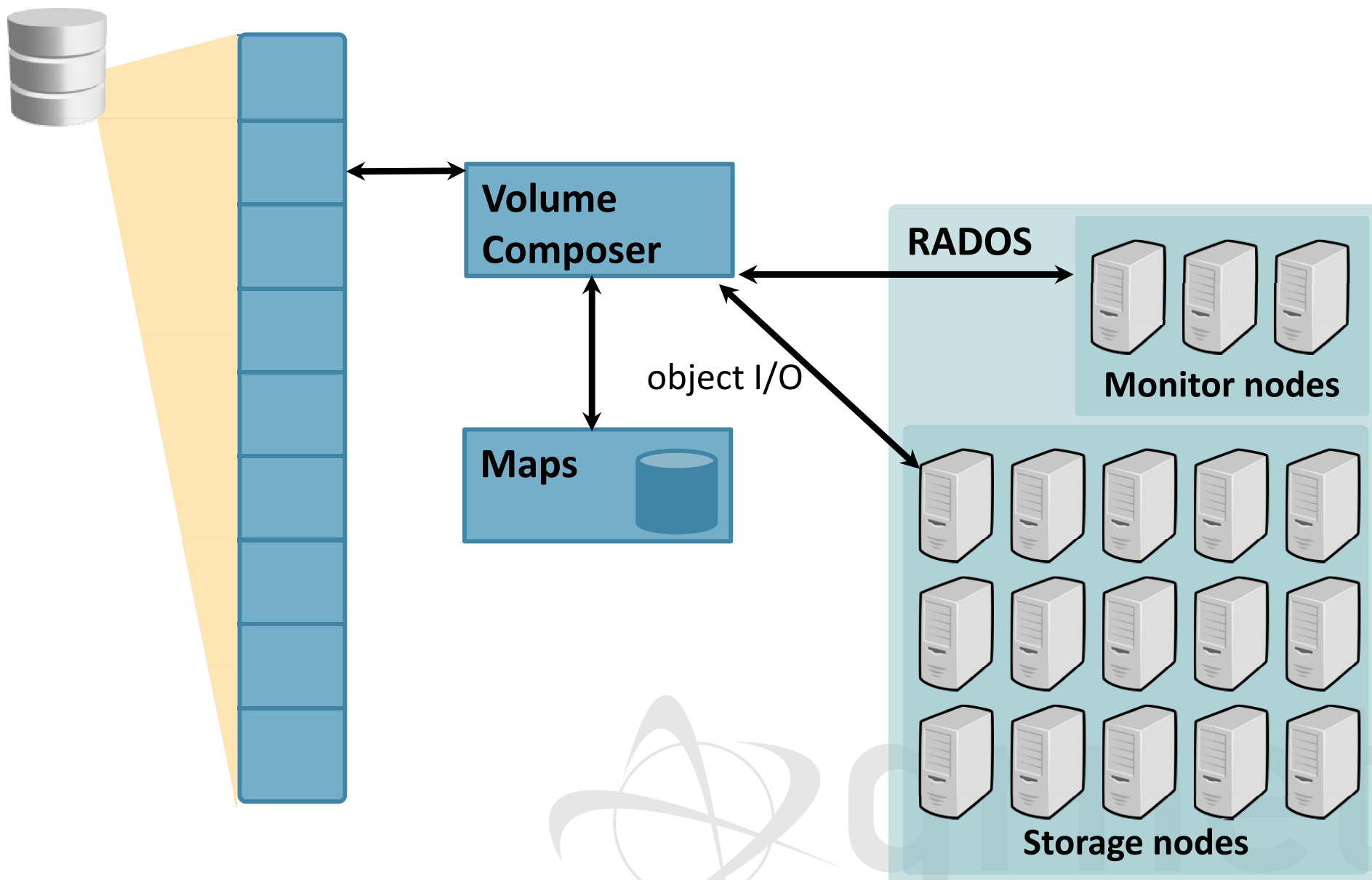
Monitor nodes



Storage nodes



IaaS – Storage



IaaS – Storage (1)

◆ First-phase deployment

- ➔ System-provided *and* custom user Images
- ➔ Redundant storage based on DRBD
- ➔ VMs survive physical node downtime or failure

◆ Currently under testing

- ➔ Reliable distributed storage over RADOS
- ➔ Combined with custom software for snapshotting, cloning
- ➔ Dynamic virtual storage volumes



IaaS – Storage (2)

- ◆ Multi-tier storage architecture
 - ➔ Dedicated Storage Nodes (SSD, SAS, and SATA storage)
 - ➔ OSDs, e.g., for RADOS
- ◆ Custom storage layer: Archipelago
 - ➔ manages snapshots, creates clones over block pools
 - ➔ OS Images held as snapshots
- ◆ VMs created as clones of snapshots

Integration

oceanos
Service

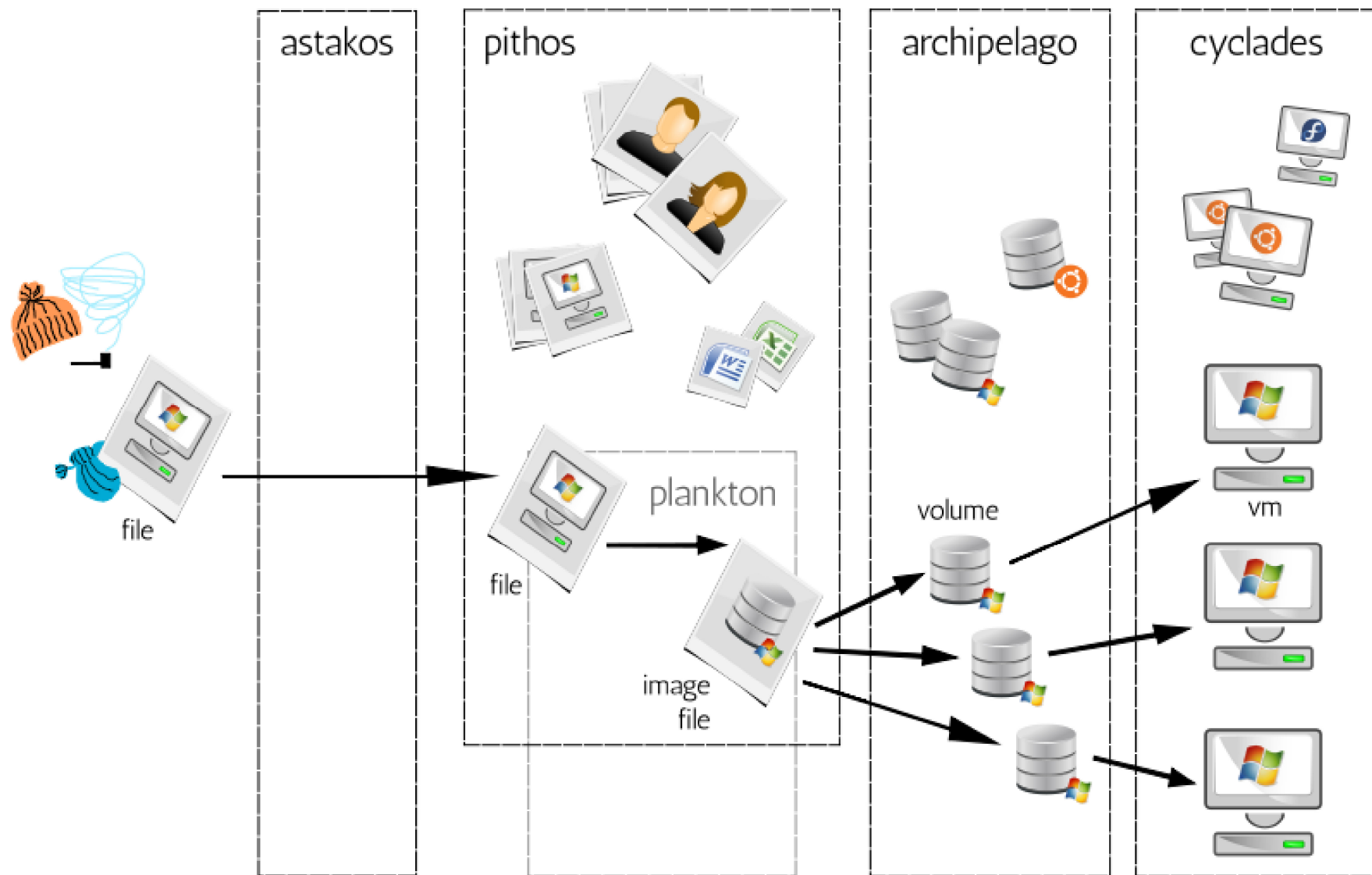
Identity
Management

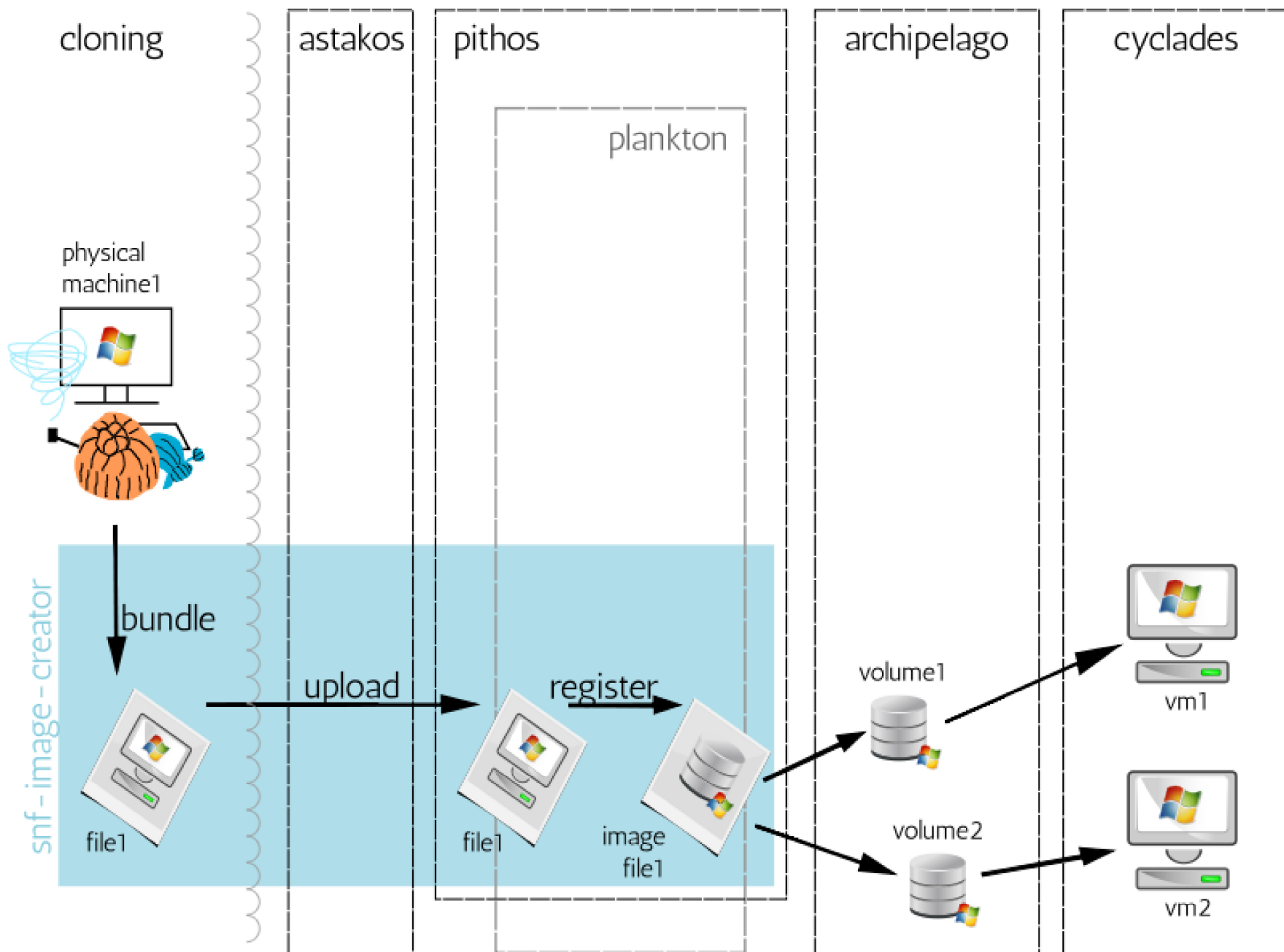
Storage
Service

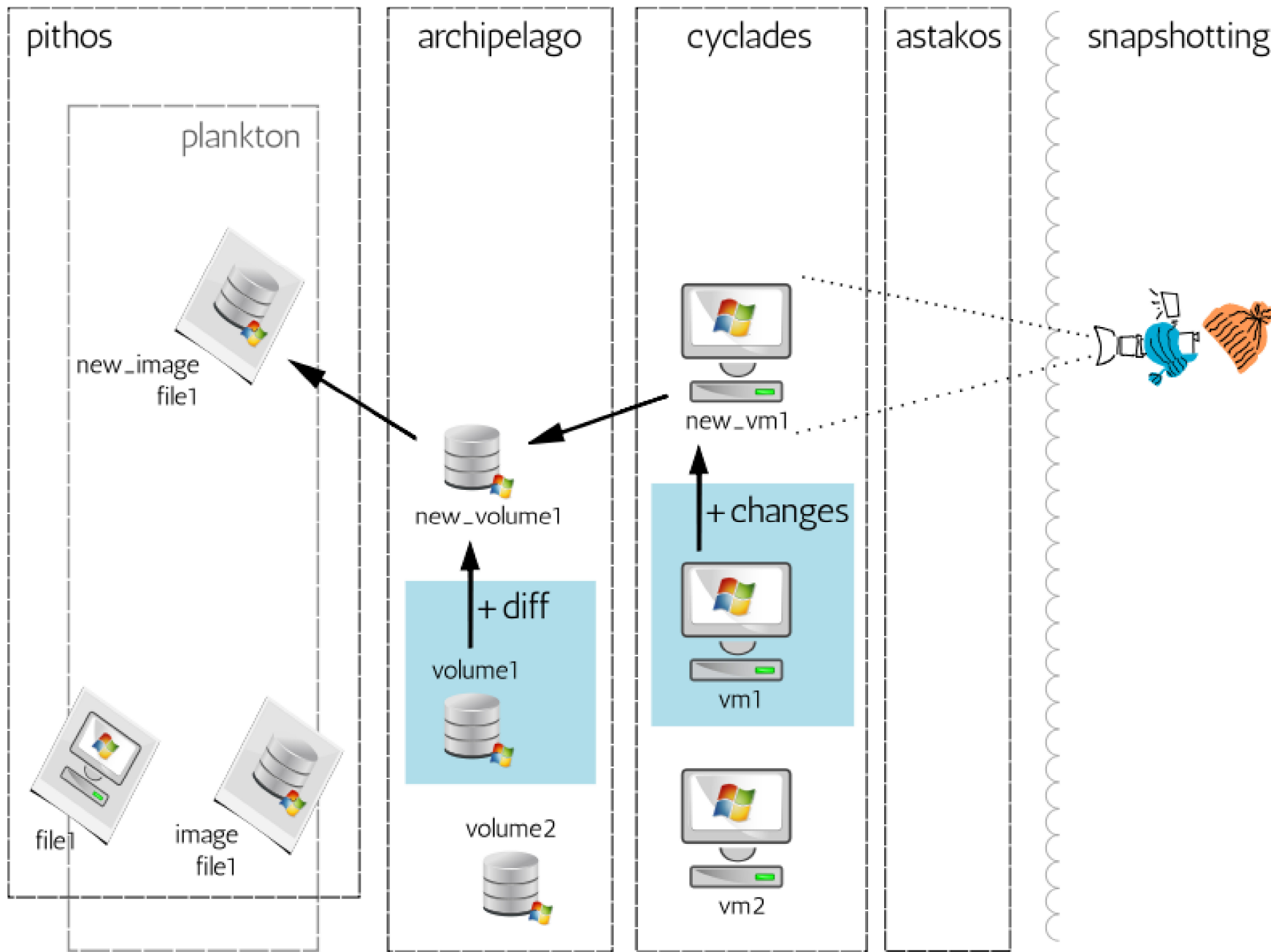
Image
Service

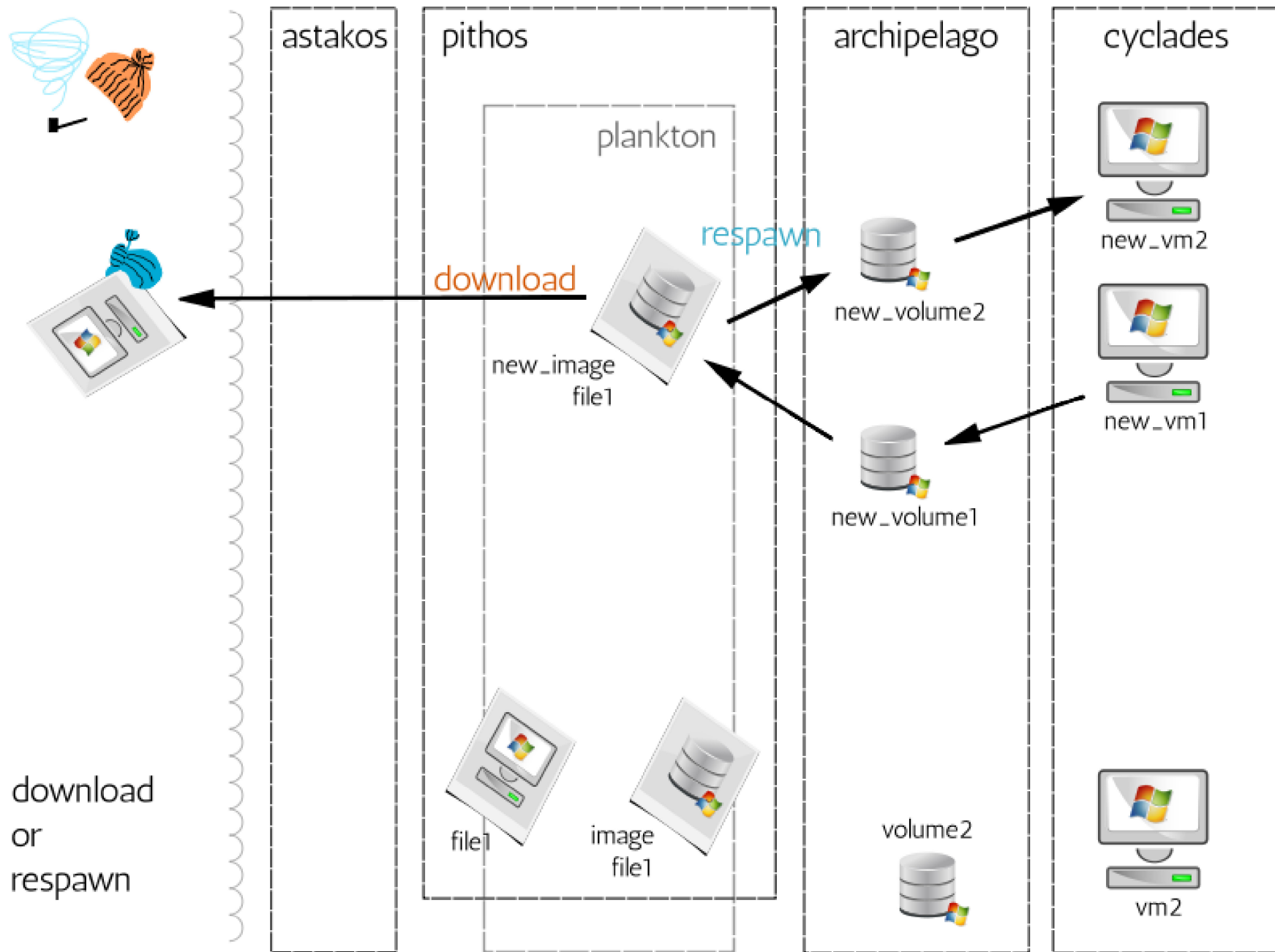
Volume
Service

Compute/Network
Service









Support services

◆ Identity: Astakos

- ➔ Provides the user base for ~oceanos
- ➔ Once authenticated, the user retrieves a common auth token for programmatic access



Automation

./kamaki

```
$ ./kamaki
```

```
Usage: kamaki <group> <command> [options]
```

```
...
```

```
--api=API      API can be either openstack or synnefo
```

```
--url=URL      API URL
```

```
--token=TOKEN use token TOKEN
```

```
...
```

```
Commands:
```

```
  flavor info      get flavor details
```

```
  flavor list      list flavors
```

```
...
```

```
  image create     create image
```

```
  image delete     delete image
```

```
$ ./kamaki server shutdown 101 --url=http://localhost:8000/api/v1.1  
--token=1234527db2...
```



./kamaki

```
$ ipython
```

```
In [1]: from kamaki.client import Client
```

```
In [2]: c = Client('http://localhost:8000/api/v1.1', "1234527db2..")
```

```
In [3]: c.list_flavors()
```

```
...
```

```
In [4]: i = c.list_images()
```

```
In [5]: i[5]
```

```
{u'created': u'2011-06-09T00:00:00+00:00',  
  u'id': 7,  
  u'metadata': {u'values': {u'OS': u'windows',  
                             u'size': u'11000'}}},
```

```
  u'name': u'Windows',
```

```
  u'progress': 100,
```

```
  u'status': u'ACTIVE',
```

```
  u'updated': u'2011-09-12T14:47:12+00:00'}
```

```
In [6]: c.create_server('mywin1', 3, 5)
```



Sights

Live Demo



Live Demo

- ◆ Prepare and upload Image from local template VM
- ◆ Spawn compute cluster to run MPI app
- ◆ Make local modifications and repeat

- ◆ ... What if it was over a 3G connection?
 - ➔ Time needed to upload 1GB Image file? 😊
 - ➔ Time needed to prepare and spawn virtual nodes?



Live Demo

- ◆ Prepare and upload Image from local template VM
- ◆ Spawn compute cluster to run MPI app
- ◆ Make local modifications and repeat

- ◆ ... What if it was over a 3G connection?
 - ➔ Time needed to upload 1GB Image file? 😊
 - ➔ Time needed to prepare and spawn virtual nodes?



Upcoming

Current and Upcoming features

- ◆ Now: Alpha2
 - ➔ Common user base, custom user images on Pithos+
- ◆ short-term: Synnefo v0.12, Beta
 - ➔ Ultra-lightweight VMs on Archipelago with RADOS backend
- ◆ medium-term
 - ➔ Volumes: clonable / snapshottable / attachable disks
 - ➔ Network and storage hotplugging
- ◆ Upcoming beta in fully populated datacenter

Opensource



Opensource

◆ Synnefo: Cyclades / Pithos+ / Astakos

➔ <https://code.grnet.gr/projects/synnefo>

➔ <https://code.grnet.gr/projects/pithos>

➔ <https://code.grnet.gr/projects/astakos>

◆ kamaki

➔ <https://code.grnet.gr/projects/kamaki>



Opensource

◆ Synnefo: Cyclades / Pithos+ / Astakos

➔ <https://code.grnet.gr/projects/synnefo>

➔ <https://code.grnet.gr/projects/pithos>

➔ <https://code.grnet.gr/projects/astakos>

◆ kamaki

➔ <https://code.grnet.gr/projects/kamaki>

pip install or **apt-get install** everything!



 okeanos

<http://okeanos.io>

Thank You!

Questions?

