



<http://oceanos.grnet.gr>

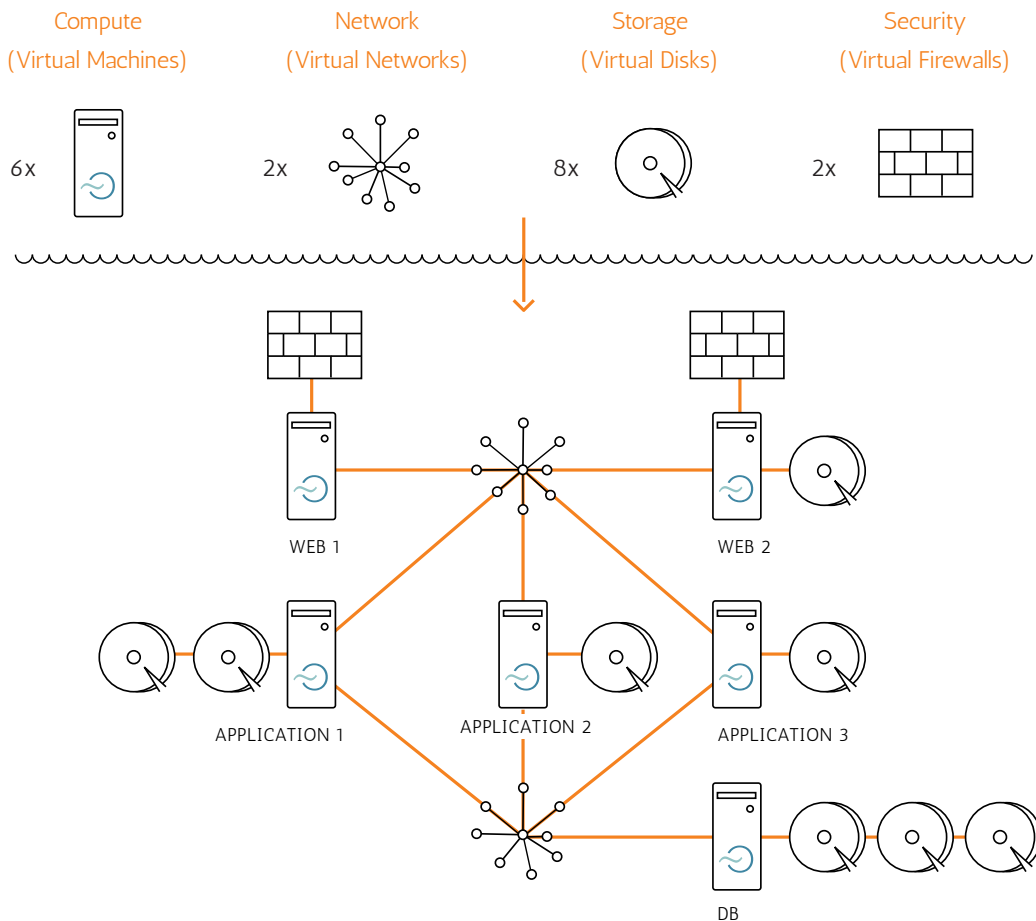
# ~oceanos Datacenter Operating System

The ~oceanos Datacenter Operating System is a complete software solution for the rapid and easy deployment of IaaS cloud services, either public or private, hosted on existing or new datacenters.

~oceanos technology brings the 'plug and play' model to the datacenter resources. Due to its unique LEGO-style philosophy, all resources can be easily managed from a simple cockpit. Existing hardware infrastructures are commoditized and offered as the building bricks for any tailor-made information system, in a few minutes.

Image 1: Multi-tiered topology using ~oceanos-provided cloud blocks

~oceanos transforms all existing hardware resources to commodity virtual components:



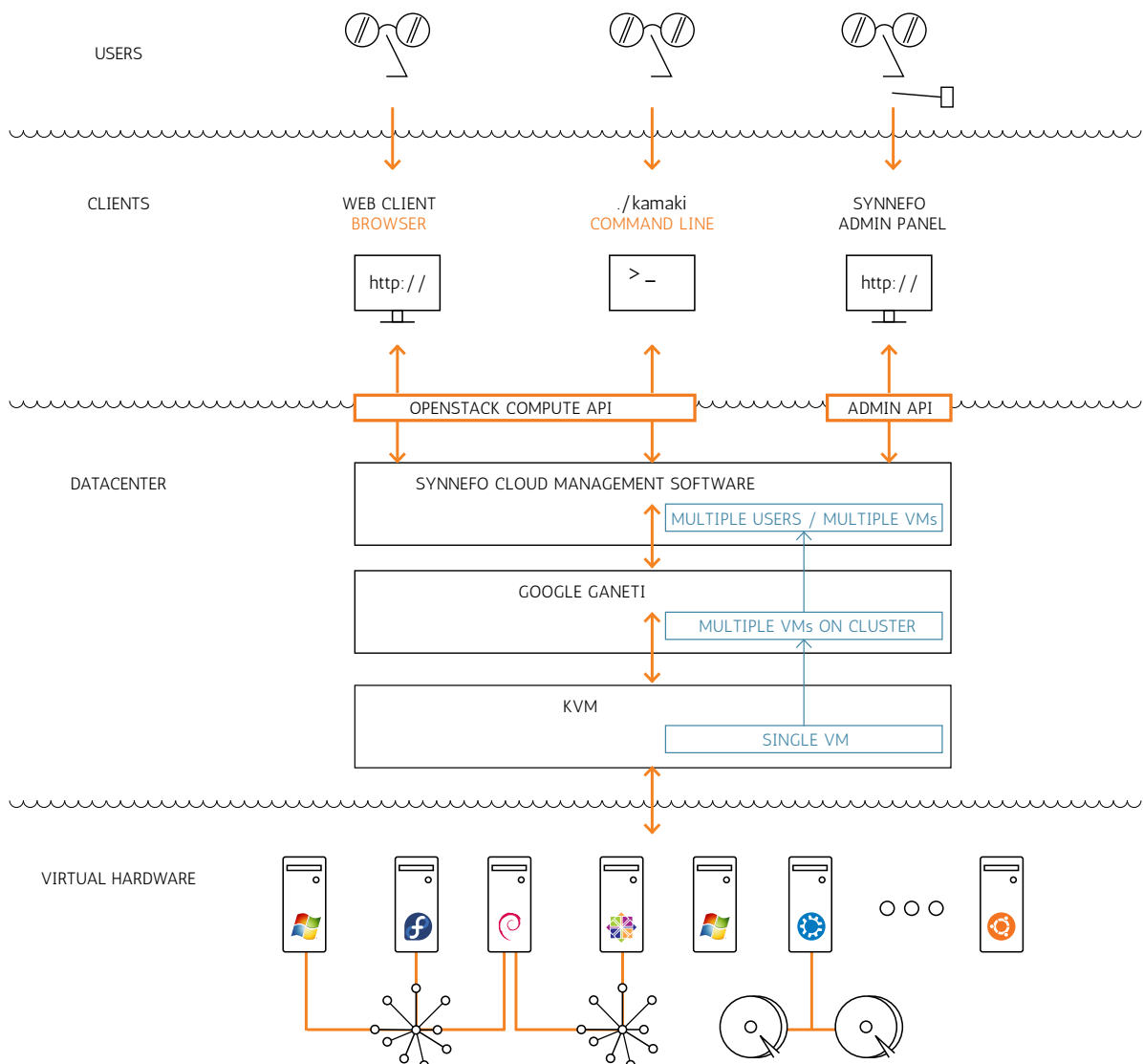
# Architectural Overview

Essentially, -okeanos enables users to build compute and storage clouds, making more efficient use of their processing and storage resources, within their Datacenter, to handle a variety of workloads. They can prepare in a few minutes, numerous of custom ordered machines, with all flavors of server OS, connected in any possible way, secured and easily managed.

-okeanos offers a no-hassle, intuitive Web UI for all custom architectures. -okeanos can be programmatically accessed, via command line interface plus a RESTful API. Thus integration to existing environments or usage of existing tools is done seamlessly. -okeanos encapsulates the Synnefo Cloud Management Layer and the Google Ganeti Cluster management software on top of KVM virtualized Computing resources. The API is a superset of the OpenStack Compute API to allow use of 3rd party tools with -okeanos

Image 2: Architectural overview of an -okeanos deployment

The main software layers comprising an -okeanos deployment are outlined in the following figure.



# Key Features

~okeanos provides the following key features:

## KVM-based Virtual Machines

~okeanos supports KVM-based VMs, managed by Google Ganeti. KVM does full system virtualization, and supports Microsoft Windows, Linux, and BSD deployments inside its VMs. So far, server Images for Red Hat Enterprise Linux / CentOS, Fedora, Debian Linux, Ubuntu/Kubuntu, and Microsoft Windows Server 2008R2 have been tested extensively inside ~okeanos deployments, using virtio-based storage and network drivers for minimal virtualization overhead.

## Google Ganeti for VM control

~okeanos uses Google Ganeti for VM management at the backend. The Ganeti-based control plane ensures stability, redundancy, manageability, upgradability and above all production readiness. Ganeti employs a simple, clean design with readable, well-commented and well-maintained code. Having a distinct, well-defined VM control plane allows ~okeanos deployments to benefit from the evolution process of Ganeti and provides for cleanly-separated layers. Ganeti gives a competitive advantage regarding VM migrations, intelligent resource allocation and handling of physical node downtimes.

## Clean RESTful API

Every bit of functionality provided by ~okeanos is exported to end users via a clean, RESTful API called the ~okeanos API. The ~okeanos API is a superset of OpenStack Compute API v1.1. Following an open standard ensures compatibility with a multitude of third-party cloud management tools and lowers the barrier to entry for migration of existing software deployments on the cloud.

## Intuitive Web UI

~okeanos comes with a clear, simple Web UI through which the user may quickly provision new and manage existing compute, network and storage resources. The Web UI is a client-side Javascript/jQuery application using the ~okeanos API behind the scenes, which means two things: (a) the Web UI runs on the client

side, eliminating unnecessary server roundtrips, (b) The API implementation is always up-to-date, with all functionality available both programmatically and over the UI.

## Command-line management

When the need arises to provision and manage resources automatically and in bulk, the ./kamaki command-line tool can be used to perform low-level administrative tasks. ./kamaki is just another client to the ~okeanos API, targeted to advanced end users and developers.

## Public Networking

~okeanos supports full IPv4 and IPv6 connectivity to the public Internet for its VMs. The network implementation is deployment-specific, behind Ganeti, and may be customized extensively to the customer's individual needs. A reference ~okeanos implementation supports host-based routing for multiple IP address pools, with minimal overhead, and no Network Address Translation. Eliminating the need for NAT allows VMs to migrate freely between physical hosts, without introducing SPOFs.

## Private Networking

~okeanos provides virtual Ethernets as a separate resource, giving the user freedom to create arbitrary network topologies of interconnected VMs, e.g., for multi-tiered deployments of enterprise software. Private networks are supported by the API and are exposed all the way to the UI. Each private network is an isolated Ethernet segment, carrying raw L2 Ethernet frames. This gives unrestricted choice of IP addressing schemes, allows running own DHCP services, and supports non-IP traffic as well. VMs see a separate virtual Ethernet NIC for each private LAN they are part of.

## Out-of-the-box Firewalling

The user may protect each public IPv4/IPv6 interface with a virtual firewall, choosing from a number of predefined firewall configurations. Firewalling is provided as a virtual appliance by the infrastructure and works independently from the guest OS running inside a VM.

## Bridging of cloud-based with physical resources

One cannot expect a complete deployment of physical resources to be migrated overnight to the cloud. ~okeanos exploits private networking functionality to form secure bridges between virtual networks and your existing physical network in your server room or datacenter, essentially bringing cloud-based resources right next to your physical servers. Need your virtual resources to allocate IP addresses from your physical DHCP server? Now that's possible.

## Self-healing distributed storage backend

VM storage volumes are physically stored as objects in a distributed, redundant, object-based storage backend. The storage backend is deployed in commodity physical nodes, with no need for proprietary hardware or custom interconnects; it is a distributed, shared-nothing

architecture, with no SPOFs. Storage bandwidth and capacity scales with the number of storage nodes. Nodes are added and removed in a live system, with dynamic object replication and automatic rebalancing. Having a shared-storage backend allows for seamless VM migrations among physical nodes.

**Intelligent Storage with snapshots and clones**

Efficient VM provisioning is key to delivering elastic scaling of cloud resources. A custom volume composition layer in *-okeanos* provides for low-overhead snapshots of VM storage volumes

and creation of clones from them. In this scheme, VM images correspond to snapshots in the storage backend; everybody can mass deploy long-lived, persistent VMs running on cloned storage volumes, in a matter of seconds. Users can keep point-in-time snapshots of their work; they can revert to them and start over, or follow a diverging path in history by cloning them to a new VM. Storage functionality is exported all the way up to the UI via a clean RESTful API for volume management.

# Deployment details

Image 3: Main components of an *-okeanos* deployment

