Virtual Datacenter
or
Virtualization in the datacenter

(OpenStack)

Larry Rudolph
A merge of several public presentations

- Rackspace & NASA started it off, and OpenStack has grown dramatically

- All possible because of virtualization

- Certainly evolving towards
  - Virtual Data Center / Virtual Cloud
  - Personal Cloud (?)
About OpenStack
The Pieces of OpenStack

OpenStack Compute (Nova)

OpenStack Object Storage (Swift)

OpenStack Image Service (Glance)
The Pieces of OpenStack

- Dashboard
- Identity Management
- Networking
- Load balancers
- Database
- Queueing
Some Stats

• 1,500+ active participants

• 17 countries represented at Design Summit

• 60,000+ downloads

• Worldwide network of user groups (North America, South America, Europe, Asia and Africa)
OpenStack, the Cloud Operating System

Management Layer That Adds Automation & Control

![Diagram showing connections between apps, users, admins, and OpenStack](image)

- Connects to apps via APIs
- Self-service Portals for users
- Creates Pools of Resources
- Automates The Network
OpenStack in a nutshell

• A cloud operating system that turns datacenters into pools of resources – the next evolution from server virtualization

• Provides a management layer for controlling, automating, and efficiently allocating these resources

• Empowers operators, sys admins and end users via self-service portals ("I want AWS-type service!")

• Gives developers the capability to build cloud-aware applications via standard APIs
Why Service Providers Are Adopting

- Originated by Rackspace and NASA
- Designed to scale cost effectively
- Emerging standard backed by large ecosystem
- Open source with no lock-in or license
- No desire to build proprietary clouds
Challenges of Cloud Interoperability
OpenStack Goes Beyond A Single Datacenter…

Enterprise Private Clouds
Run Cloud Operating Systems

→

Public Clouds Run Cloud Operating Systems
OpenStack Goes Beyond A Single Datacenter…

Imagine Having an Open, Common Platform Across Clouds…

Seamlessly transporting workloads

This is true Cloud Federation!
An Open, Common Platform is Here

**Private Clouds**
Run OpenStack Software In Your Own Corporate DC or Colocation Facility

**Public Clouds**
Run Highly Scalable Cloud Software Proven by Some of the World’s Largest Cloud Providers
Well… almost

- Today, there are still technical challenges that have to be addressed before users of OpenStack clouds can reliably create cross-cloud compatible applications.
Key Use Case

• Anne is running an application on a Private Cloud. She is now experiencing higher than average workload, so she launches several dozen application instances on a Public Cloud.
Challenges

• Even if we assume the same API amongst cloud providers, there are a variety of challenges that we have identified that need to be addressed if we want to enable Anne’s use case.
Challenges

- If the implementation of all clouds were the same, accomplishing federation would be easy.
- However, public and various private clouds are likely to have different requirements that will dictate different internal technology choices, even if they share the exact same external API.

<table>
<thead>
<tr>
<th>Private OpenStack Cloud</th>
<th>Public OpenStack Cloud</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hypervisor: KVM</td>
<td>Hypervisor: XenServer</td>
</tr>
<tr>
<td>Image Format: qcow2</td>
<td>Image Format: VHD</td>
</tr>
<tr>
<td>Networking: FlatDHCP</td>
<td>Networking: Flat + multi-nic</td>
</tr>
<tr>
<td>Instance Auth: Public Key</td>
<td>Instance Auth: Password</td>
</tr>
</tbody>
</table>
Challenges

• Thus, in order to achieve interoperability between OpenStack clouds, points of compatibility must be defined for:
  • Image Format
    • A variety of disk formats are used in different clouds (qcow2, vhd, raw) – which must be supported by all? How can they be constructed such that they work on multiple hypervisors?
  • Instance Networking
    • Specific deployments may use different networking models internally, may have different numbers of nics, etc
  • Instance Configuration
    • How should instances configure authentication, handle user specific data, installation of paravirtualization drivers, etc?
Image Format

• Depending on choice of hypervisor, and specific desired operational characteristics, deployers are likely to prefer the use of ‘non standard’ internal image formats.

• For interoperability, the community needs to agree on a common “Golden” format that all clouds must be able to support.
  • All clouds must be able to import/export the “Golden” format. Internally, clouds will likely decide to use a format that is optimal for their own configuration.

• Idea: use RAW as a golden image format
Instance Networking

• At present, there are a few techniques that are available to configure networks in OS
  • DHCP
    • What about multi-nic?
  • Flat injection
  • Cloud configuration drive?

• We need to define a standard method for instance networking that will work in a wide variety of deployment scenarios
  • What happens when a cloud instance with one DHCP configured nic gets migrated to a public cloud with 2 nics?

• Idea: Use DHCP for basic networking. Use a guest agent + metadata service for multi-nic and other advanced networking functions
Instance Configuration

• At present, there are a variety of techniques that are available in OS for instance configuration
  • EC2 Metadata
  • File injection
  • Cloud configuration drive
• Idea: OS Metadata service + OS guest agent
Putting it all together – example flow

• Anne exports an image from Private cloud in RAW format
• Anne imports her image into the Public Cloud
  • Internally, this instance is converted to an optimized format
• Anne boots her image
• Her instance gets an ip address for the primary interface
• OS guest agent launches, and connects to a metadata service to get additional information
  • Configures a secondary interface
  • Installs Paravirtualization drivers (like xenserver tools)
  • Executes user data
• Anne accesses her instance and launches her app
An Introduction to OpenStack and its use of KVM

Daniel P. Berrangé <berrange@redhat.com>
About me

- Contributor to multiple virt projects
- Libvirt Developer / Architect 8 years
- OpenStack contributor 1 year
- Nova Core Team Reviewer
- Focused on Nova libvirt + KVM integration
Talk Structure

- Introduction to OpenStack services
- Compute service architecture
- Guest boot sequence
- Recent developments
What is OpenStack?

- Public or private cloud
- Self-service user API and dashboard
- Apache 2.0 licensed
- Broad community contribution
What is in OpenStack?

- Compute (Nova)
- Network (Neutron)
- Image storage (Glance)
- Block storage (Cinder)
- Object storage (Swift)*
- Identity (Keystone)*
- Metering (Ceilometer)*
- Orchestration (Heat)*
- Dashboard (Horizon)*

* not discussed in this presentation
What is Nova?

- Execution of compute workloads
- Technology agnostic
  - Virtual machine or container virt
- Virtualization agnostic
  - Libvirt (KVM, QEMU, Xen, LXC), XenAPI, Hyper-V, VMware ESX, PowerVM, Docker, Bare-metal
What is Glance?

- Write-once, read-many storage of images
- Image copied on use by Nova
- Format agnostic
  - eg raw, qcow2, etc
- Metadata properties
  - eg specify virtual hardware preferences
What is Cinder?

- Persistent block storage
- Multiple storage backends
  - eg LVM, RBD, Gluster, Sheepdog, ...+ more...
- Exposed to compute host via iSCSI
- Optional direct access by compute
  - Gluster
- Pre-requisite for live migration
What is Neutron?

- Network infrastructure management
- Concepts
  - Networks
  - Routers
  - Subnets
  - Ports
- Multiple technologies
  - OpenVSwitch, Linux Bridge, Vendor plugins
Nova architecture

- Concepts
  - Instances
  - Flavours
  - Virt drivers
  - Security group

- Dual APIs
  - OpenStack REST
  - EC2 compatible REST
Nova architecture (simple)
Nova architecture (scaling)

- REST API
  - Load Balancer
    - nova-api
      - nova-scheduler
      - nova-conductor
  - AMQP
    - nova-compute
    - Libvirt+KVM
- DB
Nova cells

- Partition cloud infrastructure
  - Resilience within a data center
  - Scale out across data centers
  - Technology variation (e.g., KVM vs Hyper-V)
Nova cells architecture (Part 1)
Nova cells architecture (Part 2)
Nova cells architecture (Part 3)
Nova scheduler

- Places instances on compute hosts
- Pluggable filtering rules
  - CPU model / architecture
  - Virtualization type
  - PCI device availability
  - CPU, RAM, Disk usage
  - Trusted boot (TXT)
  - +more...
Nova conductor

• Mediates database access
• No database access from compute hosts
  - Compute hosts relatively untrusted / high risk
  - Scalability bottleneck for database
• Compute hosts issues RPC calls
• Conductor updates database state
Nova graphics proxy

- No direct compute access for VNC/SPICE
- Nova VNC/SPICE websockets proxy
- HTML5 VNC/SPICE browser clients
- Obtain auth token via REST API
- Pass to websockets proxy to authenticate
- Data proxied between compute & proxy
Instance boot step 1

1

Identity
Dashboard
Orchestration
Metering

Object Storage
Image Service
Block Storage
Networking
Compute
Instance boot step 2

1. Identity
2. Dashboard
   - Orchestration
   - Metering
      - Object Storage
      - Image Service
      - Block Storage
      - Networking
      - Compute
Instance boot step 3

1. Identity
2. Dashboard
3. Orchestration
4. Metering

- Object Storage
- Image Service
- Block Storage
- Networking
- Compute
Instance boot step 4

1. Object Storage
2. Image Service
3. Block Storage
4. Networking

Identity, Dashboard, Orchestration, Metering

Compute
Instance boot step 6

1. Human figure
2. 1
3. 3
4. 4
5. 5
6. Compute

Services:
- Identity
- Dashboard
- Orchestration
- Metering
- Object Storage
- Image Service
- Block Storage
- Networking
Nova KVM config part 1

- CPU
  - Named model or host model or host passthrough
- NIC model
- Disk bus type
- PCI device assignment
- Serial console x2 (1x log, 1x interactive)
- Disk devices
Nova KVM config part 2

- SMBios info
- CPU pinning
- VNC or SPICE
- QEMU + SPICE agents
- Clock (PIT, RTC) parameters
- Scheduler, disk, network tunables
New in Havana

• Released Thursday 17th Oct
• Notable features
  - Block storage backend migration
  - Store images in RBD
  - Gluster native boot
  - QEMU guest agent assisted snapshots
  - PCI device assignment
Coming in Icehouse

- Target 17\textsuperscript{th} Oct + 6 months
- Planning summit in Hong Kong Nov 4\textsuperscript{th}-8\textsuperscript{th}
- Notable blueprints
  - VM ensembles
  - VM migration with storage
  - Live snapshots (disk + RAM)
  - Host reservation (user request entire host)
OpenStack Tutorial
IEEE CloudCom 2010

Bret Piatt
Community Stacker
Twitter: @bpiatt
Application Platforms Undergoing A Major Shift

70’s – 80’s
Mainframe Era

90’s-2000’s
Client Server Era

2010-beyond
Cloud Era

2010 IT budgets aren’t getting cut..
..but CIOs expect their spend to go further.

- #1 Priority is Virtualization
- #2 is Cloud Computing

[Based on a Gartner Study]
Overview of Rackspace

- Founded in 1998
  - Publicly traded on NYSE: RAX
  - 120,000+ customers

- $628m revenue in 2009 across two major businesses
  - Dedicated Managed Hosting
  - Cloud Infrastructure & Apps (Servers, Files, Sites, Email)

- Primary focus on customer service ("Fanatical Support")
  - 3,000+ employees

- 9 datacenters in the US, UK and Hong Kong
  - 65,000+ physical servers
Rackspace Cloud: 3 Products with Solid Traction

- **Compute:** Cloud Servers
  - Virtualized, API-accessible servers with root access
  - Windows & Linux (many distros)
  - Sold by the hour (CPU/RAM/HDD) with persistent storage
  - Launched 2009
  - Based on Slicehost
  - Xen & XenServer HVs

- **Storage:** Cloud Files
  - Launched 2008
  - Object file store
  - v2.0 in May 2010

- **PaaS:** Cloud Sites
  - Launched 2006
  - Formally Mosso
  - Code it & Load it: .Net, PHP, Python apps autoscaled

Source: Guy Rosen (http://www.jackofallclouds.com)
Active Ecosystem on Rackspace APIs

- Open ReST APIs released July 2009 (Creative Commons License)
- Included in major API bindings: Libcloud, Simple Cloud, jclouds, σ-cloud
- Supported by key cloud vendors and SaaS services
OpenStack: The Mission

"To produce the ubiquitous Open Source cloud computing platform that will meet the needs of public and private cloud providers regardless of size, by being simple to implement and massively scalable."
OpenStack History

2005 2010

March
- Rackspace Cloud developed
- Rackspace Decides to Open Source Cloud Software

May
- NASA Open Sources Nebula Platform

June
- OpenStack formed b/w Rackspace and NASA

July
- Inaugural Design Summit in Austin
OpenStack History

2011

July: OpenStack launches with 25+ partners

October: First ‘Austin’ code release with 35+ partners

November: First public Design Summit in San Antonio

February: Second ‘Bexar’ code release planned
OpenStack Founding Principles

- Apache 2.0 license (OSI), open development process
- Open design process, 2x year public Design Summits
- Publicly available open source code repository
- Open community processes documented and transparent
- Commitment to drive and adopt open standards
- Modular design for deployment flexibility via APIs
Community with Broad Commercial Support

- Dell
- Intel
- NTT Data
- Citrix
- anso Labs LLC
- NASA
- cloudscaling
- Internap
- Microsoft
- Rightscale
- Morph
- Rackspace Hosting
- Opscode
- Nicira
- Puppet Labs
- Iomart
- Scalr
- Cirrascale
- SoftLayer
- Enterprise DB
- Gigaspaces
- Zenoss
- AMD
- Autonomic Resources
- Enstratus
- Limelight Networks
- Zuora
- Spiceworks
- sonian
- Cloudkick
- FathomDB
- Vyatta
- Path
- Intalio
- Riptano
- OpenStack
OpenStack Isn't Everything

- Consultants
- Business Process Automation

- Database Engineers
- Operating System Technicians
- Systems Security Professionals
- Network Experts

- Servers, Firewalls, Load Balancers
- Operating Systems
- Storage
- Management Tools
- Virtualization

- Data Center
- Networking
- Power
Software to provision virtual machines on standard hardware at massive scale

OpenStack Compute

creating open source software to build public and private clouds

Software to reliably store billions of objects distributed across standard hardware

OpenStack Object Storage
OpenStack Release Schedule

- **Bexar:** February 3, 2011
  - OpenStack Compute ready for enterprise private cloud deployments and mid-size service provider deployments
  - Enhanced documentation
  - Easier to install and deploy

- **Cactus:** April 15, 2011
  - Community gathers to plan for next release, likely Fall 2011
  - OpenStack Compute ready for large service provider scale deployments
  - This is the ‘Rackspace-ready’ release; need to communicate Rackspace support and plans for deployment

- **Design Summit:** April TBA 2011
Business Prerequisites

- Must be willing to change
- Virtualization = Consolidation
- Cloud = Automation
- Desire to integrate
Technical Prerequisites

Cloud has a minimum practical scale
- Proof of concept: 5+ servers
- Pilot: 20+ servers
- Production: 40+ servers

Datacenter made “Cloud ready”
- Networking
- Power

Equipment built for cloud
- CPUs with virtualization and power management support
- Storage platforms with flexible workload capabilities
Cloud Ready Datacenter Requirements

- Homogeneous Configuration
- Increased Power Density
- Fat Tree / Mesh Networks
- “Lights Out” Operation
Bootstrapping the Host Machines

1. Physical Hardware
   - Rack
   - Cable

2. Remote Management
   - Dell DRAC
   - HP ILO
   - IPMI

3. Host Networking
   - Static
   - DHCP

4. Host Seed OS Install
   - BOOTP / TFTP
   - GPXE

5. Host OS Install
   - Preseed
   - Kickstart
   - YAST

6. Post OS Configuration
   - Puppet
   - Chef
   - CFEngine
Zettabyte

1,000 Exabytes
1,000,000 Petabytes
All of the data on Earth today
(150GB of data per person)
Zettabyte

2% OF THE DATA ON EARTH IN 2020
Data Must Be Stored Efficiently

If we stored all of the global data as “an average” enterprise..

<table>
<thead>
<tr>
<th>ITEM</th>
<th>MONTHLY FIGURES</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENTERPRISE AVERAGE STORAGE COST</td>
<td>$1.98 PER GIGABYTE</td>
</tr>
<tr>
<td>WORLD GDP</td>
<td>$5.13 TRILLION</td>
</tr>
<tr>
<td>COST TO STORE A ZETTABYTE</td>
<td>$1.98 TRILLION</td>
</tr>
</tbody>
</table>

..it would take..

..38.5% of the World GDP!
Object Storage Summary

- Fully Distributed
- Commodity Hardware
- Features Optimized for Scale
- Data Protection in Software
- Not a Filesystem
- Augments SAN/NAS/DAS, Doesn’t Replace
Object Storage Key Features

- ReST-based API
- Data distributed evenly throughout the system
- Scalable to multiple petabytes, billions of objects

Account/Container/Object structure (not file system, no nesting) plus Replication (N copies of accounts, containers, objects)

No central database

Hardware agnostic: standard hardware, RAID not required
System Components

- **The Ring**: Mapping of names to entities (accounts, containers, objects) on disk.
  - Stores data based on zones, devices, partitions, and replicas
  - Weights can be used to balance the distribution of partitions
  - Used by the Proxy Server for many background processes

- **Proxy Server**: Request routing, exposes the public API

- **Replication**: Keep the system consistent, handle failures

- **Updaters**: Process failed or queued updates

- **Auditors**: Verify integrity of objects, containers, and accounts
System Components (Cont.)

- **Account Server**: Handles listing of containers, stores as SQLite DB
- **Container Server**: Handles listing of objects, stores as SQLite DB
- **Object Server**: Blob storage server, metadata kept in xattrs, data in binary format
  - Recommended to run on XFS
  - Object location based on hash of name & timestamp
Software Dependencies

Object Storage should work on most Linux platforms with the following software (main build target for Austin release is Ubuntu 10.04):

- Python 2.6
- rsync 3.0

And the following python libraries:

- Eventlet 0.9.8
- WebOb 0.9.8
- setuptools
- Simplejson
- Xattr
- Nose
- Sphinx
Evolution of Object Storage Architecture

Version 1: Central DB
(Rackspace Cloud Files 2008)

Version 2: Fully Distributed
(OpenStack Object Storage 2010)
Example Small Scale Deployment

5 Storage Nodes, $0.13 per GB (monthly)

490TB of disk, 160TB usable

18.5kVA, 35 RU, 5 “half cabinets”

$0.114 per GB OPEX (monthly)

$0.014 per GB CAPEX ($200k, 36 month refresh)

$0.38/GB (with 3 copies), < 20% of “average”
Example OpenStack Object Storage Hardware

5 Zones
2 Proxies per 25 Storage Nodes
10 GigE to Proxies
1 GigE to Storage Nodes
24 x 2TB Drives per Storage Node

Example Large Scale Deployment -- ManyConfigs Possible
Building an OpenStack Cloud
Compute
OpenStack Compute Key Features

- ReST-based API
- Asynchronous eventually consistent communication
- Horizontally and massively scalable
- Hypervisor agnostic: support for Xen, XenServer, Hyper-V, KVM, UML and ESX is coming
- Hardware agnostic: standard hardware, RAID not required
Cloud Controller: Global state of system, talks to LDAP, OpenStack Object Storage, and node/storage workers through a queue.

API: Receives HTTP requests, converts commands to/from API format, and sends requests to cloud controller.

OpenStack Compute

Glance: HTTP + OpenStack Object Storage for server images

Host Machines: workers that spawn instances

ATAoE/iSCSI
System Components

- **API Server**: Interface module for command and control requests
  - Designed to be modular to support multiple APIs
  - In current release: OpenStack API, EC2 Compatibility Module
  - Approved blueprint: Open Cloud Computing Interface (OCCI)
- **Message Queue**: Broker to handle interactions between services
  - Currently based on RabbitMQ
- **Metadata Storage**: ORM Layer using SQLAlchemy for datastore abstraction
  - In current release: MySQL
  - In development: PostgreSQL
- **User Manager**: Directory service to store user identities
  - In current release: OpenLDAP, FakeLDAP (with Redis)
- **Scheduler**: Determines the placement of a new resource requested via the API
  - Modular architecture to allow for optimization
  - Base schedulers included in Austin: Round-robin, Least busy
System Components (Cont.)

- **Compute Worker**: Manage compute hosts through commands received on the Message Queue via the API
  - Base features: Run, Terminate, Reboot, Attach/Detach Volume, Get Console Output
- **Network Controller**: Manage networking resources on compute hosts through commands received on the Message Queue via the API
  - Support for multiple network models
    - Fixed (Static) IP addresses
    - VLAN zones with NAT
- **Volume Worker**: Interact with iSCSI Targets to manage volumes
  - Base features: Create, Delete, Establish
- **Image Store**: Manage and deploy VM images to host machines
Hypervisor Independence

- Cloud applications should be designed and packaged abstracted from the hypervisor, deploy and test for best fit for your workload
- Manage application definition and workload, not the machine image
  - Configuration management
  - Abstract virtual machine definition

Open Virtualization Format

Document Number: DSP0243
Date: 2009-02-22
Version: 1.0.0
Network Models

- Private VMs on Project VLANs or Public VMs on flat networks
Network Details

- **Security Group**: Named collection of network access rules
  - Access rules specify which incoming network traffic should be delivered to all VM instances in the group
  - Users can modify rules for a group at any time
    - New rules are automatically enforced for all running instances and instances launched from then on
- **Cloudpipe**: Per project VPN tunnel to connect users to the cloud
  - **Certificate Authority**: Used for Project VPNs and to decrypt bundled images
  - **Cloudpipe Image**: Based on Linux with OpenVPN
Example OpenStack Compute Hardware (other models possible)

Public Network

Private Network (intra data center)

Server Groups
Dual Quad Core
RAID 10 Drives
1 GigE Public
1 GigE Private
1 GigE Management

Management
Example innovation: Simcloud

SimCloud

Fig 1. Design overview
Thank You!

Bret Piatt
Email: bret@openstack.org
Twitter: @bpiatt

Questions & Answers
How to Compare VMware & OpenStack™

REFERENCE CHECKLIST

Mirantis
www.mirantis.com
How to think about a comparison

- VMware solves for the enterprise; OpenStack solves for the cloud
  - Both technology and philosophy differ
  - Not an apples-to-apples comparison
- Select features can be compared side by side
  - This slide deck provides a reference checklist
- Complete discussion at http://www.mirantis.com/vmware-openstack
## VMWare & OpenStack

<table>
<thead>
<tr>
<th>VMWARE FEATURE</th>
<th>COMPARABLE OPENSTACK FEATURE</th>
</tr>
</thead>
<tbody>
<tr>
<td>vMotion</td>
<td>KVM Live Migration</td>
</tr>
<tr>
<td>DRS</td>
<td>Nova Scheduler</td>
</tr>
<tr>
<td>DPM</td>
<td>TBD</td>
</tr>
<tr>
<td>Storage vMotion</td>
<td>TBD</td>
</tr>
<tr>
<td>SDRS</td>
<td>Nova Scheduler</td>
</tr>
<tr>
<td>HA</td>
<td>TBD</td>
</tr>
<tr>
<td>FT</td>
<td>TBD</td>
</tr>
<tr>
<td>vShield</td>
<td>Nova-network service</td>
</tr>
<tr>
<td>vDS</td>
<td>Quantum/Nova-network service</td>
</tr>
<tr>
<td>NetI/O</td>
<td>Quantum</td>
</tr>
<tr>
<td>SIOC</td>
<td>TBD</td>
</tr>
<tr>
<td>Storage APIs</td>
<td>Nova-volume service</td>
</tr>
<tr>
<td>Data Recovery</td>
<td>TBD</td>
</tr>
</tbody>
</table>

*MIRANTIS Engineering for OpenStack™. All rights reserved 2005 – 2012*
VMotion & KVM Live Migration

- Live migrations of VMs w/0-downtime between different physical hosts w/ broad hardware compatibility options.
- Low-latency link required
  - (max 5 ms RTT).
- Metro vMotion allows migration over metro level networks (up to 10 ms RTT) by dynamically adjusting TCP buffer.
- Other limitations include:
  - shared storage (VMFS, SAN, shared NAS) required
  - source and target nodes connected to single L2 segment
  - CPUs compatibility required on source and target.
- Live migration w/up to 2 seconds service suspend
- Migration without shared storage supported w/libvirt block migration feature
Distributed Resource Scheduler

- Manages workload ‘intelligently’ by grouping virtual machines; pooling resources of physical hosts and prioritizing VMs in these pools; and moving VMs between physical hosts in accordance to vMotion rules.
- Supports affinity rules governing placement of virtual machines onto physical hosts.
- Collects usage metrics from physical hosts and VMs; generates optimization recommendations executed manually or automatically – on initial placement or continuously during runtime.
- Maintenance mode supports moving workload w/vMotion live migration out of physical host to perform maintenance (in auto or manual mode).

- Not supported in OpenStack
- Can be implemented using external monitoring, FilterScheduler and Live Migration
DRM

- Allows shut down of physical hosts not in use by VMs. Brings hosts up once workload increases and more physical hosts are required.

- Not supported in OpenStack
Storage vMotion

• Allows configuring datastore cluster instead of single store location.
• Provides recommendations for initial virtual disk placement (VMDK)
• Allows moving virtual disks between nodes in datastore cluster (e.g. for scheduled maintenance of a node)
• Supports affinity and anti-affinity rules for VMs/VMDKs
• Only persistent VMDKs, or raw device mapped (RDM) virtual disks can be migrated w/ Storage vMotion

• Not supported in OpenStack
• Can be implemented for LVM-based persistent volumes
SDRS

- Allows moving volume storage workloads over datastore cluster with Storage vMotion.
- Automated recommendations for moving VMDKs based on space utilization & I/O load of nodes in datastore cluster.
- Supports affinity/anti-affinity rules.
- Supports manual and auto execution of recommendations.

- Volume type-aware scheduler is in nova roadmap (scheduled for Folsom)
- Local storage of VMs resides on local disks of compute nodes
- nova-scheduler supports boot-from-volume feature that allows to use iSCSI persistent volume as boot VM device
**HA**

- Automated monitoring of physical servers availability, system health and restart of VMs on another or same physical server depending on the nature of the failure.
- Provides suggestions for optimal placing of restarted VMs and exposes health data to the user interface.

- VM-level HA is on OpenStack roadmap for Folsom release.
FT

- Allows seamless stateful failover of protected virtual machines, ensures that protected machines are always backed by standby copies.
- Hypervisor-level ‘mirroring’ of instructions (vLockstep) is used to create standby clones of protected VMs.
- Supported for single VCPU VMs only.

- Instructions mirroring is not supported by KVM
- Kemari project (instructions mirroring for QEMU-KVM) seems to be dormant
vShield

- Includes components; each is a separate virtual appliance.
- Zones component provides IP-based applic.-aware stateful firewall that recognizes IP addresses, ports, protocol (5-tuple), application type.
- Edge component provides connection on network edge, including 5-tuple firewall, NAT, DHCP server for VMs, site-to-site IPSec VPN, simple load balancing.
- App component provides firewall on virtual NIC level, w/traffic analysis and reporting.
- Endpoint component provides integrated anti-virus w/ability to scan VMs using introspection

- Supports per-VM firewall that uses 5-tuple to policy network access w/ Security Groups
- Network controller performs edge firewall, DHCP, DNS, NAT functions for VMs
- Cloudpipe allows VPN connection to VMs (OpenVPN)
- Opensource IDS/IPS can be integrated into network controller
vDS

- End-to-end physical and virtual network management w/ Cisco Nexus 1000V.
- Teaming algorithm for virtual switch port groups.
- Enables Net IO Control

- End-to-end physical and virtual network management in Quantum w/following plugins:
  - Cisco Nexus 1000V
  - Nicira virtual switch
  - Open vSwitch
  - Linux bridges
  - Ryu network operating system (designed for OpenStack specifically; still in early development stage)

- Nova network offers QoS API for Xen hypervisor (XenServer)
NetLOC

- Set and enforce network priorities (per VM) across the cluster.
- Bridge virtual and physical network QoS w/802.1q tagging.

- QoS support on per-tenant basis for Cisco plugin
- QoS framework exists and service can be implemented for other plugins
SIOC

- Supports and manages I/O queue to enforce priorities by dedicating more slots to VMs with higher Storage QoS set
- iSCSI prioritizing is not supported in Linux
Storage APIs

- VASA: allows to integrate storage solution management w/vCenter for management functionality
- VAAI: allows hardware-supported acceleration features: deduplication, thin provisioning, cloning blocks
- VAMP: allows I/O path selection to storage device (multipath)
- Must be supported by storage appliance

- Driver configuration allows utilization of back-end storage appliance advanced features, including CoW, thin provisioning and deduplication (Nexenta); no API provided by OpenStack
Data Recovery

- Provide ability to backup snapshots of VMs to any virtual datastore supported
- Automatically deduplicate backup snapshots, incrementally update previous backups
- Nova-compute provides VM snapshot capability
- Implementation required for scheduled backups, automated backup upload w/Nova API
- Existing opensource backup solutions can be integrated for deduplication, incremental backups etc
VMware and OpenStack

www.mirantis.com/vmware-openstack