What’s in a cloud?
An Overview of Virtualization and Openstack

2014-03-24
Demistifying “The Cloud”

There’s a lot of talk about Cloud Computing - but what do we even mean when we say “Cloud?”
Demistifying “The Cloud”

There’s a lot of talk about Cloud Computing - but what do we even mean when we say “Cloud?”

- Software-as-a-Service?
Demistifying “The Cloud”

There’s a lot of talk about Cloud Computing - but what do we even mean when we say “Cloud?”

- Software-as-a-Service?
- Platform-as-a-Service?
Demistifying “The Cloud”

There’s a lot of talk about Cloud Computing - but what do we even mean when we say “Cloud?”

- Software-as-a-Service?
- Platform-as-a-Service?
- Infrastructure-as-a-Service?
Demistifying “The Cloud”

There’s a lot of talk about Cloud Computing - but what do we even mean when we say “Cloud?”

- Software-as-a-Service?
- Platform-as-a-Service?
- Infrastructure-as-a-Service?
- Magic?
Demistifying “The Cloud”

There’s a lot of talk about Cloud Computing - but what do we even mean when we say “Cloud?”

• Software-as-a-Service?
• Platform-as-a-Service?
• Infrastructure-as-a-Service?
• Magic?
What exactly is a cloud?

NIST Definition

Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.
What exactly is a cloud?

**NIST Definition**

Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.

Cloud means:

- On-demand (self service)
- Elasticity (easily scale up/down)
- Multi-tenancy
In the most basic cloud-service model, providers of IaaS offer computers - physical or (more often) virtual machines - and other resources.

Wikipedia
Infrastructure-as-a-Service (IaaS)

**Wikipedia**

In the most basic cloud-service model, providers of IaaS offer computers - physical or (more often) virtual machines - and other resources.

Other resources can be:

- Network (virtual networks, Load Balancing, etc...)
- Storage (Object and Block)
In the most basic cloud-service model, providers of IaaS offer computers - physical or (more often) virtual machines - and other resources.

Other resources can be:

- Network (virtual networks, Load Balancing, etc...)
- Storage (Object and Block)
IaaS types

IaaS clouds come in two flavors: **public** and **private**
IaaS types

IaaS clouds come in two flavors: **public** and **private**

With a private IaaS cloud:

+ Can take full advantage of virtualization
IaaS types

IaaS clouds come in two flavors: **public** and **private**

With a private IaaS cloud:

- Can take full advantage of virtualization
- Own the hardware

- Have to manage cloud stack on top of infrastructure
IaaS types

IaaS clouds come in two flavors: public and private

With a private IaaS cloud:

+ Can take full advantage of virtualization
+ Own the hardware
- Own the hardware

- Have to manage cloud stack on top of infrastructure

With a public IaaS cloud:

+ Scaling & cost better (can go beyond your HW)
+ Don't own the hardware or the cloud stack
- Vendor lock-in

IaaS types

IaaS clouds come in two flavors: **public** and **private**

With a private IaaS cloud:

+ Can take full advantage of virtualization
+ Own the hardware
  - Own the hardware
  - Have to manage cloud stack on top of infrastructure

With a public IaaS cloud:

+ Scaling & cost better (can go beyond your HW)
+ Don’t own the hardware or the cloud stack
  - Don’t own the hardware
  - Vendor lock-in
IaaS types

IaaS clouds come in two flavors: **public** and **private**

With a private IaaS cloud:

+ Can take full advantage of virtualization
+ Own the hardware
  - Own the hardware
  - Have to manage cloud stack on top of infrastructure

With a public IaaS cloud:

+ Scaling & cost better (can go beyond your HW)
+ Don’t own the hardware or the cloud stack
  - Don’t own the hardware
  - Vendor lock-in
IaaS types

IaaS clouds come in two flavors: **public** and **private**

With a private IaaS cloud:

+ Can take full advantage of virtualization
+ Own the hardware
  - Own the hardware
  - Have to manage cloud stack on top of infrastructure

With a public IaaS cloud:

+ Scaling & cost better (can go beyond your HW)
IaaS types

IaaS clouds come in two flavors: **public** and **private**

With a private IaaS cloud:

+ Can take full advantage of virtualization
+ Own the hardware
  - Own the hardware
  - Have to manage cloud stack on top of infrastructure

With a public IaaS cloud:

+ Scaling & cost better (can go beyond your HW)
+ Don’t own the hardware **or** the cloud stack
IaaS types

IaaS clouds come in two flavors: **public** and **private**

With a private IaaS cloud:
- Can take full advantage of virtualization
- Own the hardware
  - Own the hardware
  - Have to manage cloud stack on top of infrastructure

With a public IaaS cloud:
- Scaling & cost better (can go beyond your HW)
- Don’t own the hardware **or** the cloud stack
  - Don’t own the hardware **or** the cloud stack
IaaS types

IaaS clouds come in two flavors: **public** and **private**

With a private IaaS cloud:
- Can take full advantage of virtualization
- Own the hardware
  - Own the hardware
  - Have to manage cloud stack on top of infrastructure

With a public IaaS cloud:
- Scaling & cost better (can go beyond your HW)
- Don’t own the hardware **or** the cloud stack
  - Don’t own the hardware **or** the cloud stack
  - Vendor lock-in
Virtualization
Virtual Machines

“Virtual machines have finally arrived. Dismissed for a number of years as merely academic curiosities, they are now seen as cost-effective techniques for organizing computer systems resources to provide extraordinary system flexibility and support for certain unique applications.”
“Virtual machines have finally arrived. Dismissed for a number of years as merely academic curiosities, they are now seen as cost-effective techniques for organizing computer systems resources to provide extraordinary system flexibility and support for certain unique applications.”

– Robert Goldberg, 1974
Virtual Machines

“Virtual machines have finally arrived. Dismissed for a number of years as merely academic curiosities, they are now seen as cost-effective techniques for organizing computer systems resources to provide extraordinary system flexibility and support for certain unique applications.”

– Robert Goldberg, 1974

• Main idea: transform a single machine into many
• Don’t emulate every instruction, emulate only privileged ones
• Virtual Machine Manager → VMs
A Virtual Machine Manager should exhibit:

- Efficiency
A Virtual Machine Manager should exhibit:

- Efficiency
  - Most instructions should be executed natively

---

1 Popek and Goldberg, “Formal Requirements for Virtualizable Third Generation Architectures,” 1974
A Virtual Machine Manager should exhibit:

- Efficiency
  - Most instructions should be executed natively
- Equivalence

---

1 Popek and Goldberg, “Formal Requirements for Virtualizable Third Generation Architectures,” 1974
A Virtual Machine Manager should exhibit:

- **Efficiency**
  - Most instructions should be executed natively

- **Equivalence**
  - Behavior in VM = Behavior in bare metal

---

1 Popek and Goldberg, “Formal Requirements for Virtualizable Third Generation Architectures,” 1974
A Virtual Machine Manager should exhibit:

- **Efficiency**
  - Most instructions should be executed natively

- **Equivalence**
  - Behavior in VM = Behavior in bare metal

- **Resource control**
A Virtual Machine Manager should exhibit:

- **Efficiency**
  - Most instructions should be executed natively

- **Equivalence**
  - Behavior in VM = Behavior in bare metal

- **Resource control**
  - VMM has complete control
Types of hypervisors

Examples:
Xen, VMWare ESX

**TYPE 1** native (bare metal)
Types of hypervisors

**TYPE 1** native (bare metal)

Examples: Xen, VMware ESX

**TYPE 2** (hosted)

Examples: Virtualbox, QEMU
x86 and Privilege Rings

- Only ring 0 can execute privileged instructions
- Linux/Windows: ring 0 (supervisor/kernel) and 3 (user)
Virtualization on x86

x86: reputation for being unfriendly to virtualization
• Not all privileged operations generate traps (e.g. popf)
Virtualization on x86

x86: reputation for being unfriendly to virtualization

- Not all privileged operations generate traps (e.g. `popf`)
- No MMU virtualization
Virtualization on x86

x86: reputation for being unfriendly to virtualization

- Not all privileged operations generate traps (e.g. `popf`)
- No MMU virtualization
- 1998 VMware: binary translation
Virtualization on x86

x86: reputation for being unfriendly to virtualization
- Not all privileged operations generate traps (e.g. `popf`)
- No MMU virtualization
- 1998 VMware: binary translation
Virtualization on x86

x86: reputation for being unfriendly to virtualization

- Not all privileged operations generate traps (e.g. `popf`)
- No MMU virtualization
- 1998 VMware: binary translation
Virtualization on x86

x86: reputation for being unfriendly to virtualization
- Not all privileged operations generate traps (e.g. popf)
- No MMU virtualization
- 1998 VMware: binary translation

Question

What are the advantages/disadvantages of this method?
Para-virtualization

Paravirtualization

Guest OS  Guest OS  Guest OS  Guest OS  Guest OS  Guest OS

Modification  Modification  Modification  Modification  Modification  Modification

Hypervisor

Host OS

HARDWARE
Para-virtualization

**Question**

What are the advantages/disadvantages of this method?
Hardware Assisted Virtualization (HAV)

In 2006, Intel VT-x and AMD AMD-V ⇒ VMM much easier to implement

- Allowed for an unmodified guest OS
- Introduced guest-mode execution
- When a guest performs a privileged instruction, trap to VMM via a VM Exit
VM Entry/Exit Example

Guest mode

Host mode
VM Entry/Exit Example

Guest mode

Host mode

Launch VM
VM Entry/Exit Example

Guest mode

Host mode

Launch VM

VM Entry
VM Entry/Exit Example

Guest mode

Host mode

Launch VM

VM Entry
VM Entry/Exit Example

Guest mode

Host mode

Launch VM

Allocate new memory

VM Entry
VM Entry/Exit Example

Guest mode

Host mode

Launch VM

Allocate new memory
VM Exit: PAGE_FAULT

VM Entry
VM Entry/Exit Example

Guest mode
Host mode
Launch VM
Allocate new memory
VM Exit: PAGE_FAULT
VM Entry
VM Entry/Exit Example

Guest mode
Launch VM

Host mode

Allocate new memory
VM Exit: PAGE_FAULT
VM Entry
VM Entry
VM Entry/Exit Example

Guest mode

Host mode

Launch VM

VM Entry

VM Entry

Allocate new memory

VM Exit: PAGE_FAULT
VM Entry/Exit Example

Guest mode

Host mode

Launch VM

Allocate new memory

VM Exit: PAGE_FAULT

Context Switch

VM Entry

VM Entry
VM Entry/Exit Example

Guest mode

Launch VM

Allocate new memory

VM Exit: PAGE_FAULT

VM Entry

Host mode

Context Switch

VM Exit: CR_ACCESS

VM Entry

Zak Estrada

What's in a Cloud? An Overview of Virtualization and Openstack
VM Entry/Exit Example

Guest mode
Launch VM

Host mode

Allocate new memory
VM Exit: PAGE_FAULT

VM Entry

Context Switch
VM Exit: CR_ACCESS

VM Entry

Guest mode
Host mode
Launch VM
VM Entry/Exit Example

Guest mode

Host mode

Launch VM

Allocate new memory
VM Exit: PAGE_FAULT

Context Switch
VM Exit: CR_ACCESS

...
VM Entry/Exit Example

Guest mode

Host mode

Launch VM

Allocate new memory

VM Exit: PAGE_FAULT

Context Switch

VM Exit: CR_ACCESS

...

...

...

VM Exit: VM_OFF

VM Entry

Deallocate VM

Turn off machine

VM Entry
A new datastructure: VMCS

- On VM Exit, contains the CPU & control state of a guest
- Relevant guest state is passed to the VMM (I/O port, etc...)

In 2008, VT-d features: more than CPU (EPT/IOMMU)

**Question**

What are the advantages/disadvantages of HAV?
HyperTap

- Can use HAV for robust security+reliability monitoring

---

Open-Source Virtual Machine Monitors (VMMs)
Xen

- Xen para-virt upstream in Linux since 3.0
- also supports HVM (HAV)
- Amazon EC2
• vCPUs scheduled as processes in Linux
• Paravirtualized drivers: virtio
• Default for Openstack
Thoughts on Virtualization

- Virtualization has overhead
- Many VMs run only one application
- Many people running Linux on Linux (e.g. KVM w/Linux guest)

Question
Can we do better?
Linux Containers (LXC)

- All “guests” share the same kernel - no “trap and emulate”
- Isolation via cgroups/namespaces

Question

What are the advantages/disadvantages of containers?
Virtualization Summary

Paravirtualization
- Xen

HAV
- KVM

Overhead/double work
- Linux Containers
Openstack
How to manage VMs?

- Xen provides mgmt, KVM doesn’t
- need for common interface
How to manage VMs?

- Xen provides mgmt, KVM doesn’t
- need for common interface

- Standardized method for managing multiple hypervisors
- Provides APIs/tools for managing VMs (mainly on Linux)

**Question**
- Network across entire environment?
- No rapid VM creation
- Weak multi-tenancy
- Not a cloud (no utility abstraction)
libvirt

- Provides APIs/tools for managing VMs (mainly on Linux)
- Can setup network/storage

Question:

- Network across entire environment?
- No rapid VM creation
- Weak multi-tenancy
- Not a cloud (no utility abstraction)
libvirt

- Provides APIs/tools for managing VMs (mainly on Linux)
- Can setup network/storage
- Utilizes local user/group services

What are the limitations of something like libvirt?
- Network across entire environment?
- No rapid VM creation
- Weak multi-tenancy
- Not a cloud (no utility abstraction)
libvirt

- Provides APIs/tools for managing VMs (mainly on Linux)
- Can setup network/storage
- Utilizes local user/group services

Question

- Network across entire environment?
- No rapid VM creation
- Weak multi-tenancy
- Not a cloud (no utility abstraction)
libvirt

- Provides APIs/tools for managing VMs (mainly on Linux)
- Can setup network/storage
- Utilizes local user/group services

**Question**

What are the limitations of something like libvirt?
libvirt

- Provides APIs/tools for managing VMs (mainly on Linux)
- Can setup network/storage
- Utilizes local user/group services

**Question**

What are the limitations of something like libvirt?

- Network across entire environment?
- No rapid VM creation
- Weak multi-tenancy
- Not a cloud (no utility abstraction)
Openstack Overview

- Originated in 2010, by Rackspace and NASA
- Rapid development - new (stable?) release every 6 months
- Modular architecture made up of multiple projects that provide a separate piece of an IaaS service, each with their own REST API
- Command-line and GUI interfaces
- Seeks to be to IaaS what Linux is to OSes
Interest in Openstack

Google Trends for Openstack

Date

Jun 2010
Dec 2010
Jun 2011
Dec 2011
Jun 2012
Dec 2012
Jun 2013
Dec 2013

Interest

0
20
40
60
80
100

Jun 2010
Dec 2010
Jun 2011
Dec 2011
Jun 2012
Dec 2012
Jun 2013
Dec 2013

What’s in a Cloud? An Overview of Virtualization and Openstack
Openstack Architecture

What's in a Cloud? An Overview of Virtualization and Openstack
Okay, that wasn’t so bad, why do you complain about the complexity of dealing with this?
Openstack Architecture

Question
That looks more complex than libvirt, why do I want to do this?

- Life is easier AFTER setting it up (mostly)
- Lots of automation available - not always if you want to do something special
Piece-by-piece: Identity (keystone)

- Provides authentication for users and applications

Example command to list known services:

```
root@logging:~# keystone service-list
```

<table>
<thead>
<tr>
<th>id</th>
<th>name</th>
<th>type</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>6dc206b8e4dd4b378123dcf354aebb8f</td>
<td>nova</td>
<td>compute</td>
<td>Openstack Compute Service</td>
</tr>
<tr>
<td>785fb6f0251742afa43a06b2c6f8d730</td>
<td>cinder</td>
<td>volume</td>
<td>Cinder Service</td>
</tr>
<tr>
<td>8fccfcdac18a4efc98b7f1e149c752ec</td>
<td>keystone</td>
<td>identity</td>
<td>OpenStack Identity Service</td>
</tr>
<tr>
<td>b1f1c9dd25864eca89aee1b52dc6e66d</td>
<td>glance</td>
<td>image</td>
<td>Openstack Image Service</td>
</tr>
<tr>
<td>f484b860fc8542928bb9d0db2a1b83b3</td>
<td>nova_ec2</td>
<td>ec2</td>
<td>EC2 Service</td>
</tr>
</tbody>
</table>
Piece-by-piece: Identity (keystone)

- Provides authentication for users and applications
- Contains a list of all the components in the system
Piece-by-piece: Identity (keystone)

- Provides authentication for users and applications
- Contains a list of all the components in the system
- Multiple backends (LDAP, MySQL, etc.)

Example command to list known services:
```
root@logging:~# keystone service-list
```

<table>
<thead>
<tr>
<th>id</th>
<th>name</th>
<th>type</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>6dc206b8e4dd4b378123dcf354aebb8f</td>
<td>nova</td>
<td>compute</td>
<td>Openstack Compute Service</td>
</tr>
<tr>
<td>785fb6f0251742afa43a06b2c6f8d730</td>
<td>cinder</td>
<td>volume</td>
<td>Cinder Service</td>
</tr>
<tr>
<td>8fccfcdac18a4efc98b7f1e149c752ec</td>
<td>keystone</td>
<td>identity</td>
<td>OpenStack Identity Service</td>
</tr>
<tr>
<td>b1f1c9dd25864eca89aee1b52dc6e66d</td>
<td>glance</td>
<td>image</td>
<td>Openstack Image Service</td>
</tr>
<tr>
<td>f484b860fc8542928bb9d0db2a1b83b3</td>
<td>nova_ec2</td>
<td>ec2</td>
<td>EC2 Service</td>
</tr>
</tbody>
</table>
Piece-by-piece: Identity (keystone)

- Provides authentication for users and applications
- Contains a list of all the components in the system
- Multiple backends (LDAP, MySQL, etc..)
- Also acts as the index for **API endpoints**
Piece-by-piece: Identity (keystone)

- Provides authentication for users and applications
- Contains a list of all the components in the system
- Multiple backends (LDAP, MySQL, etc.)
- Also acts as the index for API endpoints
  - e.g. You can connect to the compute service via http://cloud1:8774
Piece-by-piece: Identity (keystone)

- Provides authentication for users and applications
- Contains a list of all the components in the system
- Multiple backends (LDAP, MySQL, etc..)
- Also acts as the index for **API endpoints**
  - e.g. You can connect to the compute service via `http://cloud1:8774`
Piece-by-piece: Identity (keystone)

- Provides authentication for users and applications
- Contains a list of all the components in the system
- Multiple backends (LDAP, MySQL, etc.)
- Also acts as the index for **API endpoints**
  - e.g. You can connect to the compute service via http://cloud1:8774

Example command to list known services:

```
root@logging:~# keystone service-list
+----------------------------------+----------+----------+----------------------------+
| id | name | type | description                |
+----------------------------------+----------+----------+----------------------------+
| 6dc206b8e4dd4b378123dcf354aebb8f | nova | compute | Openstack Compute Service  |
| 785fb6f0251742afa43a06b2c6f8d730 | cinder | volume | Cinder Service             |
| 8fccfcf2c18a4efc98b7f1e149c752ec | keystone | identity | OpenStack Identity Service |
| b1f1c9dd25864eca89ae1b52dc6e66d | glance | image | Openstack Image Service    |
| f484b860fc8542928bb9d0db2a1b83b3 | nova_ec2 | ec2 | EC2 Service                |
+----------------------------------+----------+----------+----------------------------+
```
Piece-by-piece: Compute (nova)

- Originally contributed by NASA
• Originally contributed by NASA
• Communicates with the hypervisor to create/destroy virtual machines
Piece-by-piece: Compute (nova)

- Originally contributed by NASA
- Communicates with the hypervisor to create/destroy virtual machines
- Similar to job scheduler in HPC
Piece-by-piece: Compute (nova)

- Originally contributed by NASA
- Communicates with the hypervisor to create/destroy virtual machines
- Similar to job scheduler in HPC
  - Support for libvirt, ESX, Hyper-V, etc...
Piece-by-piece: Compute (nova)

- Originally contributed by NASA
- Communicates with the hypervisor to create/destroy virtual machines
- Similar to job scheduler in HPC
  - Support for libvirt, ESX, Hyper-V, etc...
- Each compute node connects to the others via a message bus (AMQP)
Piece-by-piece: Image (glance)

• Provides VM images to the compute service for booting
Piece-by-piece: Image (glance)

- Provides VM images to the compute service for booting
- Copies image from storage to the compute node, with a copy-on-write file for each instance
Piece-by-piece: Image (glance)

- Provides VM images to the compute service for booting
- Copies image from storage to the compute node, with a copy-on-write file for each instance
- Can use local storage or an object store
glance and nova with libvirt

• Compute server pulls the image from glance at instance start
• Instance disks are copy-on-write images (e.g. QCOW) off of this base
• QCOW images are deleted upon instance termination
Piece-by-piece: Block Storage (cinder)

- Block storage: persistent disk for instances
Piece-by-piece: Block Storage (cinder)

- Block storage: persistent disk for instances
- Can be used as boot disks or additional storage - dynamically

Zak Estrada
What’s in a Cloud? An Overview of Virtualization and Openstack
Piece-by-piece: Block Storage (cinder)

- Block storage: persistent disk for instances
- Can be used as boot disks or additional storage - dynamically
- Built on top of iSCSI, etc
• Block storage: persistent disk for instances
• Can be used as boot disks or additional storage - dynamically
• Built on top of iSCSI, etc
• Integration for enterprise block storage (NetApp, IBM, etc...)

Piece-by-piece: Block Storage (cinder)
Block storage: persistent disk for instances
Can be used as boot disks or additional storage - dynamically
Built on top of iSCSI, etc
Integration for enterprise block storage (NetApp, IBM, etc...)
NFS/Ceph/gluster as well
Piece-by-piece: Object Storage (swift)

• Originally contributed by Rackspace
Piece-by-piece: Object Storage (swift)

- Originally contributed by Rackspace
- Object storage: distributed object/file store
Piece-by-piece: Object Storage (swift)

- Originally contributed by Rackspace
- Object storage: distributed object/file store
- Similar to Amazon S3
Piece-by-piece: Object Storage (swift)

- Originally contributed by Rackspace
- Object storage: distributed object/file store
- Similar to Amazon S3
- Eventually consistent
Piece-by-piece: Object Storage (swift)

- Originally contributed by Rackspace
- Object storage: distributed object/file store
- Similar to Amazon S3
- Eventually consistent
- Online capacity adjustment
Piece-by-piece: Object Storage (swift)

- Originally contributed by Rackspace
- Object storage: distributed object/file store
- Similar to Amazon S3
- Eventually consistent
- Online capacity adjustment
- Versioned writes
Piece-by-piece: Object Storage (swift)

- Originally contributed by Rackspace
- Object storage: distributed object/file store
- Similar to Amazon S3
- Eventually consistent
- Online capacity adjustment
- Versioned writes
- Quota management

rsync-based replication
Piece-by-piece: Object Storage (swift)

- Originally contributed by Rackspace
- Object storage: distributed object/file store
- Similar to Amazon S3
- Eventually consistent
- Online capacity adjustment
- Versioned writes
- Quota management
- rsync based replication
Piece-by-piece: Network (neutron)

- In earlier OpenStack releases, networking was a part of compute.
In earlier OpenStack releases, networking was a part of compute.
Now it’s an own project (formerly Quantum).

- Provides an abstraction layer for creating virtual networks.
- Includes Open vSwitch, Linux Bridge, OpenFlow, Cisco, etc...
- Also provides services like load balancing.
- Part of the official release since 2012.2.
Piece-by-piece: Network (neutron)

- In earlier openstack releases, networking was a part of compute
- Now its own project (formerly quantum)
- Provides abstraction layer for creating virtual networks
Piece-by-piece: Network (neutron)

- In earlier openstack releases, networking was a part of compute
- Now its own project (formerly quantum)
- Provides abstraction layer for creating virtual networks
  - OpenVSwitch, Linux Bridge, OpenFlow, Cisco offerings, etc...
Piece-by-piece: Network (neutron)

- In earlier openstack releases, networking was a part of compute
- Now its own project (formerly quantum)
- Provides abstraction layer for creating virtual networks
  - OpenVSwitch, Linux Bridge, OpenFlow, Cisco offerings, etc...
- Also provides services like load balancing
Piece-by-piece: Network (neutron)

- In earlier openstack releases, networking was a part of compute
- Now its own project (formerly quantum)
- Provides abstraction layer for creating virtual networks
  - OpenVSwitch, Linux Bridge, OpenFlow, Cisco offerings, etc...
- Also provides services like load balancing
- Part of the official release since 2012.2
Tying it all together (simplified)
But wait, there’s more!

I only focused on the “core components” for a working cloud, but there are plenty more

• Ceilometer - Metering/Monitoring
• Heat - Automation (Similar to AWS CloudFormation)
• Savannah - Tighter Hadoop Integration
An actual production use case
ANL Magellan experiences

- CPU Performance was good
- I/O bandwidth was decent
- I/O latency was terrible

“Developer productivity went through the roof”
- Easy, low-cost experimentation
- IaaS good for data-centric workloads (bioinformatics)
  - Many folks that run hadoop do so on top of AWS
ANL Magellan experiences

- CPU Performance was good
- I/O bandwidth was decent
- I/O latency was terrible
- Not worth running tightly-coupled HPC workloads
- Need OS bypasses for infiniband and GPUs
ANL Magellan experiences

- CPU Performance was good
- I/O bandwidth was decent
- I/O latency was **terrible**
- Not worth running tightly-coupled HPC workloads
- Need OS bypass for infiniband and GPUs
- “Developer productivity went through the roof”
  - Easy, low-cost experimentation
- IaaS good for data-centric workloads (bioinformatics)
  - Many folks that run hadoop do so on top of AWS
My Experiences

- Set up 5 environments (Essex, Folsom x2, Grizzly, Havana)
My Experiences

- Set up 5 environments (Essex, Folsom x2, Grizzly, Havana)
  - Always run into some problem
My Experiences

• Set up 5 environments (Essex, Folsom x2, Grizzly, Havana)
  ○ Always run into some problem
  ○ Never the same problem twice

Automated tools are your friend (packstack, puppet)

• Really convenient for kernel hacking

• Developers like it a lot
  ○ Can give them root

Rapid Release

• Things are always changing
  + Noticeable improvement with each release

• VMs are just as reliable as with libvirt, mgmt not so much
My Experiences

- Set up 5 environments (Essex, Folsom x2, Grizzly, Havana)
  - Always run into some problem
  - Never the same problem twice
- Automated tools are your friend (packstack, puppet)
My Experiences

- Set up 5 environments (Essex, Folsom x2, Grizzly, Havana)
  - Always run into some problem
  - Never the same problem twice
- Automated tools are your friend (packstack, puppet)
- Really convenient for kernel hacking
My Experiences

• Set up 5 environments (Essex, Folsom x2, Grizzly, Havana)
  ○ Always run into some problem
  ○ Never the same problem twice
• Automated tools are your friend (packstack, puppet)
• Really convenient for kernel hacking
• Developers like it a lot
My Experiences

- Set up 5 environments (Essex, Folsom x2, Grizzly, Havana)
  - Always run into some problem
  - Never the same problem twice
- Automated tools are your friend (packstack, puppet)
- Really convenient for kernel hacking
- Developers like it a lot
  - Can give them root
My Experiences

• Set up 5 environments (Essex, Folsom x2, Grizzly, Havana)
  ◦ Always run into some problem
  ◦ Never the same problem twice
• Automated tools are your friend (packstack, puppet)
• Really convenient for kernel hacking
• Developers like it a lot
  ◦ Can give them root
• Rapid Release
My Experiences

• Set up 5 environments (Essex, Folsom x2, Grizzly, Havana)
  ○ Always run into some problem
  ○ Never the same problem twice

• Automated tools are your friend (packstack, puppet)

• Really convenient for kernel hacking

• Developers like it a lot
  ○ Can give them root

• Rapid Release
  — Things are always changing
My Experiences

• Set up 5 environments (Essex, Folsom x2, Grizzly, Havana)
  ○ Always run into some problem
  ○ Never the same problem twice

• Automated tools are your friend (packstack, puppet)

• Really convenient for kernel hacking

• Developers like it a lot
  ○ Can give them root

• Rapid Release
  – Things are always changing
  + Noticeable improvement with each release
My Experiences

- Set up 5 environments (Essex, Folsom x2, Grizzly, Havana)
  - Always run into some problem
  - Never the same problem twice
- Automated tools are your friend (packstack, puppet)
- Really convenient for kernel hacking
- Developers like it a lot
  - Can give them root
- Rapid Release
  - Things are always changing
  + Noticeable improvement with each release
- VMs are just as reliable as with libvirt, mgmt not so much
Hypervisor Performance Comparison
Experimental Setup

Systems:
- Dell R720 (2013) (Machine 1)
- Homebuilt (2011) (Machine 2)
- Ubuntu 12.04 LTS

Bioinformatics Applications:
- Paired-end short read alignment
- Burrows-Wheeler Aligner
- Novoalign
Memory Performance

STREAM Results

- PHY
- KVM
- XEN-pv
- LXC

MiB/s

Copy  Scale  Add  Triad

0  1000  2000  3000  4000  5000  6000  7000  8000 MiB/s

Copy  Scale  Add  Triad
Mean Runtime (normalized to physical)

BWA Runtime

PHY KVM XEN-pv LXC

0.95 1.00 1.05 1.10 1.15 1.20 1.25

1.19
1.09
1.00

Mean Runtime (normalized to physical)

BWA Runtime (Machine 2)

PHY KVM XEN-pv LXC

0.96 0.98 1.00 1.02 1.04

0.96 0.98 1.00 1.02 1.04

Mean Runtime (normalized to physical)

Novoalign Runtime

PHY KVM XEN-pv LXC

0.96 0.98 1.00 1.02 1.04

0.96 0.98 1.00 1.02 1.04

Mean Runtime (normalized to physical)
### BWA Runtime

- PHY: 0.95
- KVM: 1.00
- XEN-pv: 1.05
- LXC: 1.10

Mean Runtime (normalized to physical): 1.00

### Novoalign Runtime

- PHY: 0.96
- KVM: 0.98
- XEN-pv: 1.00
- LXC: 1.02

Mean Runtime (normalized to physical): 1.00
Mean Runtime (normalized to physical)

PHY | KVM | XEN-pv | LXC
---|---|---|---
BWA Runtime
1.00 | 1.19 | 1.09 | 1.00

BWA Runtime (Machine 2)
0.95 | 1.00 | 1.05 | 1.10 | 1.15 | 1.20 | 1.25
Mean Runtime (normalized to physical)
1.00 | 1.11 | 1.00 | 1.00

Novoalign Runtime
0.96 | 0.98 | 1.00 | 1.02 | 1.04
Mean Runtime (normalized to physical)
1.00 | 1.06 | 0.99 | 0.00

Zak Estrada
What’s in a Cloud? An Overview of Virtualization and Openstack
49/55
Multiple VMs

- Only on the Dell R720
- Still not utilizing this machine fully
• Gathered using `collectl`
• Only for a single run
Some Conclusions

• CPU pinning can be effective (but be careful)
Some Conclusions

- CPU pinning can be effective (but be careful)
- Performance for a given VMM is dependent on the application being executed
Some Conclusions

• CPU pinning can be effective (but be careful)
• Performance for a given VMM is dependent on the application being executed
• CPU-bound applications run at near-baremetal speeds in virtualized environments
Some Conclusions

• CPU pinning can be effective (but be careful)
• Performance for a given VMM is dependent on the application being executed
• CPU-bound applications run at near-baremetal speeds in virtualized environments
• VMM scheduling overhead is low when VMs are not overprovisioned
Some Conclusions

- CPU pinning can be effective (but be careful)
- Performance for a given VMM is dependent on the application being executed
- CPU-bound applications run at near-baremetal speeds in virtualized environments
- VMM scheduling overhead is low when VMs are not over provisioned
- Linux Containers have low performance overhead
For more info...

https://www.usenix.org/legacy/events/sec2000/robin.html
http://wiki.xen.org/wiki/Event_Channel_Internals
http://wiki.openstack.org
http://devstack.org
Slide theme courtesy Flip Tanedo, Cornell
Images are rights of their respective owners
Machine Statistics

Machine 1:
- dual-socket 8 core Intel Xeon E5-2660 2.2GHz CPUs (3.0GHz Turbo boost), 20MiB cache
- 128GiB of DDR3-1333MHz
- 8 SAS 10K RAID 10, 1024MiB cache

Machine 2:
- dual-socket 6 core Intel Xeon E5645 2.40GHz (2.67 GHz Turbo boost), 12MiB cache
- 32GiB of DDR3-1333Mhz
- 1 TiB SATA Disk