Comparing Open Source Private Cloud (IaaS) Platforms

Lance Albertson
OSU Open Source Lab
Associate Director of Operations

lance@osuosl.org / @ramereth
About me

- OSU Open Source Lab
- Server hosting for Open Source Projects
- Open Source development projects
- Gentoo Developer
- Sysadmin
- Jazz trumpeter
Disclaimer
What I'll Cover

● Compare 4 IaaS Platforms
● IaaS Components
● Discuss Strengths / Weaknesses
● Provide best uses for each platform
● NOT covering PaaS or SaaS platforms
Background Experience

- Used Xen+iSCSI for several years
- Researched an alternative tool
- Picked Ganeti+KVM 3 years ago
- Have had excellent experience for our use case
- Created web front-end for Ganeti
- Looking at augmenting services with OpenStack
Current State of Private IaaS

- Many options
- AWS API support
- Maturity of the projects
- Solving different problems
- Complexity of the platform
- Differences in backend architecture
What do you want in an IaaS?

- Ease of use
- Fault tolerance
- Low-cost of entry/maintenance
- Performance
- Ease of expansion
- API provisioning
- Compatibility with other platforms
- Agility / Fast provisioning
Major components of IaaS

- Storage
- VM Image management
- Self service / Web interface
- Networking
- Fault tolerance
- User management
- API / Hybrid Cloud Readiness
- Installation / Maintenance
Platforms I'm comparing

openstack

ganeti

CloudStack

Eucalyptus
OpenStack History

● Joint project with Rackspace & NASA
● Launched in June 2010
● Enable anyone to create and offer cloud computing services
● Many corporations joined
OpenStack Components

- Nova (compute)
- Swift (object storage)
- Glance (image service)
- Keystone (identity management)
- Horizon (gui interface)
Eucalyptus History

- Started as a research project at UC Santa Barbara
- Company founded in 2009 to commercialize the project
- Split into two editions:
  - Open-core
  - Open source
- June 2012 back to fully open source
Eucalyptus Components

- Cloud Controller (CLC)
  - Manages the virtualization resources and APIs
  - Provides web interface
- Walrus (S3 storage)
- Cluster Controller (CC)
  - Controls execution of VMs and their networking
- Storage Controller (SC)
  - Provides block-level storage to VMs (EBS)
- Node Controller (NC)
  - Controls VMs via hypervisors
CloudStack History

- Originally developed by Cloud.com
- Open Sourced in May 2010 (GPLv3)
- Citrix purchased Cloud.com in Aug 2011
- Donated to ASF in Feb 2012
CloudStack Components

- Management Server
- Hypervisor Nodes
- Storage Nodes
- Layers: Zone, Pod, Cluster, Host, Primary Storage, Secondary Storage
Ganeti History

- Started as internal Google
- Open sourced in August 2007
- Used primarily for back-office servers for Google
- Focus on hardware fault-tolerance
- Local block-level storage
- Cheap commodity hardware
Ganeti Components

- **Master daemon**
  - Controls overall cluster coordination
- **Node daemon**
  - Controls node functions (storage, VMs, etc)
- **Conf daemon**
  - Provide a fast way to query configuration
- **API daemon**
  - Provide a remote API
- **Htools**
  - Auto-allocation & rebalancing tools
Component Comparison
# Storage Comparison

<table>
<thead>
<tr>
<th>Type</th>
<th>OpenStack</th>
<th>Eucalyptus</th>
<th>CloudStack</th>
<th>Ganeti</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disk Images</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes [1]</td>
</tr>
</tbody>
</table>

1. Disk Image support has limitations
2. Via an elastic block storage service
3. iSCSI, OCFS2, CLVM (depends on hypervisor)
4. Primary storage method, also has sharedfs support
5. Uses rsync in the backend
6. Not added until version 3.0, uses DRBD
7. Parts are built-in, Storage is on your own
# VM Image Comparison

<table>
<thead>
<tr>
<th>Type</th>
<th>OpenStack</th>
<th>Eucalyptus</th>
<th>CloudStack</th>
<th>Ganeti</th>
</tr>
</thead>
<tbody>
<tr>
<td>Image Service</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>Amazon API</td>
<td>yes [3]</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
</tbody>
</table>

1. Ability for users to create and manage their own VM images
2. Third-party applications can offer this
3. Some support
## Self Service Comparison

<table>
<thead>
<tr>
<th>Type</th>
<th>OpenStack</th>
<th>Eucalyptus</th>
<th>CloudStack</th>
<th>Ganeti</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web Interface</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes [1]</td>
</tr>
<tr>
<td>Users &amp; Quotas</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes [1]</td>
</tr>
<tr>
<td>Console access</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes [1]</td>
</tr>
<tr>
<td>User management</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes [1]</td>
</tr>
</tbody>
</table>

1. Available via third-party application Ganeti Web Manager
## Networking Comparison

<table>
<thead>
<tr>
<th>Type</th>
<th>OpenStack</th>
<th>Eucalyptus</th>
<th>CloudStack</th>
<th>Ganeti</th>
</tr>
</thead>
<tbody>
<tr>
<td>Auto-allocation</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no [1]</td>
</tr>
<tr>
<td>Floating IPs</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>User defined</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>Layer 2</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
</tbody>
</table>

1. Proposal submitted but not yet implemented
## Other factors

<table>
<thead>
<tr>
<th></th>
<th>OpenStack</th>
<th>Eucalyptus</th>
<th>CloudStack</th>
<th>Ganeti</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Codebase</strong></td>
<td>Python</td>
<td>Java, C</td>
<td>Java</td>
<td>Python, Haskell, Shell</td>
</tr>
<tr>
<td><strong>Hypervisors</strong></td>
<td>Xen, KVM, UML, LXC, VMware</td>
<td>Xen, KVM, VMware</td>
<td>Xen, KVM, VMware, Citrix XenServer</td>
<td>Xen, KVM, LXC</td>
</tr>
<tr>
<td><strong>Installation Requirements</strong></td>
<td>Medium</td>
<td>Large</td>
<td>Medium/Large</td>
<td>Low</td>
</tr>
<tr>
<td><strong>Maintenance [1]</strong></td>
<td>Many components to maintain</td>
<td>Depends on your size</td>
<td>Medium</td>
<td>Easy</td>
</tr>
</tbody>
</table>

1. Base on my observation and opinion
Ease of Installation

- Included via distribution
- Amount of upfront configuration needed for a base install
- Ease of initialization of a cluster

<table>
<thead>
<tr>
<th>OpenStack</th>
<th>Eucalyptus</th>
<th>CloudStack</th>
<th>Ganeti</th>
</tr>
</thead>
<tbody>
<tr>
<td>Included in Ubuntu</td>
<td>Excellent Install Guide</td>
<td>Provide their own repos</td>
<td>Included in Debian/Ubuntu</td>
</tr>
<tr>
<td>Lots of configuration</td>
<td>Yum/Apt repos</td>
<td>Excellent install guide</td>
<td>Good Docs</td>
</tr>
<tr>
<td>required</td>
<td>Few commands for initialization</td>
<td>Minimal configuration needed</td>
<td>Simple initialization</td>
</tr>
<tr>
<td>Puppet Labs Module</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>OpenStack</td>
<td>Eucalyptus</td>
<td>CloudStack</td>
</tr>
<tr>
<td>---------------</td>
<td>-----------</td>
<td>-----------</td>
<td>------------</td>
</tr>
<tr>
<td><strong>Weakness</strong></td>
<td>Young codebase</td>
<td>Install requirements</td>
<td>Very GUI centric</td>
</tr>
<tr>
<td></td>
<td>Uncertain future</td>
<td>Configurable but not very customizable</td>
<td>Single java core</td>
</tr>
<tr>
<td></td>
<td>Initial configuration</td>
<td>Community Inclusion</td>
<td>AWS integration weak</td>
</tr>
<tr>
<td><strong>Strengths</strong></td>
<td>Single codebase</td>
<td>Excellent commercial support</td>
<td>Well-rounded GUI</td>
</tr>
<tr>
<td></td>
<td>Growing community</td>
<td>Fault-tolerance</td>
<td>Stack is fairly simple</td>
</tr>
<tr>
<td></td>
<td>Corporate support</td>
<td>Offers a hybrid-cloud solution with AWS</td>
<td>Customization of the storage backend</td>
</tr>
</tbody>
</table>
Which platform do you choose?

- Size of deployment
- Types of services to be hosted
- User-base
- Hardware/Budget limitations
- Complexity of the system
- Fault tolerance importance
- Compatibility with other clouds
# Summary of Comparisons

<table>
<thead>
<tr>
<th></th>
<th>OpenStack</th>
<th>Eucalyptus</th>
<th>CloudStack</th>
<th>Ganeti</th>
</tr>
</thead>
<tbody>
<tr>
<td>Philosophy</td>
<td>public &amp; private cloud,</td>
<td>hybrid private/public cloud</td>
<td>Private, highly customized cloud</td>
<td>Private, node failure tolerant, local storage</td>
</tr>
<tr>
<td></td>
<td>standardized API</td>
<td>compatibility</td>
<td>standardized API</td>
<td></td>
</tr>
<tr>
<td>Public Cloud</td>
<td>Some AWS</td>
<td>Excellent AWS</td>
<td>Some AWS</td>
<td>None</td>
</tr>
<tr>
<td>Compatibility</td>
<td>Large group of machines for</td>
<td>Large group of machines for</td>
<td>Medium group of machines</td>
<td>Smaller group of machines for highly trusted</td>
</tr>
<tr>
<td></td>
<td>lots of users</td>
<td>lots of semi-trusted users</td>
<td>for semi-trusted users</td>
<td>users with fault tolerance</td>
</tr>
<tr>
<td>Ideal Setting</td>
<td>Some built-in</td>
<td>Good with recent versions</td>
<td>Some built-in</td>
<td>Fully tolerant / Designed</td>
</tr>
<tr>
<td>Fault-tolerance</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Choosing Openstack

- Very young project
- Lots of corporate backing
- Codebase is simplified (python only)
- Excellent for large deployments
- Web interface is young, limited
- Only use the components you need
- Medium complexity
- Excellent APIs
Choosing Eucalyptus

- Fairly mature project
- Lots of features
- Codebase is complicated
- Complex installation requirements
- Great commercial support
- Excellent hybrid-cloud platform
- Re-focused effort back to Open Source
Choosing CloudStack

- No Distribution Support
- Lots of features
- Medium complexity to setup
- Fault-tolerance built into parts
- AWS compatibility is weak
- Monolithic component architecture
- Recent ownership shifts
- Used by several large hosting providers
Choosing Ganeti

- Fault tolerance built-in
- Ideal for smaller clusters
- Less complex, but less featureful
- No EC2 compatibility
- Better performance
- Local storage
- Only solves the compute problem
- Can be augmented with GlusterFS & other third-party applications
What about the others?

- OpenNebula - HPC community
- Nimbus - Scientific community
- oVirt - libvirt
No single winner or loser

- Solving different problems
- No perfect solution for everything
- Try each platform out first
- Map out what your end goal is
- Think about:
  - scalability
  - manageability
  - fault-tolerance
Questions?

Lance Albertson
lance@osuosl.org
@ramereth
http://lancealbertson.com