How to build an IaaS cloud: Xen, XenCloud, Open Stack

Roy Campbell

Adapted from Slides by Lars Kurth and Derek Daggit
A Brief History of Xen in the Cloud

Late 90s

**XenoServer Project**
(Cambridge Univ.)

The XenoServer project is building a public infrastructure for wide-area distributed computing. We envisage a world in which XenoServer execution platforms will be scattered across the globe and available for any member of the public to submit code for execution.

**Global Public Computing**

“This dissertation proposes a new distributed computing paradigm, termed global public computing, which allows any user to run any code anywhere. Such platforms price computing resources, and ultimately charge users for resources consumed.”

Evangelos Kotsovinos, PhD dissertation, 2004
A Brief History of Xen in the Cloud

Late 90s

XenoServer Project
(Cambridge Univ.)

Xen Repository
Published

Nov ‘02

Xen Presented
at SOSP

Oct ‘03

‘06

Amazon EC2
and SliceHost
launched

‘08

Rackspace
Cloud

‘09

XCP 1.x
Xen in Linux
Kronos
Cloud Mgmt

‘11

XCP
Announced
Xen.org

- Guardian of Xen Hypervisor and related OSS Projects
- Xen project Governance similar to Linux Kernel
- Projects
  - Xen Hypervisor (led by Citrix)
  - Xen Cloud Platform aka XCP (led by Citrix)
  - Xen ARM (led by Samsung)
  - PVOPS : Xen components and support in Linux Kernel (led by Oracle)
The Xen Community
Xen Contributions & Vendors

By Change Sets *

2011 Contributions by KLOC **

*) Does not count activity on XenARM (as not yet in an official repo)

**) Activity on Development branch (not yet in xen-unstable)

***) Figures up to end of Q3 2011
Community & Ecosystem Map

xen.org/community/projects
Xen Overview
Basic Xen Concepts

Control Domain aka Dom0
- Dom0 kernel with drivers
- Xen Management Toolstack
- Trusted Computing Base

Guest Domains
- Your apps
- E.g. your cloud management stack

Driver/Stub/Service Domain(s)
- A “driver, device model or control service in a box”
- De-privileged and isolated
- Lifetime: start, stop, kill
Para-Virtualized Domains

Linux PV guests have limitations:
- limited set of virtual hardware

Advantages
- Fast
- Works on any system (even without virt extensions)

Driver Domains
- Security
- Isolation
- Reliability and Robustness

*) Can be MiniOS
HVM & Stub Domains
(Hardware assisted Virtual Machines)

Disadvantages
- Slower than PV due to Emulation (mainly I/O devices)

Advantages
- Install the same way as native Linux

Stub Domains
- Security
- Isolation
- Reliability and Robustness
PV on HVM

- A mixture of PV and HVM
- Linux enables as many PV interfaces as possible
- This has advantages
  - install the same way as native
  - PC-like hardware
  - access to fast PV devices
  - exploit nested paging
  - Good performance trade-offs
- Drivers in Linux 3.x

<table>
<thead>
<tr>
<th></th>
<th>HVM</th>
<th>PV on HVM</th>
<th>PV</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Boot Sequence</strong></td>
<td>Emulated</td>
<td>Emulated</td>
<td>PV</td>
</tr>
<tr>
<td><strong>Memory</strong></td>
<td>HW</td>
<td>HW</td>
<td>PV</td>
</tr>
<tr>
<td><strong>Interrupts, Timers &amp; Spinlocks</strong></td>
<td>Emulated</td>
<td>PV*</td>
<td>PV</td>
</tr>
<tr>
<td><strong>Disk &amp; Network</strong></td>
<td>Emulated</td>
<td>PV</td>
<td>PV</td>
</tr>
<tr>
<td><strong>Privileged Operations</strong></td>
<td>HW</td>
<td>HW</td>
<td>PV</td>
</tr>
</tbody>
</table>

*) Emulated for Windows
Xen and the Linux Kernel

Xen was initially a University research project.

Invasive changes to the kernel to run Linux as a PV guest.

Even more changes to run Linux as dom0.
Xen and the Linux Kernel

Xen support in the Linux kernel not upstream

Great maintenance effort on distributions

Risk of distributions dropping Xen support

Xen harder to use
Current State

PVOPS Project

Xen Domain 0 in Linux 3.0+
(it is functional but not yet fully optimized)

On-going work to round out the feature set in Linux 3.2, 3.4 +
XCP Project
XCP

- Complete vertical stack for server virtualization
- Distributed as a closed appliance (ISO) with CentOS 5.5 Dom0, misc DomU’s, network & storage support and Xen API
- Open source distribution of Citrix XenServer
XCP Overview

• Open source version of Citrix XenServer

• Enterprise-ready server virtualization and cloud platform
  ▪ Extends Xen beyond one physical machine and other functionality
  ▪ Lots of other additional functionality compared to Xen

• Built-in support and templates for Windows and Linux guests

• Datacenter and cloud-ready management API
  ▪ XenAPI (XAPI) is fully open source
  ▪ CloudStack and OpenStack integration

• Open vSwitch support built-in
Project “Kronos”: XAPI on Linux

- Make the XAPI toolstack independent of CentOS 5.5
- Extend the delivery model
  - Deliver Xen, XAPI and everything in between (storage manager, network support, OCaml libs, etc.) via your favorite Linux distro
    - “apt-get install xcp-xapi” or “yum install xcp-xapi”

- Debian
- Next: Ubuntu 12.04 LTS
- Later: other major Linux distro (Fedora, CentOS, etc.)
  - Volunteers are welcome!
# Xen vs. XCP vs. XAPI on Linux

<table>
<thead>
<tr>
<th>Xen</th>
<th>XCP (up to 1.1)</th>
<th>XAPI on Linux</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hypervisor: latest</td>
<td>lagging</td>
<td>Linux distro</td>
</tr>
<tr>
<td><strong>Dom0 OS</strong>: CentOS, Debian, Fedora, NetBSD, OpenSuse, RHEL 5.x, Solaris 11, ...</td>
<td>CentOS 5.5</td>
<td>Debian, Ubuntu, ...</td>
</tr>
<tr>
<td>Dom 0: 32 and 64 bits</td>
<td>32 bits</td>
<td>32 and 64 bits</td>
</tr>
<tr>
<td>Linux 3 PVOPS Dom0: Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Toolstack: XM (deprecated), XL or Libvirt</td>
<td>XAPI + XE (lots of additional functionality to Xen)</td>
<td>Same as XCP</td>
</tr>
<tr>
<td>Storage, Network, Drivers: build and get yourself</td>
<td>Integrated with Open vSwitch, multiple storage types &amp; drivers</td>
<td>Get them yourself</td>
</tr>
<tr>
<td>Configurations: Everything</td>
<td>constrained by XAPI</td>
<td>Same as XCP</td>
</tr>
<tr>
<td>Usage Model: Do it yourself</td>
<td>Shrink wrapped and tested</td>
<td>Do it yourself</td>
</tr>
<tr>
<td>Distribution: Source or via Linux Unix distributions</td>
<td>ISO</td>
<td>Via host Linux distribution</td>
</tr>
</tbody>
</table>
XCP 1.5, 1.6

- **Architectural Improvements**: Xen 4.1, GPT, smaller Dom0
- **GPU pass through**: for VMs serving high end graphics
- **Performance and Scalability**:
  - 1 TB mem/host
  - 16 VCPUs/VM, 128 GB/VM
- **Networking**: Open vSwitch (default), Active-Backup NIC Bonding
- **Virtual Appliance**: multi-VM and boot sequenced, OVF support
- More guest OS templates
XAPI Overview
XAPI: What is it?

• XAPI is the backbone of XCP
  – Provides the glue between all components
  – Is the backend for all management applications

• Call it XAPI or XenAPI

• It's a XML-RPC style API, served via HTTPS
  – Provided by a service on every XCP dom0 host
  – Designed to be highly programmable
  – API bindings for many languages: .NET, Java, C, Powershell, Python

• XAPI is Extensible via plugins
  – E.g. used by OpenStack
XAPI from 30000 Feet

- user
- session
- host
- PIF
- network
- VIF
- pool
- task
- event
- VM
- SM
- SR
- VDI
- VBD
- BBD_metrics
- Host_metrics
- PIF_metrics
- VM_metrics
- VM_guest_metrics
- consol
- crash_dump
- VM_guest_metrics
SR XAPI Classes

- **Storage Repository (SR):** is a storage target, containing virtual disks. The SR object provides XE with a set of mechanisms to use to manage disks on that particular type of storage, and allows provisioning operations, such as creating a new virtual disk, to be mapped onto a wide variety of storage types.

- **Physical Block Device (PBD):** A PBD represents the interface between a physical host and an attached SR. PBDs are connector objects that allow a given SR to be mapped to a host.

- **Virtual Disk Image (VDI):** A VDI is an on-disk representation of a virtual disk provided to a guest VM. It is the fundamental unit of virtualized storage in XenServer.

- **Virtual Block Device (VBD):** A VBD is a connector object (similar to the PBD described above), that allows mappings between VDIs and VMs.
XAPI Functionality Overview

- VM lifecycle: live snapshots, checkpoint, migration
- Resource pools: live migration, auto configuration, disaster recovery
- Flexible storage and networking
- Event tracking: progress, notification
- Upgrade and patching capabilities
- Real-time performance monitoring and alerting

Open vSwitch

- Software switch, similar to:
  - VMware vNetwork Distributed Switch
  - Cisco Nexus 1000V

- Distribution agnostic. Plugs right into Linux kernel.

- Reuses existing Linux kernel networking subsystems.

- Backwards-compatible with traditional userspace tools.

Why use Open vSwitch with Cloud?

- Automated control: OpenFlow
- Multi-tenancy
- Monitoring and QoS
XAPI Management Options

• XAPI frontend command line tool: XE (tab-completable)
• Desktop GUIs
  o Citrix XenCenter (Windows-only)
  o OpenXenManager (open source cross-platform XenCenter clone)
• Web interfaces
  o Xen VNC Proxy (XVP)
    ▪ lightweight VM console only
    ▪ user access control to VMs (multi-tenancy)
  o XenWebManager (web-based clone of OpenXenManager)
• XCP Ecosystem:
  o xen.org/community/vendors/XCPProjectsPage.html
  o xen.org/community/vendors/XCPPProductsPage.html
OpenXenManager
XCP and Cloud Orchestration Stacks
Cloud Computing : OpenStack

“The OpenStack project has been created with the audacious goal of being the ubiquitous software choice for building cloud infrastructures.”

— Ken Pepple, Deploying OpenStack, O’Reilly

“Cloud computing is a computing model, where resources such as computing power, storage, network and software are abstracted and provided as services on the Internet in a remotely accessible fashion. Billing models for these services are generally similar to the ones adopted for public utilities. On-demand availability, ease of provisioning, dynamic and virtually infinite scalability are some of the key attributes of cloud computing.”

— docs.openstack.org
“OpenStack is a collection of open source software projects that enterprises/service providers can use to setup and run their cloud compute and storage infrastructure.”

— docs.openstack.org

The OpenStack Consortium has grown rapidly in the past year:

- NASA
- Rackspace
- Citrix
- Dell
- AMD
- Intel
- Cisco
- HP
- Over 140 others

OpenStack services can be made available via Amazon’s S3 and EC2 APIs. Applications written for Amazon Web Services can work with OpenStack.
OpenStack’s Core Components

• **Compute (“Nova”) → EC2**
  Orchestrates large networks of Virtual Machines. Responsible for VM instance lifecycle, network management, and user access control.

• **Object Storage (“Swift”) → S3**
  Provides scalable, redundant, long-term storage for things like VM images, data archives, and multimedia.

• **Image Service (“Glance”)**
  Manages VM disk images. Can be a stand-alone service. Supports private/public permissions, and can handle a variety of disk image formats.
OpenStack Nova

Nova was contributed by NASA from the Nebula platform.

Nova allows users to create, destroy, and manage virtual machines using user-supplied images.

Corresponds to Amazon’s EC2.

Users can use OpenStack API or Amazon’s EC2 API.

Uses Python and Web Server Gateway Interface (WSGI).
OpenStack Nova: Architecture

Diagram showing the Nova architecture with components such as nova-api, nova-network, nova-scheduler, nova-volume, and nova-compute.
OpenStack Nova: nova-api

A daemon that is the workhorse of Nova.

- Handles API requests.
- Manages most orchestration.
- Enforces some policies.

If it can, it will handle the request on its own with help from the database.

Otherwise, it will delegate to the other nova daemons using the message queue as well as the database.
OpenStack Nova: nova-compute

Worker that does the actual work of starting and stopping virtual machine instances.

Takes its orders from the message queue, and executes the appropriate VM API calls to accomplish the task.

Commonly uses “libvirt” (RedHat), but can use Xen, vSphere (VMware), or Windows Management Interface.
OpenStack Nova: nova-network

Worker that does the actual work of configuring the network.

Network is specified as one of three types:
- Flat
- FlatDHCP
- VLAN
OpenStack Nova: nova-scheduler

Worker that simply takes a VM instance request from the queue and chooses a host.

- Simple:
  Least Load
- Chance:
  Random available host *(default)*
- Zone:
  Random within availability zone

Large deployments will want to implement more comprehensive and sophisticated algorithms.
OpenStack Nova: nova-volume

Worker that manages volumes for persistent storage.

Compatible with several vehicles, including AoE, iSCSI, and Sheepdog.
OpenStack Nova: Message Queue

The queue is used for message passing among daemons.

Currently implemented by RabbitMQ (VMware, OSS, Erlang).

Could swap out RabbitMQ with any system that supports Advanced Message Queuing Protocol (AMQP).

Queue Types:
- Fanout
- Host
- Topics
OpenStack Nova: Database

Stores the state of running cloud infrastructure.

Supports the usual suspects: sqlite3, MySQL, PostgreSQL, etc.
OpenStack Swift

Swift was contributed by Rackspace, and powers their Cloud Files product.

Swift is not a distributed file system, but a distributed object store.

Access files via the API (OpenStack/S3), not via NFS or the like.
OpenStack Swift: Architecture
OpenStack Swift: Presentation

• swift-proxy
  • Handles incoming requests and delegates to the appropriate process.
  • Uses OpenStack API out-of-the-box or Amazon S3 with middleware.
  • Capable of authentication and authorization.
OpenStack Swift: Resource

- **swift-account**
  operates an sqlite3 database for accounts

- **swift-container**
  operates an sqlite3 database for object “containers”

- **swift-object**
  maps objects themselves onto the node’s storage

All three manage replication and self-consistency.
OpenStack Glance

Virtual Machine Disk Image Service

Can be used stand-alone, but integrates well as service between Nova and Swift.

Supports a range of virtual disk image formats and container formats.

(VMDK, ISO, AMI, etc.)
OpenStack Glance: Architecture
OpenStack Glance: glance-api

Similar to nova-api, delegates requests as appropriate.

Communicates with glance-registry and with the Image Store (Swift, Amazon S3, a file system, read-only HTTP, etc.)

Basic REST API, JSON
OpenStack Glance: glance-registry

Interacts with glance-api and manages simple database holding image metadata.

Ships with a reference implementation using sqlite3.

(If scaling, replace this reference implementation.)
Identity & Dashboard Components

OpenStack will integrate two additional components in the next release—planned for Q2 of 2012—named “Essex.”

• Identity (“Keystone”)
  Provides integrated authentication among OpenStack components, and can leverage external authentication systems.

• Dashboard (“Horizon”)
  Provides a browser-based “control panel” application for administrators and users.
Getting Involved

• Community
  • http://openstack.com/community/

• Documentation
  • http://docs.openstack.org/