What is OpenVZ?

OpenVZ is a project that combines the following container virtualization technologies for Linux:

- **Virtuozzo kernel,** a Linux kernel with patches that implements OpenVZ kernel functionality.
- **Management utilities,** such as vzctl, for managing container life cycle.
- **Checkpoint/Restore In Userspace,** or CRIU (pronounced *kree-oo*, IPA: */krɪʊ/*, Russian: *крыу*), is a software tool for Linux that enables you to freeze a running application (or a part of it) and checkpoint it to a hard drive as a collection of files. You can then use the files to restore and run the application from the point it was frozen at. The distinctive feature of the CRIU project is that it is mainly implemented in userspace. Docker and LXC use CRIU for migrating containers between servers.
- **Ploop** is a disk loopback block device, not unlike loop but with many features like dynamic resize, snapshots, backups etc. The main idea is to put container filesystem in a file.
- **P.Haul** is the project on top of CRIU that implements the live migration usage scenario.
- **LibCT** is a container management library that provides a convenient API for front-end programs for managing the entire container life cycle.

Use cases

Server Consolidation

- Uniform management.
- Easy to upgrade from Virtuozzo OpenVZ edition to the commercial Virtuozzo.
- Scalable.
- Fast migration.

Development and Testing

- Different distros can coexist.
- A container can be created in a minute.
- A server can have hundreds of containers.
- Cloning, snapshots, rollbacks are available.
- A container is a sandbox: one can work and play without fear.

Security

- Give each app its own isolated container.
- Security hole in an app will not affect others.
- Dynamic resource management controls runaway processes.

Hosting

- Users are isolated.
- A container is like a real server, just cheaper.
- Much easier to administer.

Education

- Every student can have root access.
- Different distributions are supported.
- Low hardware requirements.

Containers can run various Linux distributions:

OpenVZ
Virtuozzo Containers

Container-based virtualization for Linux.
Fast, lightweight, secure.
Choose three.

Supported by
Odin

Checkpoint and Restore In Userspace

www.openvz.org www.criu.org

OpenVZ Decennial

a short history of the OpenVZ project

1999 – SWsoft prepared first concept of product with container virtualization

2000 – Limited public beta testing of new product

2002 – SWsoft initially released a product for Linux named Virtuozzo

2005 – SWsoft created the OpenVZ Project to release the core of Virtuozzo under GNU GPL

2006 – New ports: SPARC and PPC; OpenVZ is available in Debian Linux; OpenVZ adds live migration capability

2008 – Published templates for Ubuntu 7.10, created OpenVZ port to ARM

2009 – Parallels company is in Top 10 Linux kernel contributors with their patches for Linux containers

2011 – Prepared first implementation of our new project CRIU

2012 – First release of CRIU project, vzctl for upstream Linux kernel is available

2013 – OpenVZ maintenance partnership

2014 – Parallels decided to merge OpenVZ and Virtuozzo into single common open source codebase

2015 – Source code of RHEL7-based kernel was published and kernel development process become open

Full story is here: http://openvz.org/History

Frequently Asked Questions

What is a container (Virtual Environment, Virtual Private Server)?

A container (CT) is an isolated entity which works exactly like a standalone server. Containers can be rebooted independently and have root access, users/groups, IP addresses, memory, processes, files, applications, system libraries, and configuration files.

What is a virtual machine?

A virtual machine (VM) is an emulation of a particular computer system. Virtual machines operate based on the computer architecture and functions of a real or hypothetical computer, and their implementations may involve specialized hardware, software, or a combination of both.

What are the highlights of OpenVZ technology?

OpenVZ is a highly scalable virtualization technology for Linux with near-zero overhead, strong isolation and rapid customer provisioning that is ready for production use out of the box. Deployment of OpenVZ improves efficiency, flexibility, and quality of service in the enterprise environment.

How is OpenVZ different from other technologies?

Virtual machines boot separate kernels on emulated hardware instances. OpenVZ runs all containers under a single Linux kernel. OpenVZ offers a much higher density, enabling to host thousands of containers on a single physical server, but can only run Linux in those containers. Virtual machine solutions usually top out at a few dozen instances, but can run different operating systems in each.

What is the relationship between OpenVZ and LXC?

OpenVZ develops a new container technology that then goes upstream into the vanilla Linux kernel. OpenVZ has an about 5 year head start on LXC, but is actively feeding the technology upstream into vanilla containers. Several internal details currently differ (OpenVZ adds new system calls, vanilla uses the cgroups filesystem, new clone flags, and other mechanisms).

What applications can run inside an OpenVZ container?

Applications and services do not have to be aware of OpenVZ, and most are installed without any modifications: Java, Oracle, DB/2, Weblogic, Websphere, and many other big applications run just fine inside OpenVZ containers. However, direct access to hardware is not available by default; if required it must be provided by the system administrator.

How scalable is OpenVZ?

OpenVZ scales as well as Linux does: we've tested 64 CPUs with 128 GB of RAM. It scales down to embedded devices like smart phones or plug computers. A single container can dynamically scale to take from a tiny fraction to all available resources and that can be adjusted without restart.

How does OpenVZ improve efficiency?

OpenVZ improves utilization of existing hardware by increasing average load while still providing the ability to handle peak loads. When buying new servers, using a few powerful boxes instead of many small ones allows better reliability, better peak performance and typically longer lifespan.

How does OpenVZ improve flexibility of services?

Each container is hardware-independent and can be moved to another OpenVZ-based system over network in seconds. This eases hardware maintenance (move out all containers and do whatever you need with the box) and improves availability (keep a synchronized copy of your container elsewhere and start it up if primary service fails). When your old box can no longer cope with peak load, you can live migrate your containers to a new one.

What is the performance overhead?

Near zero. There is no emulation layer, only security isolation and resource accounting. All checking is done in the kernel without context switching.

Where do I get (or put) more answers?

OpenVZ wiki is your friend. See http://wiki.openvz.org/