OpenVZ ist eine Container basierende Virtualisierungstechnologie, welche auf Linux aufbaut. Es werden isolierte, sichere Container, sogenannte VEs or VPS, auf einer physischen Hardware erstellt. Dies hat den Vorteil, dass die in den Containern laufenden Dienste nicht in Konflikt mit Diensten anderer Container treten. Jeder Container verhält sich wie ein nativer Stand-Alone-Server und kann separat neu gestartet werden. Mit OpenVZ ist es möglich, mehrere hundert Container auf einem physischen Server zu betreiben.

Container können als leichtgewichtige virtuelle Maschinen angesehen werden. OpenVZ setzt voraus, dass das Host- und das Gast-Betriebssystem Linux sind (Distributionen können je Container unterschiedlich sein). Der Virtualisierungs-Overhead beträgt bei dieser Art der Virtualisierung nur ca. 1-3%, verglichen mit einem Stand-Alone-Server.

OpenVZ ist freie Software, jedermann darf es unter den Vorgaben der GNU Gerneral Public License verwenden, weitergeben und verändern.

OpenVZ basiert auf einem modifizierten Linux Kernel und User-Space Programmen. Der Kernel erhält die Eigenschaft, Container abzubilden, stellt die Virtualisierung zur Verfügung, kümmert sich um die Sicherheit als auch um das Ressourcen Management und verfügt über die Möglichkeit von Checkpoints sowie Live Migration von Containern (von Hardware zu Hardware).

Virtualisierung und Isolierung

Jeder Container hat seine eigenen:
Dateien (System Bibliotheken, Programme, `/proc` und `/sys` Filesysteme);
Prozesse (Seperater Prozessbaum, mit einem init-Prozess mit der PID 1);
Benutzer und Gruppen (einschließlich root);
Netzwerk (eigene virtualisierte Netzwerk Devices, IP Adressen, Routing Tabellenund Netfilter (iptables) Regeln);
IPC Objekte (shared memory, semaphores, messages);
… und vieles mehr.

Ressourcen Management

Der Kernel gibt dem Container Ressourcen und limitiert diese, damit kein Single-Container Systemressourcen missbrauchen kann. Die drei wichtigsten Sub-Systeme sind:

User Beancounters, sind ein Satz von Ressource Zählern, Limits und garantierten Ressourcen. Diese Zähler umfassen ca. 20 Werte, welche individuell eingestellt werden können, um das Verhalten eines Containers zu beeinflussen. Diese Werte verhindern, dass ein einzelner Container alle Systemressourcen einer Hardware vereinnahmt. Ressourcen wie zum Beispiel Kernel Speicher, Netzwerk Buffer, physikalischer und virtueller Speicher usw. werden überwacht. Das alles kann per Container als ulimit bezeichnet werden.

Fairer CPU Scheduler. Der CPU Scheduler verteilt die CPU Zeit zwischen den Containern auf der Basis von vorgegebenen Prioritäten, somit kann kein Container die gesamte Rechenleistung beanspruchen. Das alles kann verwendet werden, um harte CPU Zeit Limits und Garantien zu vergeben.

I/O scheduler. Der I/O Scheduler stellt die Möglichkeit zur Verfügung, die I/O Bandbreite zwischen den Containern auf der Basis von vorgegebenen Prioritäten zu verteilen. Es werden detaillierte Statistiken der I/O Aktivität des Containers aufgezeichnet.

Zweistufiges Disk-Quota. Die erste Stufe ist ein per Container Disk Quota, die zweite Stufe stellt ein Standard Unix per User und Gruppen Festplatten Quota innerhalb eines Containers dar.

Live Migration und Checkpointing

Der OpenVZ Kernel ist in der Lage, einen kompletten Status eines Containers einzufrieren und in eine Dump-Datei zu speichern (bekannt als Checkpointing). Dieser Status kann dann wieder hergestellt werden und der Container kann wieder weiterlaufen, ähnlich einem Suspend-to-Disk. Der Unterschied ist, dass bei OpenVZ nur ein einzelner Container gestoppt wird, nicht das ganze System.

Dieser eingefrorene Container kann auf einem anderen physischem System wiederhergestellt werden. Das ermöglicht die sogenannte Live Migration. Es gibt keinen Neustart eines Containers, was aus Anwendersicht bedeutet, dass es zu einer Verzögerung aber nicht zu einem Ausfall des Systems kommt.

Benutzer Level Programme

vzctl ist ein high-level Kommandozeilenprogramm, um einen Container zu kontrollieren. Es wird verwendet, um einen Container zu erstellen, zu starten, zu stoppen und wieder zu löschen. Außerdem können mit ihm verschiedene Container Parameter verändert und gesetzt werden, wie zum Beispiel IP Adresse, User Beancounter Limits, CPU shares, Festplatten Quotas usw.

Übliche vzctl Befehle:

```
# vzctl create 101 --ostemplate centos-5
# vzctl set 101 --name foo --save
# vzctl set foo --ipadd 10.1.2.2 --save
# vzctl set foo --diskspace 2G --save
# vzctl set foo --numproc 200 --save
# vzctl start foo
# vzctl exec foo ps ax
# vzctl enter foo
# vzctl stop foo
# vzctl destroy foo
```

vzlist ist ein Programm, um Container und deren Parameter anzuzeigen und es hat eine Menge Optionsmöglichkeiten. Es kann gut verwendet werden, um mittels Scripts verschiedene Abläufe zu automatisieren.

vzsplit wird verwendet um eine Standard Container Konfiguration zu erzeugen, welche es ermöglicht, N Container auf einer physikalischen Hardware zu betreiben.

vzmigrate ist ein Script, welches die Migration von Containern durchführt (live und offline).

vzmemcheck zeigt den gesamten physischen Ressourcenverbrauch an und wird verwendet um die Hardwarekapazität zu planen sowie einen Überverbrauch von Ressourcen z.B: Speicher zu prüfen.

Templates

Templates sind vorgefertigte Container Images von verschiedenen Linux Distributionen, welche sich für das schnelle Erstellen von Containern eignen. Man kann vorgefertigte Templates verwenden, diese bei Bedarf verändern oder neue erstellen, welche den eigenen Vorstellungen entsprechen.

Es ist einfach, neue Templates für OpenVZ zu erstellen – man muss ein minimales lauffähiges Betriebssystem erstellen. Das kann mit verschiedenen Programmen geschehen, z.B.: yum oder debootstrap, je nach verwendeter Distribution.

Die folgenden vorgefertigten Templates stehen zur Verfügung:

- **CentOS** 4 und 5
- **Debian** 3.1, 4 und 5
- **Fedora** 7, 8, 9, 10
- **openSUSE** 10.3 und 11.1
- **Ubuntu** 7.10, 8.04, 8.10 und 9.04

Folgende Templates werden von der Community zur Verfügung gestellt:

- **ALTLinux**
- **ArchLinux**
- **Gentoo**
- **Slackware**
- usw.

Häufig gestellte Fragen (und Antworten)!

Was ist ein Container (Virtuelle Umgebung, Virtueller Privater Server, VPS, VE)?

Ein Container (CT) ist eine isolierte Einheit, welche sich exakt wie ein Stand Alone Server verhält. Container können unabhängig von der Hardware neu gestartet werden. Sie haben eigenen root Zugriff, eigene Benutzer und Gruppen, IP Adressen, Speicher, Dateien, Applikationen, System Bibliotheken und Konfigurationsdateien. OpenVZ erlaubt es, mehrere hundert Container auf einem einzigen physischem Server zu betreiben.

Was sind die Highlights der OpenVZ Technologie?

Kurz, OpenVZ ist eine hoch skalierbare Technologie für Linux mit beinahe keinem Overhead, starke Isolierung und schnelle Bereitstellung, welche bereit ist für den Einsatz in produktiven Umgebungen. Der Einsatz von OpenVZ erhöht die Effizienz, Flexibilität und die Qualität ihres Services im Enterprise Umfeld.

Was unterscheidet OpenVZ von anderen Technologien? Was sind die Fallstricke?

Anders als virtuelle Maschinen oder Paravirtualisierung verwendet OpenVZ einen einzigen Betriebssystemkern – den Linux Kernel. Das ermöglicht eine sehr effiziente Virtualisierung, mit dem Nachteil, dass keine anderen Betriebssysteme betrieben werden können (z.B. Windows).

Welche Anwendungen können in einem OpenVZ Container laufen?

Die meisten Applikationen können in einem Container installiert werden (ohne Anpassung). Oracle, DB/2, Weblogic, Websphere und viele andere große Applikationen funktionieren bestens innerhalb eines Containers. Programme und Dienste merken nichts von OpenVZ, jedoch, der direkte Hardwarezugriff ist standardmäßig nicht verfügbar.

Wie gut skaliert OpenVZ?

OpenVZ skaliert sehr gut – wir haben es mit Servern, welche 16 CPU's und 64GB Ram hatten, getestet. Ein Container kann ein paar Systemressourcen verwenden oder eine gesamte Hardware verbrauchen – dynamisch. Ein Neustart eines Containers ist dafür nicht notwendig.

Wie steigert OpenVZ die Effizienz eines Services?

Bei derzeit verfügbarer Hardware erlaubt OpenVZ die Ausnutzung der Hardware von 3-5% auf 30-50%, wobei genügend Ressourcen für Spitzenlasten verbleiben. Bei der Anschaffung von neuen Servern müssen Sie nur wenige starke Server kaufen, anstatt viele schwache – dies erhöht die Spitzenperformance und ermöglicht normalerweise einen längeren Lebenszyklus.

Wie erhöht OpenVZ die Flexibilität von Diensten?

Jeder Container ist eine autonome Einheit, welche von einem OpenVZ basierenden System zu einem anderen, über das Netzwerk, verschoben werden kann. Das erleichtert die Systemwartungen. Verschieben sie die Container einfach auf einen anderen Server! OpenVZ erhöht die Verfügbarkeit (speichern Sie eine synchronisierte Kopie auf einem alternativen Speicherplatz) und starten Sie diese sobald der primäre Dienst ausfällt. Sollte eine alte Hardware die Leistungsanforderung nicht mehr bewältigen können, ist es on-the-Fly möglich den Container zu verschieben.

Wie hoch ist der Performance Verlust?

Fast Null. Es gibt keine Emulationsschicht, nur die Sicherheitsisolierung und die Überprüfung erfolgt zusätzlich durch den Kernel, jedoch ohne Context switch.

Wann steht die höchste Leistung zur Verfügung?

Spitzenleistung kann erreicht werden, wenn nur ein Container aktive Tasks hat. In diesem Fall kann ein Container 100% aller verfügbaren Ressourcen verwenden. Alle CPU's, den gesamten Memory, alle Festplatten und die komplette Netzwerkbandbreite. OpenVZ schränkt einen Container nicht auf eine einzelne CPU ein!

Gibt es Verwaltungsprogramme für OpenVZ?

Schauen Sie unter http://wiki.openvz.org/Control_panels

Wo bekomme ich mehr Antworten oder kann Fragen stellen?

OpenVZ wiki ist der beste Startpunkt - http://wiki.openvz.org/

Anwendunsfälle

Server Konsolidierung

- Einheitliches Management
- Einfaches Upgrade
- Mehr Skalierbarkeit
- Schnelle Migration
- Stromsparen und Platzsparen

Entwicklung und Testen

- Verschiedene Distributionen können parallel laufen
- Einen neuen Container zu erstellen dauert keine Minute
- Mehrere hundert Container sind möglich
- Klonen, Schnappschüsse und Rollbacks
- Ein Container stellt eine Sandbox dar: Arbeiten/Testen, keine Angst

Sicherheit

- Jede Applikation kann einen eigenen isolierten Container haben
- Eine Sicherheitslücke in einer Applikation gefährdet nicht andere
- Plus: Dynamisches Ressourcen Management

Hosting

- Isolierte Benutzer
- Ein Container ist wie ein physischer Server, nur billiger
- Wesentlich leichtere Administration

Bildungsmöglichkeiten

- Jeder Student kann root Zugriff haben
- Unterschiedliche Distributionen
- Wenig Hardwareverbrauch

Andere Container Technologien

- Parallels Virtuozzo Containers (eine kommerzielle Version von OpenVZ)
- Linux-VServer
- FreeBSD jails
- Solaris 10 Containers
- AIX 6 Workstation Partitions (WPARs)