- OpenStack是一个云管理操作系统，用来控制数据中心中的计算、存储、网络资源池
- 管理员通过一个Web界面为用户提供所需的资源

每天5分钟

Open Source Cloud Computing Platform

玩转OpenStack

CloudMan 编著

清華大學出版社

每天5分钟玩转 OpenStack

CloudMan 编著

清华大学出版社
北京

内 容 简 介

本书是一本 OpenStack 的教程和参考。读者在学习的过程中，可以跟着教程进行操作，在实践中掌握 OpenStack 的核心技能。在之后的工作中，则可以将本教程作为参考书，按需查找相关知识点。

本书共分为两大部分。第一部分介绍虚拟化和云计算基础知识，重点讲解 KVM 的理论和实践。第二部分首先介绍 OpenStack 架构，演示如何搭建 OpenStack 环境，然后逐一详细讲解 OpenStack 各个核心模块，包括 Keystone、 Glance、Nova、Cinder 和 Neutron。

本书适合 OpenStack 初学者、云计算技术人员、云计算研究人员等使用，也适合高校和培训学校相关专业的师生教学参考。

图书在版编目（CIP）数据

每天 5 分钟玩转 OpenStack / CloudMan 编著. — 北京：清华大学出版社，2017
ISBN 978-7-302-45531-8

I. ①每… II. ①C… III. ①计算机网络 IV.①TP393

中国版本图书馆 CIP 数据核字（2016）第 277403 号

责任编辑：夏毓彦
封面设计：王　翔
责任校对：闫秀华
责任印制：沈　露

出版发行：清华大学出版社
网　　址：http://www.tup.com.cn，http://www.wqbook.com
地　　址：北京清华大学学研大厦 A 座　　**邮　　编**：100084
社 总 机：010-62770175　　**邮　　购**：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质 量 反 馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 装 者：北京嘉实印刷有限公司
经　　销：全国新华书店
开　　本：190mm×260mm　　**印　　张**：27　　**字　　数**：691 千字
版　　次：2017 年 1 月第 1 版　　**印　　次**：2017 年 1 月第1 次印刷
印　　数：1～3500
定　　价：89.00 元

产品编号：070923-01

前 言

写在最前面

这是一个 OpenStack 教程，有下面两个特点：

- 系统讲解 OpenStack。从架构到各个组件；从整体到细节逐一讨论。
- 重实践并兼顾理论。主要从实际操作的角度带着大家学习 OpenStack。

为啥要写这个

简单回答是：因为 OpenStack 学习难度大，但如果掌握了，价值会很大。

先做一个自我介绍吧。

本人网名 CloudMan，在 IT 这个行当已经摸爬滚打了十多年，2005 年之前是搞上层应用开发的，那时候 Java 比较火，所以 J2EE 相关的技术搞得比较多。后来入职一家大型 IT 公司，公司的产品从中间件到操作系统，从服务器到存储，从虚拟化到云计算都有涉及。

本人所在的部门是专门做 IT 基础设施实施服务的，项目涉及服务器、存储、网络、虚拟化、云各个方面，而且这个部门的重要任务是为公司在 IT 市场最新和最热门的领域开疆扩土。比如前几年的虚拟化，这两年的云计算和大数据。

可以说部门的这个定位非常符合我的技术偏好。我对新技术长期保持着浓厚的兴趣和学习热情，所以在这个部门一待就是十几年，而且一直搞技术，虽然现在的头衔是架构师，平时还是一直坚持实际动手操作，否则会没有安全感。

好，现在回到 OpenStack 这个主题。

本人是在 2013 年开始接触 OpenStack，虽然具备比较扎实的技术功底，在经过一段时间的学习后，还是感觉 OpenStack 这个东西上手不太容易，个人认为有以下几个原因：

1. OpenStack 涉及的知识领域极广

可以说涵盖了 IT 基础设施的所有范围，计算、存储、网络、虚拟化、高可用、安全、灾备无所不包，即便是像我这种每天都在这个领域工作的人也感觉压力颇大。

2. OpenStack 是一个平台，不是一个具体的实施方案

OpenStack 的各个组件都采用 Driver 的架构，支持各种具体的实现技术。比如 OpenStack 的存储服务 Cinder 只定义了上层抽象 API，具体的实现交给下面的各种 Driver，比如基于 LVM

的 iSCSI Driver，EMC、IBM 等商业存储产品的 Driver，或者是开源的分布式存储软件，比如 Ceph、GlusterFS 的 Driver。

正是因为这种架构上的灵活性，使得初学者在学习 OpenStack 的时候不会像学习其他具体软件产品那样容易上手。

3. OpenStack 本身是一个分布式系统

大多数搞 IT 的对分布式计算都不会太熟悉，直接冲进来会被 OpenStack 繁多的组件以及它们之间的交互方式搞得云里雾里。

虽然 OpenStack 学习曲线比较陡峭，掌握起来难度较大，但 OpenStack 目前已经是 IaaS 云的事实标准，而且前途一片光明，对于我们搞 IT 的如果能啃下这个骨头，必定能大大提升自身的竞争力。

写给谁看

这套教程的目标读者包括：

1. OpenStack 初学者

我学习 OpenStack 也是经历了一个艰辛曲折的过程，其主要原因在于没有找到一个系统讲解 OpenStack 的教程，大部分资料都比较分散，对于初学者无法有机地串起来。也正是因为这个原因，让我萌发了编写这样一套教程的想法，能够让初学者少走弯路，系统地学习、掌握和实践 OpenStack。

2. OpenStack 实施工程师

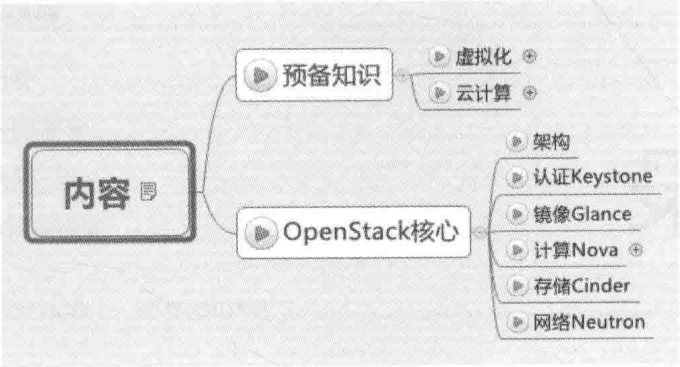
之前说了，我在公司的职位是架构师，但骨子里我更把自己定位成一位能到一线攻城拔寨的实施工程师。所以这个教程也是针对 OpenStack 的实施人员，让他们能够通过学习真正掌握部署 OpenStack 的知识、技能以及故障排查技巧。

3. 我自己

写这个教程同时也是对自己这几年学习和实践 OpenStack 的一个总结。我觉得：对于知识，只有把它写出来并能够让其他人理解才能真正说明自己掌握了这项知识。

包含哪些内容

本书两大块内容，如下图所示。



1. 预备知识

因为面向初学者，首先会有虚拟化和云计算的“预备知识”，会介绍 KVM、IaaS 等技术。

2. OpenStack 核心

这是主要内容，包含 OpenStack 的架构和各个核心组件。将会通过大量的案例、操作步骤、截图、日志来帮助大家理解 OpenStack 各组件是如何工作的。其目标是让各位可以根据客户的需求进行配置和调整。

怎样的编写方式

在当下这个共享经济时代，我觉得应该用互联网的方式来分享知识和心得。这个教程会通过我的微信公众号（cloudman6）每周一、周三、周五定期发布。

用公众号我觉得有两个好处：

（1）可以随时随地查看和浏览已推送的内容。

（2）可以通过公众号跟我互动，提出问题和建议 。

为啥叫《每天 5 分钟玩转 OpenStack》

为了降低学习的难度并且考虑到移动端碎片化阅读的特点，每次推送的内容大家只需要花 5 分钟就能看完（注意，这里说的是看完，有时候要完全理解可能需要更多时间哈），每次的内容只包含 1~3 个知识点，这也是我把教程命名为《每天 5 分钟玩转 OpenStack》的原因。虽然是碎片化推送，但整个教程是系统、连贯和完整的，只是化整为零了。

好了，今天这 5 分钟算是开了个头，下面我们正式开始玩转 OpenStack。

编 者

2016 年 10 月

目 录

第一篇

预备知识

OpenStack 是云操作系统，要学习 OpenStack，首先需要掌握一些虚拟化和云计算的相关知识。

第1章
◂虚拟化▸

虚拟化是云计算的基础。简单地说，虚拟化使得在一台物理的服务器上可以跑多台虚拟机，虚拟机共享物理机的 CPU、内存、IO 硬件资源，但逻辑上虚拟机之间是相互隔离的。

物理机我们一般称为宿主机（Host），宿主机上面的虚拟机称为客户机（Guest）。

那么 Host 是如何将自己的硬件资源虚拟化，并提供给 Guest 使用的呢？

这个主要是通过一个叫做 Hypervisor 的程序实现的。

根据 Hypervisor 的实现方式和所处的位置，虚拟化又分为两种：1 型虚拟化和 2 型虚拟化。

1.1 1 型虚拟化

Hypervisor 直接安装在物理机上，多个虚拟机在 Hypervisor 上运行。Hypervisor 实现方式一般是一个特殊定制的 Linux 系统。Xen 和 VMWare 的 ESXi 都属于这个类型，如图 1-1 所示。

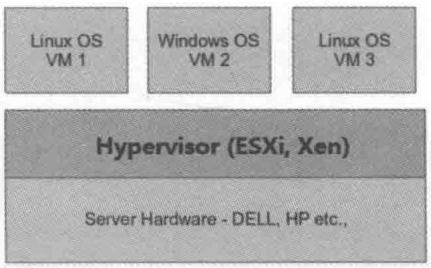


图 1-1

1.2 2 型虚拟化

物理机上首先安装常规的操作系统，比如 Redhat、Ubuntu 和 Windows。Hypervisor 作为 OS 上的一个程序模块运行，并对虚拟机进行管理。KVM、VirtualBox 和 VMWare Workstation 都属于这个类型，如图 1-2 所示。

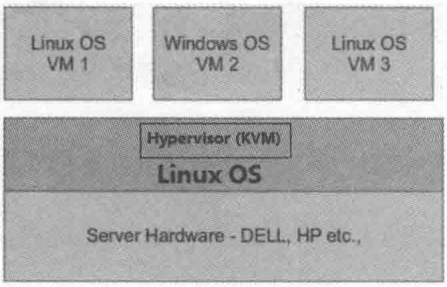


图 1-2

理论上讲：

1. 型虚拟化一般对硬件虚拟化功能进行了特别优化，性能上比 2 型要高；

2. 型虚拟化因为基于普通的操作系统，会比较灵活，比如支持虚拟机嵌套。嵌套意味着可以在 KVM 虚拟机中再运行 KVM。

1.3 KVM

下面重点介绍 KVM 这种 2 型虚拟化技术。

1.3.1 基本概念

在 x86 平台上最热门、运用最广泛的虚拟化方案莫过于 KVM 了。OpenStack 对 KVM 支持得也最好，我们的教程也理所当然选择 KVM 作为实验环境的 Hypervisor。

KVM 全称是 Kernel-Based Virtual Machine。也就是说 KVM 是基于 Linux 内核实现的。

KVM 有一个内核模块叫 kvm.ko，只用于管理虚拟 CPU 和内存。

那 IO 的虚拟化，比如存储和网络设备由谁实现呢？

这个就交给 Linux 内核和 Qemu 来实现。

说白了，作为一个 Hypervisor，KVM 本身只关注虚拟机调度和内存管理这两个方面。IO 外设的任务交给 Linux 内核和 Qemu。

Libvirt

大家在网上看 KVM 相关文章的时候肯定经常会看到 Libvirt 这个东西。

Libvirt 是啥？

简单地说就是 KVM 的管理工具。

其实，Libvirt 除了能管理 KVM 这种 Hypervisor，还能管理 Xen，VirtualBox 等。

OpenStack 底层也使用 Libvirt，所以很有必要学习一下。

Libvirt 包含 3 个东西：后台 daemon 程序 libvirtd、API 库和命令行工具 virsh。

- libvirtd 是服务程序，接收和处理 API 请求；
- API 库使得其他人可以开发基于 Libvirt 的高级工具，比如 virt-manager，这是个图形化的 KVM 管理工具，后面我们也会介绍；
- virsh 是我们经常要用的 KVM 命令行工具，后面会有使用的示例。

作为 KVM 和 OpenStack 的实施人员，virsh 和 virt-manager 是一定要会用的。

今天 5 分钟差不多了，下一节我们来玩 KVM。

1.3.2 KVM 实操

1. 准备 KVM 实验环境

上一节说了，KVM 是 2 型虚拟化，是运行在操作系统之上的，所以先要装一个 Linux。Ubuntu、Redhat、CentOS 都可以，这里以 Ubuntu14.04 为例。

基本的 Ubuntu 操作系统装好之后，安装 KVM 需要的包：

```
    $ sudo apt-get install qemu-kvm qemu-system libvirt-bin virt-manager
bridge-utils vlan
```

通过这些安装包顺便复习一下上一节介绍的 KVM 的相关知识。

- qemu-kvm 和 qemu-system 是 KVM 和 QEMU 的核心包，提供 CPU、内存和 IO 虚拟化功能。
- libvirt-bin 就是 libvirt，用于管理 KVM 等 Hypervisor。
- virt-manager 是 KVM 图形化管理工具。
- bridge-utils 和 vlan，主要是网络虚拟化需要，KVM 网络虚拟化的实现是基于 linux-bridge 和 VLAN，后面我们会讨论。

Ubuntu 默认不安装图形界面，手工安装一下：

```
    sudo apt-get install xinit sudo apt-get install gdm sudo apt-get install
kubuntu-desktop
```

apt 默认会到官网上去下载安装包，速度很慢，我们可以使用国内的镜像站点。

配置/etc/apt/sources.list：

```
    deb http://mirrors.163.com/ubuntu/ trusty main restricted universe
multiverse
    deb http://mirrors.163.com/ubuntu/ trusty-security main restricted
universe multiverse
    deb http://mirrors.163.com/ubuntu/ trusty-updates main restricted
universe multiverse
    deb http://mirrors.163.com/ubuntu/ trusty-proposed main restricted
universe multiverse
```

```
   deb http://mirrors.163.com/ubuntu/ trusty-backports main restricted
universe multiverse
   deb-src http://mirrors.163.com/ubuntu/ trusty main restricted universe
multiverse
   deb-src http://mirrors.163.com/ubuntu/ trusty-security main restricted
universe multiverse
   deb-src http://mirrors.163.com/ubuntu/ trusty-updates main restricted
universe multiverse
   deb-src http://mirrors.163.com/ubuntu/ trusty-proposed main restricted
universe multiverse
   deb-src http://mirrors.163.com/ubuntu/ trusty-backports main restricted
universe multiverse
```

然后执行下面命令更新安装包 index：

```
   # apt update
```

Redhat 和 CentOS 安装相对简单，安装过程中选择虚拟化和图形组件就可以了。

小窍门：Ubuntu 默认是不允许 root 通过 ssh 直接登录的，可以修改 /etc/ssh/sshd_config，设置：

```
   PermitRootLogin yes
```

然后重启 ssh 服务即可：

```
   # service ssh restart ssh
   stop/waiting ssh start/running, process 27639
```

在虚拟机上做实验

作为 2 型虚拟化的 KVM，支持虚拟化嵌套，这使得我们可以在虚拟机中实验 KVM。

比如我在 VMWare Workstation 中安装了一个 Ubuntu14.04 的虚拟机，为了让 KVM 能创建。嵌套的虚机，要把 CPU 的虚拟化功能打开。如图 1-3 所示，在 VMWare 中设置以下 CPU 的模式。

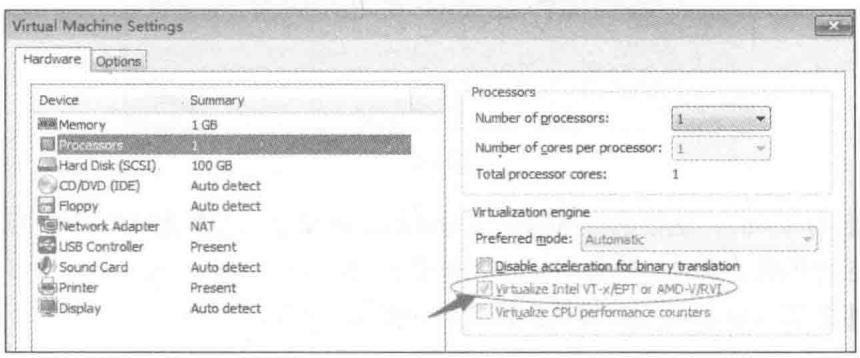


图 1-3

Ubuntu 启动后，用以下命令确认 CPU 支持虚拟化：

```
# egrep -o '(vmx|svm)' /proc/cpuinfo
# vmx
```

确认 Libvirtd 服务已经启动：

```
# service libvirt-bin status
libvirt-bin start/running, process 1478
```

2. 启动第一个 KVM 虚机

本节演示如何使用 virt-manager 启动 KVM 虚机。

首先通过命令 virt-manager 启动图形界面，如图 1-4 所示。

```
# virt-manager
```

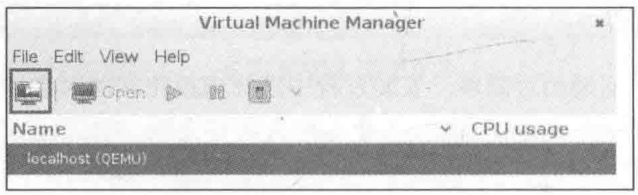


图 1-4

单击图 1-4 中框选的图标创建虚机，如图 1-5 所示。

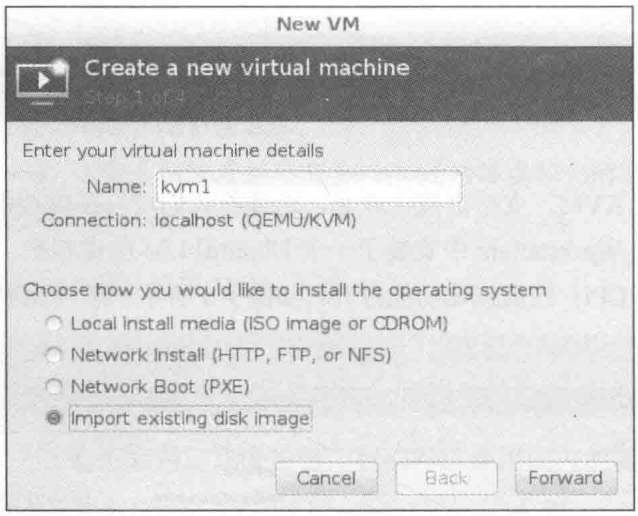


图 1-5

给虚机命名为 kvm1，这里选择从哪里启动虚机。如果是安装新的 OS，可以选择第一项。如果已经有安装好的镜像文件，选最后一项（如图 1-5 所示）。

接下来需要告诉 virt-manager 镜像的位置，如图 1-6 所示。