



# 基于 OpenStack 构建私有云计算平台 \*

李小宁, 李 磊, 金连文, 黎德生

(华南理工大学电子与信息学院 广州 510640)

## 摘要

云计算是近几年来 IT 领域非常热门的研究方向,而私有云凭借其安全性、可控性以及有效利用 IT 设备资源等特性,在云计算领域占据重要位置。本文详细介绍了 OpenStack 开源云计算平台的架构以及 OpenStack 的 5 个功能模块的关系,并重点介绍了一种基于 OpenStack 构建私有云的方案。此外,本文还分享了私有云平台搭建过程中遇到的问题和解决方法,并分析了此私有云平台的安全性以及弹性计算资源分配能力。

关键词 私有云; OpenStack; 弹性资源分配

## 1 引言

自从 2006 年 Google 首席执行官埃里克·施密特在搜索引擎大会上首次提出“云计算”(cloud computing)的概念,云计算就成为近几年来 IT 领域一个非常热门的研究方向。它使用户能够通过网络便利、按需地从一个共享的、可配置的资源池中获取计算、存储、网络等资源<sup>[1]</sup>。而在云计算的 3 种部署模式(公有云、私有云、混合云)中,公有云虽然以其低廉的价格、灵活的弹性策略吸引大量的用户,但是对于很多企业来说,和业务相关的数据是其生命线。由于公有云存储数据的地方并不是企业本地,因此在使用指定防火墙之外且更接近公共区域处的应用的安全问题引起了大家的关注<sup>[2]</sup>。

因此,构建一个私有云平台,使之主要为企业内部提供云服务,不对公众开放,在企业的防火墙内工作,并且企业 IT 人员能对其数据、安全性和服务质量进行有效的控制<sup>[3]</sup>,

成为倍受大企业青睐的选择。此外,私有云能够提供更高效、个性化的弹性计算资源分配能力,能够更有效地利用企业内部现有的 IT 设备资源,打造更符合企业需求的云平台。

鉴于上述提到的公有云的数据安全问题,本文的出发点在于:

- 如何构建一个适合本地环境的私有云;
- 针对实验室或开发环境构建一个云计算平台,使实验室可以针对数据安全、计算资源的按需获取等进行评估和探索。

在实际试验中,本文基于 OpenStack 开源工具软件包,实现了在科研实验室环境中搭建私有云计算平台,使得资源能够得到弹性分配,且确保数据安全。

## 2 系统框架

### 2.1 OpenStack 简介

OpenStack 是由 Rackspace 和美国国家航空航天局(NASA)共同开发的云计算平台,帮助服务商和企业内部实现类似于 Amazon EC2 和 S3 的云基础架构服务

\* 国家自然科学基金资助项目(No.61075021)



(infrastructure as a service, IaaS)<sup>[4]</sup>。自 2010 年 10 月开始, OpenStack 至今已经发布了 Austin、Bexar、Cactus、Diablo 以及 Essex 这 5 个版本。由于 OpenStack 采用 Apache2.0 许可证发布源代码,因此在短短两年时间内就吸引了 IBM、Cisco、HP 等 175 个企业及组织的加入。

OpenStack 采用模块化设计,它的 3 个主要模块 Nova (计算服务)、Swift (存储服务) 和 Glance (镜像服务),既可以组合在一起联合作,提供完整的云基础架构服务;又可以独立工作,分别提供虚拟化、云存储和镜像服务。此外,模块化的设计能把老、旧、第三方的技术集成进来,从而满足业务需要,也使得开发者能够更容易地对 OpenStack 进行二次开发。

## 2.2 私有云平台框架

基于 OpenStack 的开源框架介绍,本文所搭建的私有云计算平台由计算服务 (Nova)、存储服务 (Swift)、镜像服务 (Glance)、身份认证服务 (Keystone) 和 Web 界面服务 (Horizon) 共 5 个部分组成。其中 Nova 是云平台的控制器,它提供一个工具部署云,包括运行虚拟化实例、管理网络以及控制用户和其他项目对云的访问<sup>[5]</sup>。Swift 是一个大容量、可扩展的、内置冗余和容错机制的对象存储系统。对象存储支持多种应用,比如复制和存档数据、图像或视频服务、存储次级静态数据、开发数据存储整合的新应用、存储容量难以估计的数据、为 Web 应用创建基于云的弹性存储<sup>[6]</sup>等。Glance 是一个虚拟机镜像的存储、查询和检索系统,服务包括的 Restful API 允许用户通过 HTTP 请求查询 VM 镜像元数据以及检索实际的镜像。Keystone 为运行 OpenStack Compute 上的 OpenStack 云提供了认证和管理用户、账号和角色信息服务,并为 OpenStack Object Storage 提供授权服务。Horizon 是提供给用户来使用云平台的 Web 前端界面,用户可以使用它进行实例、密钥、弹性块存储、镜像、云存储以及角色权限等的管理。OpenStack 的 5 个服务的关系如图 1 所示。

## 3 构建过程

### 3.1 部署规划

本文搭建的私有云计算平台由客户端、控制器以及计算节点 3 个角色组成,其中客户端可以由实验室局域网段内任何一台物理机来构建;控制器由 Nova、Glance、Swift、Keystone 和 Horizon 组成,它们可以部署在一台物理机上,也可以分别部署在不同的物理机上,为了提高云平台的性

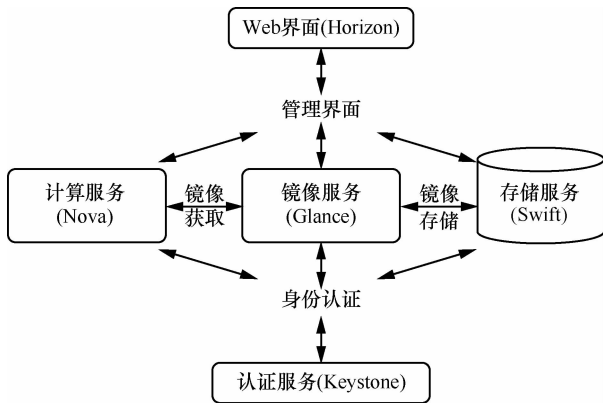


图 1 Openstack 服务关系

能,减轻大量用户对云控制器的压力,本文采用的是第二种部署方法;计算节点由支持 VT 技术的 10 台物理机组成,主要用于运行虚拟化实例,提供计算服务。

由于本文搭建的私有云主要应用于科研和中小型企业环境,云平台的各个组成部分均位于同一局域网内,这样,局域网内任何一台用户端的物理机均可以直接访问或使用云计算资源。此外,为了使得云平台更安全,Nova 提供了一种让虚拟化实例和外网隔离的网络方案,就是为虚拟化实例组建一个与外网隔离的私有网络,虚拟化实例通过一个内网交换机来通信。若实例需要和外网通信,那就再分配一个浮动的外网 IP 地址给实例。私有云的部署规划如图 2 所示。

### 3.2 部署过程

目前 OpenStack 可以采用 StackOps 光盘安装、DevStack 脚本在 Ubuntu 上安装、源代码安装等多种方式进行构建,支持 Ubuntu、Debian、Fedora 等操作系统。考虑到云平台的稳定性、后续管理等问题,本文采用 OpenStack 源代码安装的方法进行试验,使用的系统为 12.04 版 Ubuntu,OpenStack 版本为 Essex,部署过程共分为 5 个部分,分别为 Keystone 的部署、Swift 的部署、Glance 的部署、Nova 的部署和 Horizon 的部署。在部署的过程中,首先都需要从 OpenStack 的官方代码网站 <https://github.com/openstack> 获取源代码,并把它们以及依赖项安装到物理机上。

#### 3.2.1 Keystone 的部署

Keystone 是 OpenStack 框架中,负责身份验证、服务规则和服务令牌功能的部件。在部署 OpenStack 过程中,首先要安装 Keystone 并在 Keystone 中注册 Nova、Swift 和 Glance 服务,使得各项服务可以被授权和使用。Keystone 的部署流程如图 3 所示。

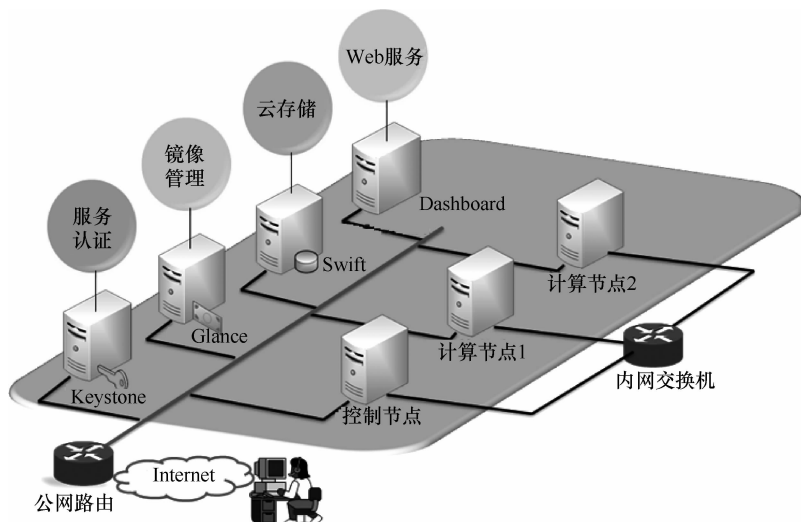


图2 私有云的部署

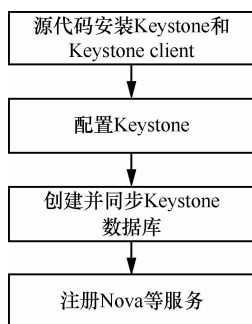


图3 Keystone的部署流程

本文采用源码方式安装,即采用 git clone<sup>[6]</sup>方式获取官方的 Keystone 和 Keystone client 源码,创建 Keystone 的配置文件路径和文件以及数据库配置,其中配置文件里面比较关键的项是 connection 和 admin\_token 参数设置,connection 指向将要使用的数据库,而 admin\_token 则是默认的管理员令牌,需要用它来访问 Keystone 的服务。完成 Keystone 的配置文件后,需要为 Keystone 数据库进行设置,Keystone 服务已经具备了数据库配置指令,即“sudo keystone-manage db\_sync”。

在完成基本的配置后,由于 Keystone、Nova 等本身就是一个服务,因此需要在 Keystone 服务内注册其服务。Keystone 的用户拓扑为租户(tenant)、用户(user)以及角色(role)的,它们分别表示了用户组、用户以及用户具备的权限等。实际上,Keystone 并没有规定租户内有多少个角色和用户,但是基本的角色有管理员和一般用户两种。在注册各项服务前,需要在 Keystone 内注册一个基本的租户、用户以及一个角色,以完成各项服务的注册。在 Keystone 内的服务注册包括服务(service)以及服务端点(endpoint)

的注册。其中服务是一个类的说明,服务端点是一个指向服务的访问链接,包括 IP 地址、端口号等。利用这些端点可以实现 Restful 方式访问 OpenStack 服务,通过 URL 调用 OpenStack API。

### 3.2.2 Swift 的部署

Swift 是 Rackspace 开发的一个分布式云存储项目,可以实现云端文件的多节点分布式存储管理,目前已经成为 OpenStack 的一个核心部件。

本文采用源码方式安装,即采用 git clone 方式获取官方的 Swift 源码。Swift 的配置流程如图 4 所示,配置的关键步骤包括以下几个。

(1)配置代理服务器(proxy server),代理服务器负责把 Swift 的其他部分连接起来。对于每一个请求,代理服务器将在环(ring)中查找用户(account)、容器(container)或者对象名(object),相应地将请求路由到对应的服务器上。对代理服务器的配置主要有代理服务器的绑定地址和端口以及 Keystone 对应的地址和端口。

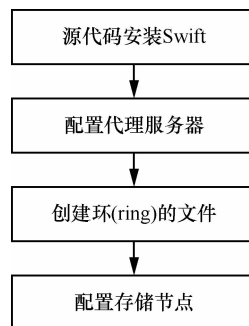


图4 Swift 部署流程



(2)配置环的信息,环负责管理存储在磁盘上的一个实体的名字和这个实体所在物理位置之间的映射关系<sup>[7]</sup>。因此要创建3个环的文件,分别对应 account、container 和 object。

(3)配置存储节点,主要步骤为格式化并挂载磁盘,分别配置 rsyncd.conf、account-server、container-server 和 object-server。Rsyncd 的作用是维持存储对象的一致性并进行数据更新操作。因此,rsyncd.conf 配置了存储节点所有 account、container 和 object 对应的信息。而 account-server、container-server 和 object-server 则分别配置了它们对应于环的端口。

(4)启动 rsyncd 以及 Swift 服务,即可使用云存储。

### 3.2.3 Glance 的部署

Glance 是 OpenStack 用于管理虚拟机镜像的工程,实现包括镜像的上传、验证、删除等管理。

本文采用源码方式安装,即采用 git clone 方式获取官方的 Glance 源码,Glance 的部署流程如图 5 所示。其配置的主要步骤包括以下几个。

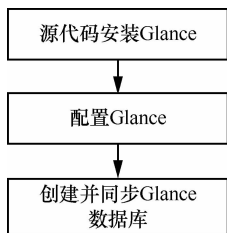


图 5 Glance 的部署流程

#### (1)Glance 存储的配置

Glance 的镜像文件可以用 Swift 来存储,也可以在本地存储。例如要用 Swift 来存储镜像文件,可把 glance-api.conf 文件里面的 default\_store=file 和 swift\_store\_create\_container\_on\_put=False 改为 default\_store=swift 和 swift\_store\_create\_container\_on\_put=True。同时,在文件 glance-api.conf、glance-api-paste.ini 和 glance-registry-paste.ini 中配置 Keystone 的相应信息。

#### (2)数据库配置

在 glance-registry.conf 配置文件中,配置 sql\_connection 参数指向 Glance 的数据库,同时为 Glance 创建一个数据库来保存镜像的 metadata,并且同步数据库。

#### (3)启动 Glance 服务

启动 Glance 服务,并通过 glance 指令来验证该服务是否可用。

### 3.2.4 Nova 的部署

Nova 为 OpenStack IaaS 服务项目工程,可提供包括虚拟机实例、网络、云硬盘(volume)等的管理,是 OpenStack 3个核心工程中最复杂的一个,目前扩展包括 Quantum 和 Melange 等孵化项目。

本文采用源码方式安装,即采用 git clone 方式获取官方的 Nova 源码,包括 nova-compute、nova-network、nova-scheduler、nova-api 和 nova-volume 等子部件的源代码。图 6 为 Nova 的部署流程,在源码安装完成后,需要做以下几个工作:

- 增加一个网络桥接设备,使得虚拟机可以通过网桥进行通信;
- 安装 tgt、iscsi 等第三方工具,使得云硬盘可在各个节点相互挂载;
- 安装 KVM 虚拟机工具,Nova 是通过第三方的虚拟机工具启动实例;
- 确保控制器与各计算节点的时间同步,这是因为计算节点的 Nova 服务会定时往控制节点的数据库写入自己的时间,然后与控制节点的本地时间做对比,假如时间差异大于某个阈值,则认定计算节点的 Nova 服务不正常。因此,为了保持控制节点与计算节点的时间一致性,需要将控制节点作为 NTP(时间同步协议)的服务器,然后修改计算节点的 NTP 配置文件,使之以控制节点为时间同步服务器。

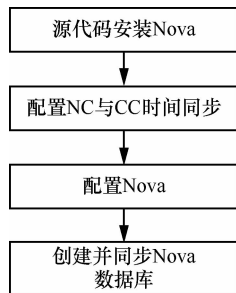


图 6 Nova 的部署流程

Nova 的基本配置主要通过统一的配置文件/etc/nova.conf 完成,其中核心的配置包括网络模式 network\_manager,本文选择 FlatDHCPManager 参数;调度策略 scheduler\_driver 参数,本文选择 MultiScheduler 和 glance\_api\_servers 参数。

完成上述配置后,创建并同步 Nova 的数据库,最后启动 Nova 服务,在控制端一般启动 nova-compute、nova-network、

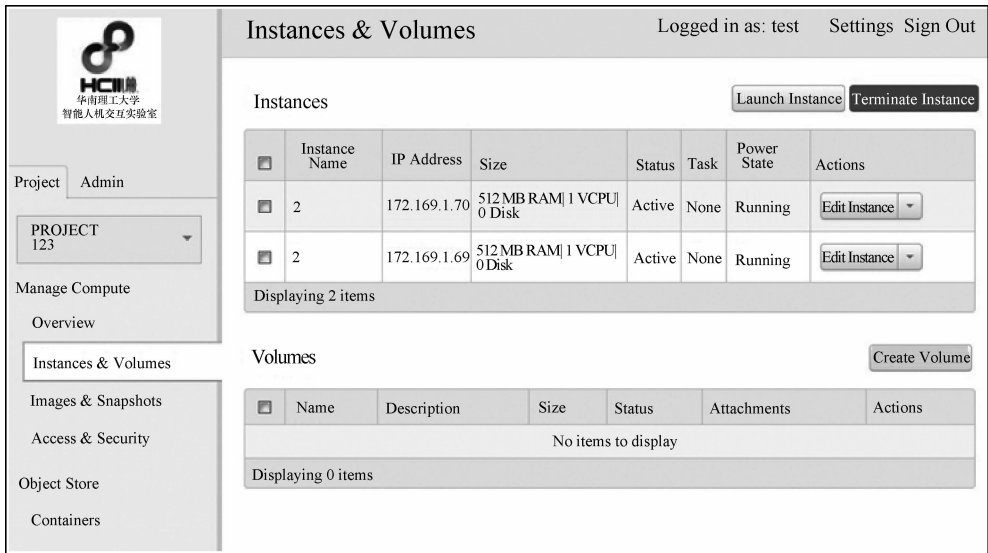


图 7 使用云平台的情况

nova-scheduler,nova-api 和 nova-volume 等服务,在计算节点仅需启动 nova-compute 和 nova-network 即可,计算节点会依据数据库的连接地址自动完成服务的注册。

### 3.2.5 Horizon 的部署

Horizon 为 OpenStack 的 Web 服务工程,采用 Django 结构开发。

本文采用源码方式安装,即采用 git clone 方式获取官方的 Horizon 源码,由于其自身具备 Web 服务,无需通过 Apache 启动。Horizon 的配置文件为源码下的 local\_settings.py,主要修改包括设置参数OPENSTACK\_HOST指向 Keystone 的 IP 地址。图 7 展示了客户端通过 Horizon 使用云平台的情况。

### 3.3 定制开发环境镜像

当 OpenStack 具备了控制节点、计算节点以及镜像服务后,管理员就可以为云平台定制所需的镜像,使得云平台可以向用户提供虚拟化实例。定制镜像流程如图 8 所示。

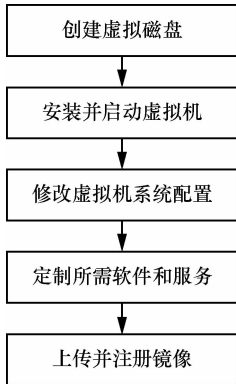


图 8 定制镜像流程

在定制过程中,创建虚拟磁盘及安装虚拟机的操作与 Linux 下运行 VM 一致,均采用 KVM 虚拟机实现<sup>[8]</sup>,可直接以正常方式启动 KVM 完成镜像的制作。

在制作 Windows 镜像时,需要安装 virtio 驱动,同时为了避免读写硬盘 I/O 速率较低,可在启动 KVM 时加一个参数 cache=none。

在制作 Linux 镜像时,由于 70-persistent-net.rules 会自动添加其他的网络接口,需要删除这个文件来避免自动添加除了 eth0 以外的接口。

在制作镜像的过程中,也可以为镜像定制软件和服务,即根据实验室的开发需要,在虚拟机上安装相应的开发环境或开启相关服务<sup>[9]</sup>。这样制作的镜像被用户实例化后,用户就可以直接使用完善的软件平台服务,避免了逐一安装软件的繁琐工作,提高了工作效率。例如制作 Linux 镜像时,可以安装并开启 VNC、远程桌面协议(RDP)等服务,这样用户就可以通过远程可视化登录使用 Linux 实例。

## 4 私有云平台搭建过程中遇到的问题和解决方法

根据构建私有云计算平台的经验,本文在此给出一些部署云平台过程中常见的问题以及解决方法。

(1) 多节点部署完成后,通过 nova-manage service list 命令发现,计算节点的各个服务不稳定

解决办法:这是计算节点和控制节点的时间不同步造成的,通过 NTP 同步计算节点与控制节点的时间即可解决问题。



(2)云平台正在运行中,但是突然间某个节点的 nova-compute 服务不正常

解决办法:导致这个问题的原因可能是某个节点的 nova-compute 服务停掉或者在某个节点上启动了两个 libvirtd 程序。因此,需要查看 nova-compute 服务是否正在运行,假如 nova-compute 服务正在运行并且有两个 libvirtd 程序启动,则重启 libvirtd 程序即可解决问题。

(3)虚拟化实例能够启动,但是无法获取私有 IP 地址

解决办法:由于 OpenStack 使用 dnsmasq 负责 dhcp 的工作,因此,假如计算节点的 nova-network 服务正常,则很可能是 dnsmasq 服务不正常,重启虚拟化实例所在的计算节点的 dnsmasq 服务即可。

(4)虚拟实例启动正常,但是无法连接

解决办法:这个问题很可能是没有打开相关端口造成的,出于安全性保障,OpenStack 在虚拟实例内部网络设置了防火墙,因此,假如要使用某些服务就要打开相关端口,如要用 ssh 连接就要打开 TCP 的 22 端口,要用 ping 服务就要打开 ICMP 的 -1 端口,要用远程桌面协议(RDP)连接就要打开 TCP 的 3389 端口。

(5)由于操作失误导致 Nova 数据库崩溃或者想推倒数据库重来

解决办法:假如是数据库里面某个 table 不正常,那么丢弃掉那个 table,然后同步数据库,就可以解决问题。假如整个 Nova 数据库已经混乱崩溃了,那么也不必重装数据库甚至重装 OpenStack,只需要丢弃 Nova 数据库,然后重建一个 Nova 数据库,给予 Nova 用户操作 Nova 数据库权限,同步数据库,再重建私有网段以及浮动网段,打开相应端口,即可使 Nova 服务恢复正常。

## 5 OpenStack 云平台部署实验及分析

本文搭建的私有云平台提供给用户的 Web 管理界面 Horizon 简单易用,功能丰富,登录界面如图 9 所示,初始的用户名和密码与在 Keystone 中注册服务时使用的一样。在 Horizon 中,管理员的权限包括创建和管理用户、项目组;修改用户组的实例可用配额;修改可用镜像类型;查看应用平台的使用情况等。用户的权限包括使用云存储;对虚拟实例进行启动、中断、挂起、关闭、创建快照、分配和回收浮动 IP 地址、分配和回收云硬盘等操作。以下将从启动实例、挂载云硬盘、使用云存储等方面进行介绍。

私有云平台的镜像有 Ubuntu11.10 桌面版、Windows

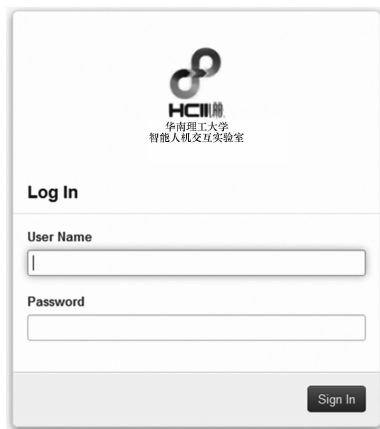


图 9 OpenStack 登录界面

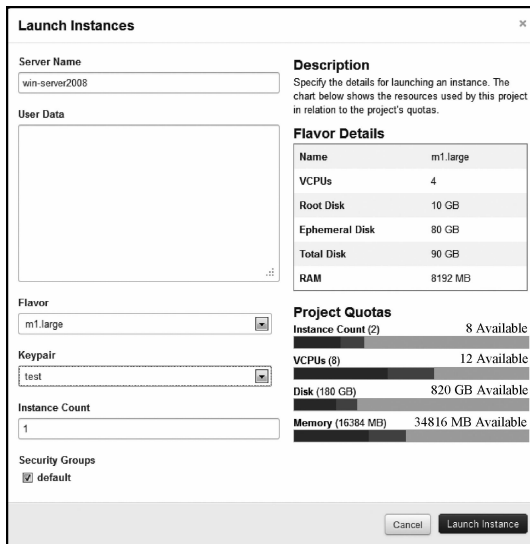


图 10 启动实例

2008 等,可以同时启动 Windows 和 Linux 虚拟实例。实例类型有单核、双核以及 4 核 CPU 等,还可按需要进行添加修改。启动实例的过程为:选择镜像、修改实例名字、选择实例类型、选择密钥以及选择创建实例的个数。启动实例过程如图 10 所示,实例启动可在几分钟内完成。访问实例有多种方式,可以通过 noVNC 从网页访问,可以通过远程桌面协议(RDP)连接访问,Linux 实例还可通过 puty、secureCRT 等软件访问。

云硬盘是 OpenStack 云平台提供给用户的可持久弹性存储。云硬盘创建时可根据用户需要配置大小,云硬盘的状态为 available(可用)时,即可挂载到实例。在 Windows 实例中使用云硬盘的步骤为:通过磁盘管理工具把挂载的云硬盘的状态改为 online(在线),初始化云硬盘为分区表格式(GPT),格式化并使用云硬盘,如图 11 所示。在 Linux 实例中使用云硬盘的步骤为:通过 fdisk 工具在云硬盘中

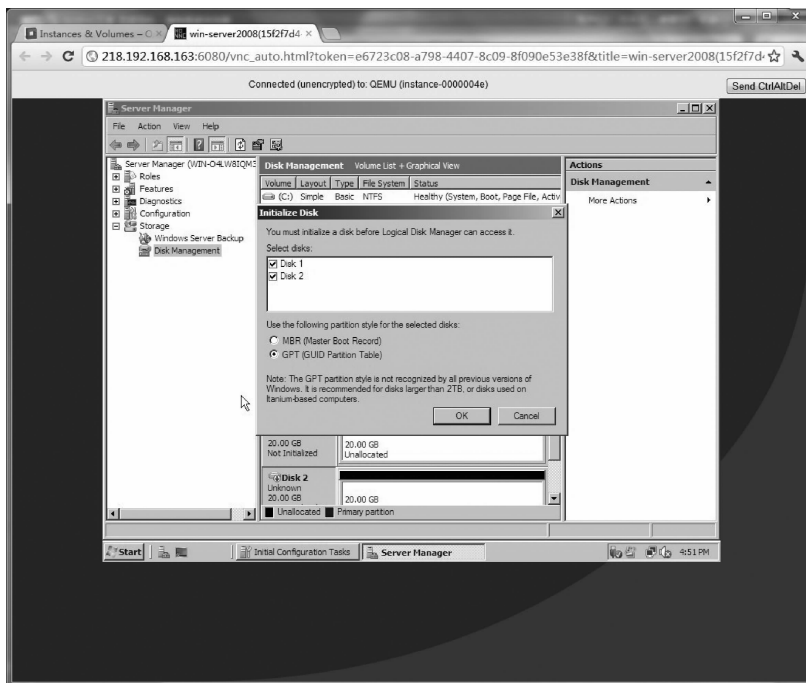


图 11 在 Windows 实例挂载云硬盘

创建分区,通过 mkfs 工具将云硬盘格式化、挂载并使用云硬盘,如图 12 所示。

```
command (m for help): p
disk /dev/vdb: 32.2 GB, 32212254720 bytes
16 heads, 63 sectors/track, 62415 cylinders
units = cylinders of 1008 * 512 = 516096 bytes

   Device Boot      Start         End      Blocks   Id  System
   /dev/vdb1             1         62415     31457128+  83  Linux

command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table
# mkfs.ext3 /dev/vdb
vdb: vdb1
# mkfs.ext3 /dev/vdb1
mke2fs 1.41.12 (17-May-2010)
Filesystem label=
OS type: Linux
block size=4096 (log=2)
fragment size=4096 (log=2)
stride=0 blocks, stripe width=0 blocks
1966080 inodes, 7864282 blocks
393214 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=4294967296
240 block groups
32768 blocks per group, 32768 fragments per group
8192 inodes per group
superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208,
        4096000

Writing inode tables: done
Creating journal (32768 blocks): done
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 35 mounts or
180 days, whichever comes first.  use tune2fs -t or -i to override.
# mknod /mnt/lee
# mount /dev/vdb1 /mnt/lee
```

图 12 在 Linux 实例挂载云硬盘

由于 Swift 内置冗余和容错机制,数据的丢失率很低;私有云与用户位于同一网段内,网络传输速度很高;私有云平台位于防火墙内,受到攻击的几率较小,且不同用户组之间存储的数据看不见,数据私密性较高,因此云存储是 OpenStack 私有云平台的一个重要功能。它的使用步骤为:创建容器,在容器内上传文件,上传文件时指定保存对象的名字并选择文件路径。

综上所述,本文搭建的 OpenStack 私有云平台具有使

用简单、按需选择实例类型、快速部署虚拟实例、高效安全地存储数据等特性,具有实际意义。

## 6 结束语

本文提出了一种基于 OpenStack 构建私有云计算平台的方案,旨在为科研实验室以及中小企业提供弹性计算服务,满足不同用户、不同阶段的需求。同时,通过在虚拟镜像中集成各种开发环境和开发工具,可以提供高可用性的云服务。此外,本文还分享了一些部署 OpenStack 云平台过程中的经验。

本文在部署、使用 OpenStack 云平台过程中,发现 OpenStack 对数据库有很强的依赖性,且网络性能并非最优,因此,如何提高数据库的容错性、稳定性以及改善网络性能,是进一步优化私有云的研究方向之一。

## 参考文献

- 1 NIST Cloud Computing Program. <http://www.nist.gov/itl/cloud/>, 2011
- 2 Ramgovind S, Eloff M M, Smith E. The management of security in cloud computing. Proceedings of Information Security for South Africa (ISSA), Sandton, South Africa, 2010:1~7
- 3 吴朱华. 云计算核心技术剖析. 北京:人民邮电出版社, 2011
- 4 董鑫. 开源云计算技术应用. 计算机光盘软件与应用, 2011(16)
- 5 OpenStack Community. <http://www.openstack.org/>, 2011



- 6 git·hub. <https://github.com/>, 2012
- 7 Swift 1.5.1-dev documentation. <http://swift.openstack.org/index.html>, 2011
- 8 李刚建. 基于虚拟化技术的云计算平台架构研究. 吉林建筑工程学院学报, 2011, 28(1):79~81
- 9 张帆, 李磊, 杨成胡等. 基于 Eucalyptus 构建私有云计算平台. 电信科学, 2011, 27(11):57~61

**[作者简介]** 李小宁, 华南理工大学硕士研究生, 主要研究方向为云计算、移动互联网终端技术、图像处理; 李磊, 工程师, 主要研究方向为云计算、通信与信息处理等; 金连文, 博士, 华南理工大学教授、博士生导师, 主要研究方向为图像处理、智能信息处理、模式识别、云计算等; 黎德生, 华南理工大学硕士研究生, 主要研究方向为云计算、移动互联网终端技术。

## Constructing a Private Cloud Computing Platform Based on OpenStack

Li Xiaoning, Li Lei, Jin Lianwen, Li Desheng

(School of Electronic and Information Engineering, South China University of Technology, Guangzhou 510640, China)

**Abstract** Cloud computing is a very popular research direction in the IT field in recent years, and private cloud is very important in cloud computing because it's safety, controllability and effective use of IT equipment resources. This paper introduces the architecture of an open source cloud computing platform OpenStack and the relationship between five functional modules in OpenStack. This paper also highlights a method of building private cloud based on OpenStack. In addition, this paper shares the problems met in building the private cloud platform and the solutions of solving the problems. In the end, this paper analyzes the security and flexible computing resource allocation capabilities of the private cloud platform.

**Key words** private cloud computing, OpenStack, flexible resource allocation

(收稿日期: 2012-08-15)

### ·简讯·

#### 烽火通信中标中国电信 ODN 集采项目

近日, 备受关注的中国电信 ODN 产品集采结果揭晓, 烽火通信凭借对 ODN 的深刻理解、优异的综合实力以及良好的品牌形象, 从近百个厂商中脱颖而出, 获得较高集采份额, 光缆交接箱、光纤配线架、室外机柜等系列产品的综合排名均位列前茅, 其中光缆交接箱产品排名 A 类第四名。

随着“宽带中国”战略的推进, 中国电信继前期的大规模宽带集采后, 又将重点投向了建设难度大、使用寿命长的 ODN 建设, 并组织了大规模集采。通过对商务、技术、服务等多方面因素的综合评比, 烽火通信在此次集采项目中取得了理想成绩, 特别是无跳接光缆交接箱解决方案是最大亮点。

烽火通信面向 FTTx 网络设计的无跳接光缆交接箱解决了分光器扩容安装以及跳纤过多、调度不易的问题。其

设计理念为整体架构尽量向传统光交靠近, 一方面产品结构件通用化程度高, 减少产品成本; 另一方面施工操作方式与传统光交类似, 易于使用, 施工人员能快速适应并上手操作; 在产品形态上兼顾不同地区的建设方式, 线路容量、内置分光器类型等功能选择基本涵盖国内外差异化需求; 产品质量秉承烽火通信 ODN 系列产品一贯优点, 满足或超出相关标准要求。

继 2011 年首次小规模将无跳接光缆交接箱纳入光缆交接箱的集采后, 2012 年中国电信进一步丰富了无跳接光缆交接箱的形态和需求, 并扩大了应用场景。从集采的总量来看, 其需求已经远超传统光交, 进一步凸显出无跳接光缆交接箱对优化 ODN 起到的积极作用。烽火通信此次在 ODN 的各类主要箱体产品上获得突破, 将为 ODN 行业的技术创新带来新的活力, 进一步夯实其在 ODN 领域中的品牌形象。