



一种云环境下的镜像文件存储系统^{*}

王洪发

(浙江水利水电学院计算机与信息工程系 杭州 310018)

摘 要:随着云计算的发展进入主流,在云资源池中的虚拟机数量也越来越多,因此为满足负载平衡和节省能源的需求,怎样快速低成本地在不同主机之间迁移虚拟机成为了研究者研究的一个热点问题。设计了一个分布式镜像文件存储系统——IFSpool。基于 Lustre 的该存储系统能够完成 3 个目标:大容量、高扩展性地支持成千上万个虚拟机镜像,高 I/O 吞吐率的数据访问和针对虚拟机启动风暴的启动优化。IFSpool 紧密地结合队列和缓存机制,并通过协调数据访问顺序提高 I/O 吞吐率和减轻启动风暴。IFSpool 的 I/O 带宽和存储容量能容易地通过动态增加存储节点和服务器扩充。实验结果表明,IFSpool 对于虚拟机迁移是非常有效的。

关键词: 镜像文件存储;云计算;虚拟化;数据存储;Lustre;I/O 调度

doi: 10.3969/j.issn.1000-0801.2014.02.011

A Storage System for Image File in Cloud Computing

Wang Hongfa

(Department of Computer and Information Science, Zhejiang University of
Water Resources and Electric Power, Hangzhou 310018, China)

Abstract: As cloud computing enters the mainstream, the numbers of virtual machine in one cloud pool is becoming more and more. So, how to quickly and cheaply migrate virtual machines between different hosts for load balance and power savings is becoming one of the hot and tough problems in this filed for the researchers. A distributed storage system was presented, which was called IFSpool. The system was based on Lustre and was designed to achieve the following three goals: high capacity and high extensibility to support thousands of VM images, high I/O throughput for data access, and booting optimization for boot storm. IFSpool combined a queue mechanism and a cache mechanism in a seamless way, to improve I/O throughput and lighten the boot storm by coordinating data access order. The I/O bandwidth and storage capacity of IFSpool can be easily aggregated and extended by adding storage nodes and servers dynamically. The experimental results show that IFSpool is effective and efficient for VM migration.

Key words: image file store, cloud computing, virtualization, data storage, Lustre, I/O dispatch

1 引言

云计算的本质是按需提供服务给组织或个人^[1]。云服务提供商采用的服务方式通常是以虚拟机形式动态分配

计算资源给用户使用,例如 Amazon Elastic Compute Cloud、OpenStack、Eucalyptus、Nimbus 和 OpenNebula。然而,随着在一个云池中的虚拟机数量增多,它会引起以下两个问题^[2,3]。

^{*} 浙江省自然科学基金资助项目(No. Y12F030062)

- 因为虚拟机镜像难以被共享或移动,一台虚拟机很难在不同主机上运行。
- 用户的虚拟机镜像和数据是不安全的,因为它们不是分开存储,并且缺乏容灾性和恢复机制。

因此,设计一个虚拟机镜像文件存储系统是非常必要的,更重要地是镜像文件存储系统能够为用户提供可伸缩、高性能、低成本的数据存储池。

目前,在云计算数据中心有3种模式的镜像存储:本地存储、集中共享存储和网络存储。本地存储是最便捷的,但是容量有限,并且镜像文件不能在不同主机之间共享,使得虚拟机迁移变得很困难。集中共享存储能提供优秀的I/O性能和容易的虚拟机迁移,例如使用光纤或 Infiniband^[4]连接主机和磁盘阵列,但是集中共享存储非常昂贵并且可扩展性不好。网络存储是由许多廉价的存储设备组成,它们之间通过TCP/IP以太网连接,并且使用分布式文件系统或并行文件系统。

在本文中,笔者设计并实现了一个基于Lustre^[5]的分布式镜像文件存储系统——IFSpool(image file store pool)。开源实现的Lustre能够提供高I/O性能和优秀的可扩展性和可靠性,并且被成功部署在商业应用环境,而笔者所做的主要工作是在主机服务器和Lustre之间添加存储适配模块。利用IFSpool中紧密的结合队列和缓存机制以及通过协调数据访问顺序来提高I/O吞吐率和减轻启动风暴。实验结果表明IFSpool能有效地满足镜像文件存储的需求。

2 镜像文件存储的需求

传统上,云服务提供商提供3类服务模式^[6,7]:基础设施即服务(IaaS)、平台即服务(PaaS)和软件即服务(SaaS)。

在IaaS模式中,云服务提供商提供虚拟机、存储、网络等服务。在PaaS模式中,云服务提供商提供计算平台和解决方案。通常包括操作系统、编程语言执行环境、数据库和Web服务器。应用程序开发人员无需购买和管理底层硬件和软件层,即可在云平台上开发和运行他们的软件解决方案。在SaaS模式中,云服务提供商提供应用服务,云用户通过网络、虚拟桌面或移动终端访问应用软件。

然而,随着在云资源池中虚拟机数量的快速增加,很多问题出现在存储领域,如容量不足、数据安全、吞吐量限制和启动风暴。

Meyer D等^[8]提供了一个精心设计的单一存储服务。在这个设计中,存储虚拟机和通用虚拟机被部署在每一个应

用程序节点。存储服务遵循POSIX I/O接口,由虚拟化提供的封装,通用虚拟机可以访问不同的存储服务和使用本地磁盘。对于一个小型虚拟机集群,这种存储系统能在共享存储设备和廉价商用本地磁盘上表现出良好的性能,然而,当虚拟机数量超过单个的存储容量时,它甚至不能满足正常的虚拟机I/O存取操作,此外,它不允许用户之间合作共享。

Liao X F等^[9]设计了一个分布式的存储系统VM Store。VM Store采用直接块索引结构来做虚拟机快照,能显著改善虚拟机的启动性能,并通过动态更改存储节点的数量,提供分布式存储的可伸缩性I/O带宽。VM Store还提出了一种智能对象的分区技术的数据预处理策略,这将更有效地消除重复数据,然而,实验结果表明VM Store不适合大规模的VM镜像存储。

Dickens P M和Logan J^[10]提出一种新的数据再分配模型,采用“多对多”通信模型能够有效地减少OST(对象存储目标)存储资源的竞争访问,缺点是降低了I/O并行性。

Casale G等^[11]提出一种简单性能模型来预测虚拟化应用程序的存储I/O性能整合的影响,并且还定义了对吞吐量、响应时间和混合的读/写请求的简单线性预测模型,但是预测模型的信息来源仅仅基于个人虚拟机隔离实验的信息收集。

为了减轻启动风暴问题,一些虚拟机系统部署在SAN或其他昂贵的存储阵列上,例如VMFS^[12]。然而,随着用户的增多,即使使用先进的存储系统降低启动风暴的发生率,系统升级还是变得很困难而且代价也很高。

因此,有必要建立一个专有的镜像文件存储系统。设计这样一个镜像文件存储系统需要考虑3个基本问题。

(1) 高I/O吞吐量和高可扩展性

在共享存储中,虚拟机共享同一个物理存储池。当虚拟机总访问负载变得更高时,对I/O访问的竞争量将影响虚拟机的性能。在这种情况下,镜像文件存储必须有高的I/O吞吐量。此外,为存储和管理成千上万的虚拟机镜像,镜像存储池可以在不打断任何操作的前提下通过添加新的存储设备增加容量。

(2) 启动优化

在云计算环境中,每个虚拟机将从数据中心主机或共享存储本地文件系统中读取引导信息。在参考文献[9]中,提到在应用程序节点上启动一个虚拟机至少需要10 Mbit/s



的 I/O 带宽,在企业中成百上千台虚拟机并发启动所需要的总带宽将超过存储服务器能力的几倍。

在大多数情况下,虚拟机被部署在企业环境中为企业员工提供相同的使用环境。员工将在上班开始时启动虚拟机并在下班时关闭虚拟机。因此,虚拟机启动或关闭很大程度上是并发操作。为了解决这个问题,结合队列和缓存机制,并通过协调数据访问顺序提高 I/O 吞吐率和减轻启动风暴,进一步通过使用固态硬盘(SSD)作为缓存队列的存储介质来提高读写性能。

(3)容灾备份

在云计算环境中,失效是一种常见的事。2010年1月,Ruby-on-Rails 应用程序托管公司在 Amazon EC2 托管的 22 台虚拟机出现故障。据报道由于这次路由器故障所引起的中断影响了托管在该公司的 44 000 个应用程序的使用^[13]。因此,对于存储和管理成千上万的虚拟机镜像的存储系统,容灾备份是一个重要的考虑因素,包括访问控制、灾难恢复工具和冗余的数据存储。所以,分布式存储系统是一个合适的选择。

3 IFSpool 设计和实现

本节将描述专有镜像文件存储系统 IFSpool 的设计和实现,并给出对提高 I/O 吞吐量和减轻启动风暴的重要改进内容。

3.1 IFSpool 的架构

图 1 展示了 IFSpool 的高层体系结构。每个部件的详细讨论如下。

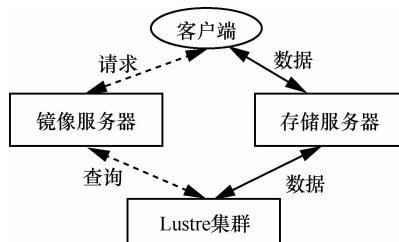


图 1 IFSpool 体系结构

镜像服务器是客户端与 Lustre 之间的通信桥梁。镜像服务器提供虚拟机镜像的发现、登记和交付服务。在本文中,采用 OpenStack Glance^[14]作为 IFSpool 的镜像服务器。Glance 允许上传多种格式的私有和公有的镜像,包括 VHD (Hyper-V)、VDI、qcow2 (Qemu/KVM) 和 VMDK (VMware)。

Lustre^[15]是一个构建在廉价商用机器上的存储系统。如图 2 所示,元数据服务器(MDS)存储着每个文件的布局、

大小和位置,客户端从 MDS 获得文件布局,对象存储服务器(OSS)为一个或多个本地对象存储目标(OST)提供文件 I/O 服务和网络请求处理,OST 把文件数据作为数据对象存储在一个或多个 OSS 上,单个 Lustre 文件系统通常有多个 OSS 和 OST,为了优化性能,每个文件被分为几个子文件并存储到不同的 OST。在 Lustre 中一个来自客户端的数据访问请求将被分解成几个更小的子请求,然后存储服务器并行访问子请求的数据。

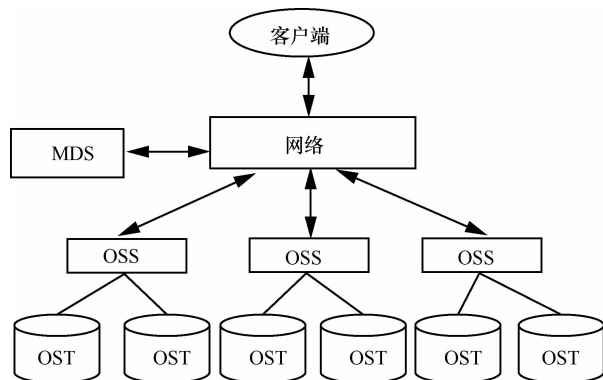


图 2 Lustre 的体系架构

3.2 存储适配器设计

存储适配器是 IFSpool 的主要部件,在 IFSpool 中,客户端也有相对应的部件:元数据客户端(MDC)、对象存储客户端(OSC)和逻辑对象卷(LOV)。MDC 是一个客户端接口模块,可以通过它访问 MDS。

OSC 和 OST 的关系是一对一的。一组 OSC 被包装成一个 LOV,如图 3 所示。

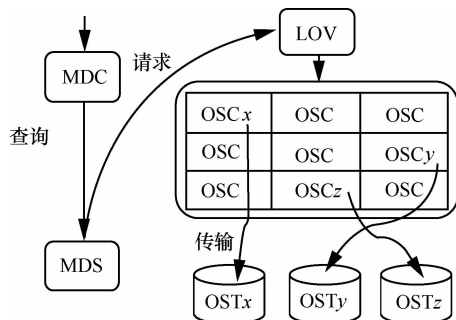


图 3 Lustre 文件系统的文件打开、读取、写入过程

当一个客户端创建、打开或读取一个镜像文件时,它首先获取存储在 MDS 的元数据信息。在 MDS 返回镜像文件的位置后,客户端会直接向相关的 OST 设备发出 I/O 访问请求。例如,在读取文件之前,客户端将通过 MDC 向 MDS 发出查询请求。MDS 告诉客户端从<OSTx, OSTy, OSTz>设备读取文件。然后客户端直接访问<OSTx, OSTy,

OST₂>设备。

假设有 4 个虚拟机 < VM1, VM2, VM3, VM4 >, 为了简单起见, 假设每个虚拟机镜像存储在 4 个 OST 设备上。图 4 显示了这些虚拟机存储在 4 个不同的 OST 上的布局。

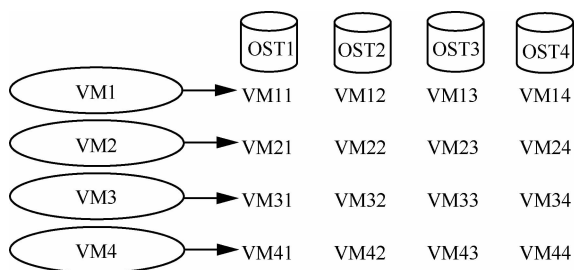


图 4 OST 的镜像文件视图

假设在某个特定时间这 4 个虚拟机同时读取镜像文件, 这样虚拟机将会同时与每个 OST 通信, 因此会产生相当大的通信开销, 而随着虚拟机数量的增加, 性能会大幅下降。主要因为虚拟机的随机磁盘访问, 而且虚拟机本身也不执行一系列连续的磁盘 I/O 操作, 这将导致随机、无序磁盘查找操作, 从而产生额外的磁盘开销和降低了文件访问的并行操作效率。

存储适配器的基本原理是对于每个 OST 维护一个缓存队列, 称为 OST 队列。这些缓存队列存储于存储适配器服务器上并且维护一个全局散列表, 其中散列键也称为 OST_ID (每个 OST 在 Lustre 集群里有一个对应的身份证号码)。因为适配器服务器的存储介质是 SSD, 所以缓存队列拥有高速的读写速度。

在 IFSpool 中, 镜像文件是以条带方式存储在几个 OST 上, 并且在 OST 上一个文件的所有数据被视为一个对象, 该对象使用 128 bit 的标识作为对象 ID。与传统的 UNIX 文件系统不同, MDS 的索引节点并不指向数据块, 相反它指向一个或多个与对象相关的文件, 所以客户端可以通过对象 ID 并行地读写文件, 换句话说, 客户机可以通过

对象 ID 直接和 OST 交换数据。

假设这 4 个虚拟机同时与这 4 个 OST 交换数据。存储适配器首先从虚拟机接收所有数据, 然后依据对象 ID 分派这些数据到相应的 OST 队列。当已使用空间达到一个阈值时, OST 队列中的数据将按序被写回到相应的 OST, 同样。当从 OST 上读取数据时, 对象首先被分派到相应的 OST 队列中, 然后每个虚拟机依据 OST_ID 从 OST 队列取回对象。图 5 显示了存储适配器模型。

存储适配器的优点是通过结合队列和缓存机制将随机访问变成顺序访问。存储适配器的缓存机制实现是不同于传统的缓存机制, 具体地说, 存储适配器没有使用替换算法, 相反适配器采用有一个阈值来决定什么时候回写数据。对于一个写操作, 缓存的功能是将随机写入变成顺序写入, 并且把小的数据块合并成大的数据块。对于读操作, 依据对象 ID 将相同的请求合并到一个特定的 OST 上, 读操作的 I/O 吞吐量可以有效地改善, 并且可以以大数据块形式顺序读取这些数据到相应的 OST 队列上。

存储适配器算法描述如下。

$Queue(i)$ 表示存储 OST_ID 为 i 的数据对象的队列, 队列的范围从 0 到 $n-1$, n 是 OST 的最大数。存储适配器使用时间片轮转方法为每个队列提供服务。Read_object(p , ost) 表示从 ost 中读取对象 p 。Insert_object(p , i) 表示将数据对象 p 加入队列 i 中。Dispatch_object(p , i) 表示将队列 i 中的数据对象 p 发送到本地驱动程序。

Begin

Writing: on arrival of data object p

$Queue(i) = \text{hash}(\text{object } p)$;

if (there is no this object exist in this queue)

{if (no free buffers left $Queue(i)$)

{ $Queue(i) = \text{must_queue}$; return -1;}

else {Insert_objcet(p , i); return 1;}}

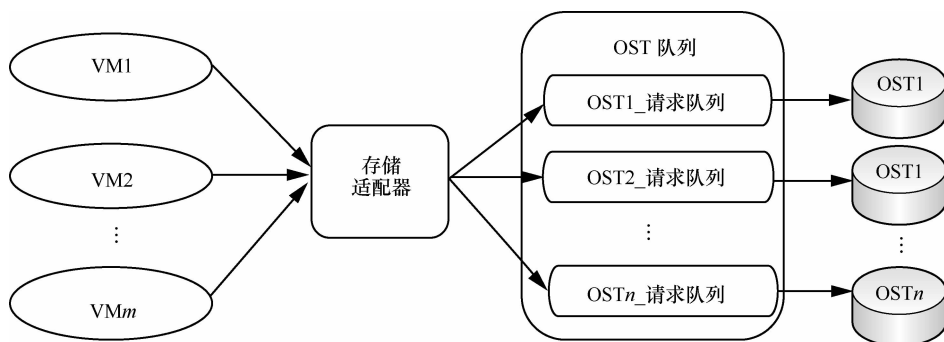


图 5 存储适配器模型



```

return 0;
Reading: on arrival of several object requests
    m_Object_ID = merge requests by Object ID;
    Read_objcet(m_Objcet_ID, OST_ID);
    wait for VMs read.
Dispatch:
    While(true) do{
        if ((threshold value == ture) || (Queue(i) == must))
            { if ( Queue(i) is not empty )
              {get a data object from OST_queue;
               Dispatch(p, i);}
            else if (there is enough slice left to wait for one
requests)
              {do{wait for data object p;
                 Dispatch(p, i);}
               While(time slice for Queue(i) is enough)}
            else {get next queue for service;}}
    End

```

4 实验结果

4.1 实验环境

计算服务器采用两台联想 Think Server TS430 S1280 服务器, Intel Xeon E3-1200 8 核处理器, 32 GB 内存, 两块 Broadcom NetXtreme II 57091 Gbit/s 以太网卡, 6 TBSAS6 硬盘。

镜像服务器配备一台戴尔电脑, Intel Xeon E7550 4 核处理器, 8 GB 内存, 380 GB SATA 硬盘。

Lustre 集群由 4 台戴尔电脑组成。每台电脑配置 Intel Xeon E7550 4 核处理器、4 GB 内存、2 块 500 GB SATA 西部数据硬盘。

存储适配服务器配备一台戴尔电脑、一块 OCZ Vertex3 60 GB 的 SSD、一块 Highpoint RocketRAID 4320 RAID 控制器。

4.2 实验分析

图 6 和图 7 分别显示了在使用 HDFS、MooseFS、Lustre 和 IFSpool 作为镜像存储池时指定数量的虚拟机平均随机读写速率。从图 6 和图 7 看到 IFSpool 具有更好的 I/O 吞吐量。Lustre 和 IFSpool 结果进一步确认存储适配器提高了读写速率, 这证实了存储适配器能够提高并行数据访问速度和降低通信开销。

笔者还测试了不同记录大小的 I/O 性能。在测试环境

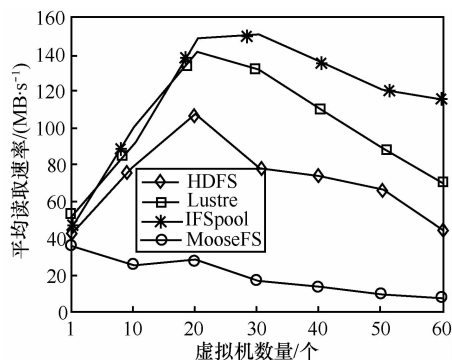


图 6 平均读取速率比较

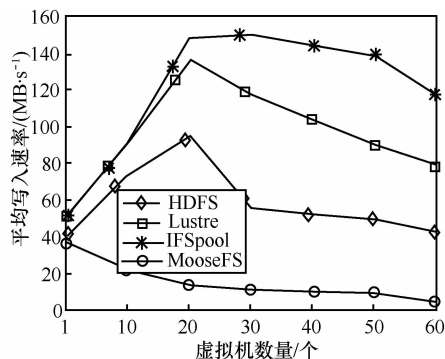


图 7 平均写入速率比较

中有 30 个虚拟机同时针对 IFSpool 作随机读写操作。从图 8 和图 9 看到存储适配器能提高读写速率, 尤其是小的记录, 这也证实了存储适配器能提高数据访问速度并行度和有效地减少通信开销。

在图 10 和图 11 中, 比较了同时启动或关闭特定数量虚拟机所消耗的时间。在实验中, 每台虚拟机配置 2 个 VCPU、1 024 MB 内存、20 GB 磁盘映像和 Windows XP, 所有虚拟机都是 VHD 格式, 并且都运行在同一台主机服务

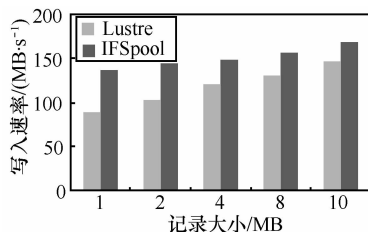


图 8 不同大小记录的写入速率比较

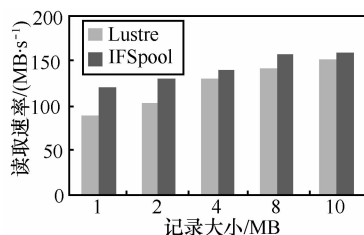


图 9 不同大小记录的读取速率比较

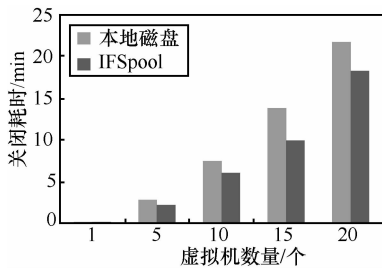


图10 多台虚拟机同时启动的时间消耗

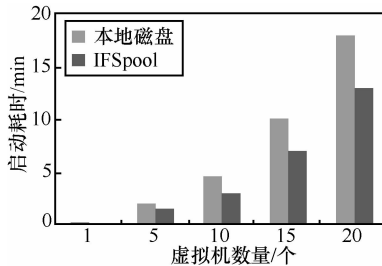


图11 多台虚拟机同时关闭的时间消耗

器上。当登录界面出现时,虚拟机启动完成;当电源关闭时,虚拟机关闭完成。

从图10和图11可以看到当启动或关闭多台虚拟机时,IFSpool能比本地磁盘消耗更少的时间,这是由于IFSpool存在并行访问和缓存机制。

5 结束语

镜像文件存储是云计算的关键部分。如果实现一个高容量、高可伸缩性和高吞吐量的镜像文件存储系统,可以很容易地不同主机或不同数据中心之间迁移虚拟机以达到负载均衡和节省电能,同时可以通过高I/O吞吐量和高容量存储池来解决启动风暴。在本文中,笔者设计并实现一个基于Lustre的存储系统IFSpool,它可以在无须中断数据访问的前提下,通过添加存储设备来增加存储容量和I/O带宽。IFSpool结合队列和缓存机制,并通过协调数据访问顺序提高I/O吞吐率和减轻启动风暴。实验结果也表明IFSpool是有效的镜像文件存储系统。

此外,当有成千上万的用户时,数据中心会存在大量的冗余数据。未来的工作,将在IFSpool中增加对重复数据删除的支持。

参考文献

- 1 Foster I, Zhao Y, Raicu I, *et al.* Cloud computing and grid computing 360-degree compared. Proceedings of IEEE Grid Computing Environments Workshop, Austin, TX, USA, 2008: 1~10
- 2 Wood T, Shenoy P, Venkataramaniand A, *et al.* Black-box and gray-box strategies for virtual machine migration. Proceedings of the 4th USENIX Conference on Networked Systems Design & Implementation, Cambridge, MA, 2007:1~14
- 3 Shafer J. I/O virtualization bottlenecks in cloud computing today. Proceedings of the 2nd Conference on I/O Virtualization, USENIX Association Berkeley, CA, USA, 2010
- 4 Hansen J G, Jul E. Lithium: virtual machine storage for the cloud. Proceedings of the 1st ACM Symposium on Cloud Computing, New York, NY, USA, 2010: 15~26
- 5 Wang F, Oral S, Shipman G, *et al.* Understanding Lustre file system internals. Technical Report ORNL/TM-2009/117, Oak Ridge National Lab, National Center for Computational Sciences, 2009
- 6 Mell P, Grance T. NIST definition of cloud computing. National Institute of Standards and Technology, 2011
- 7 Buyya R, Broberg J, Goscinski A. Cloud Computing: Principles and Paradigms. New York: Wiley Press, 2011
- 8 Meyer D, Aggarwal G, Cully B, *et al.* Parallax: virtual disks for virtual machines. Proceedings of the ACM SIGOPS/EuroSys European Conference on Computer Systems, Glasgow, Scotland, 2008
- 9 Liao X F, Li H, Jin H, *et al.* Vmstore: distributed storage system for multiple virtual machines. Science China Information Sciences, 2011, 54(6):1104~1118
- 10 Dickens P M, Logan J. A high performance implementation of MPI-IO for a Lustre file system environment. Concurrency and Computation: Practice and Experience, 2010
- 11 C asale G, Kraft S, Krishnamurthy D. A model of storage I/O performance interference in virtualized systems. Proceedings of the 1st Int Workshop on Data Center Performance, Minneapolis, MN, 2011
- 12 Vaghani S B. Virtual machine file system. ACM SIGOPS Operating Systems Review, 2010, 44(4):57~70
- 13 Rajagopalan S, Cully B, O'Connor R, *et al.* SecondSite: Disaster Tolerance as a Service. ACM 978-1-4503-1175-5/12/03, London, England, 2012
- 14 Openstack image-service. <http://www.openstack.org/projects/image-service/>
- 15 Gens F. Defining "Cloud Services" and "Cloud computing". <http://www.fiercetelecom.com/story/defining-cloud-computing-services-benefits-and-caveats/2012-03-12>, 2012

[作者简介]



王洪发,男,浙江水利水电学院教授,主要研究方向为计算机网络、优化计算。

(收稿日期:2013-09-17)