OpenStack 云平台的高可用设计与实现

罗 兵 谯 英 符 晓
(西南石油大学计算机科学学院 成都 610500)

摘 要 实现高可用性是 OpenStack 云计算管理平台研究的重要问题之一。针对 OpenStack 云计算管理平台的相关服务组件运行在单节点上易导致单点故障(SPoF)的问题,结合现有多种系统高可用性解决方案,提出一种基于 Pacemaker+Corosync+HAProxy+Ceph 的解决方案以实现 OpenStack 云计算管理平台的高可用。该方案将 Active-Active 的双活模式、Active-Passive 的主备模式及集群技术 3 种高可用设计模式融合在一起,通过软硬件冗余和服务实例故障转移等方式实现 OpenStack 云计算管理平台的高可用性。实验证明,在少量节点或链路中断的情况下 OpenStack 云计算管理平台仍然能够稳定运行,该高可用性方案具有可行性。

关键词 云计算,OpenStack,高可用性,单点故障,云平台
中图法分类号 TP393.09 **文献标识码** A

Design and Implementation of High-availability Based on OpenStack Cloud Platform

LUO Bing QIAO Ying FU Xiao
(School of Computer Science, Southwest Petroleum University, Chengdu 610500, China)

Abstract Achieving high availability is one of the most important issues in the study of the OpenStack cloud management platform. In order to solve the problem that related services for OpenStack cloud management platform components running on a single node can cause single points of failure (SPoF), we combined with existing high-availability solutions, put forward a method based on Pacemaker + Corosync + HAProxy + Ceph to realize the high availability of OpenStack cloud management platform. This solution combines with Active-Active main/standby mode, Active-Passive Double live mode and cluster technology together, and then uses the method of hardware and software redundancy and transfer failure of service instance, finally, comes up with the high availability of OpenStack cloud management platform. Experiments show that in the case of the system only having fewer nodes or lose of link, the method is effective.

Keywords Cloud computing, OpenStack, High-availability, SPoF, Cloud platform

1 引言

随着信息技术的发展,云计算正在成为互联网的新兴计算模式。云计算是一种面向服务的商业计算模型,可以为用户提供各种资源如计算资源、网络资源、存储资源等的按需服务[1-2]。对于拥有大量计算机资源的高校来说,利用 IaaS 云平台的高效、弹性的计算资源分配能力以及简易的硬件管理要求,实现高校计算机资源的合理利用,能够满足多方面学科教学和科研应用的资源需求。目前,在业界处于领先地位的开源云计算项目有 OpenStack[3],Eucalyptus,OpenNebula 和 CloudStack[4-8] 等,其中 OpenStack 的关注度最高且发展最快[9]。与其他开源云计算管理平台相比,OpenStack 具备以下的优势:OpenStack 是松耦合的结构,功能模块清晰,可以根据序用户需求实时增减功能组件节点;OpenStack 基于 Representational State Transfer(RESTful)提供应用程序接口(API),用户可以很方便地对其进行二次开发。

高可用性 HA(High-Availability)是指通过尽量缩短因日常维护操作(计划)和突发的系统崩溃(非计划)所导致的停机时间,以提高系统和应用的可用性[10]。系统 HA 是目前企业防止核心计算机系统因故障而宕机的最有效手段。云平台的高可用性已成为云平台相对于传统应用服务体现的众多优势之一,如何实现系统高可用性是设计云平台时需要重点考虑的要素之一。

近年来,已有很多企业针对不同 OpenStack 云计算管理平台提出了多种系统高可用解决方案。Mirantis OpenStack 平台的控制节点采用 Keepalived+HAProxy 实现平台的 Active-Active 模式高可用,数据库、消息队列及网络节点采用 Pacemaker+Corosync 方案实现平台的 Active-Passive 模式高可用,Oracle 公司的 OpenStack 平台的所有节点使用 Active-Passive 模式实现高可用。该方案只提供两节点 Active-Passive 方案,其可靠性和 CTO 比不上三节点方案,需要使用共享存储比如 NFS 来实现 Active-Passive 模式的数据库和消息队列,容易产生脑裂。

为了实现基于 OpenStack 而构建的云计算管理平台的高

本文受四川省教育厅专项基金 018-数字油田监控(020402000018)资助。
罗 兵(1989-),男,硕士生,CCF 会员,主要研究方向为云计算、软件定义网络;谯 英(1972-),女,博士生,副教授,主要研究方向为油气田生产物联网、云计算;符 晓(1988-),女,硕士生,主要研究方向为云计算、数据挖掘。

可用性，在构建 OpenStack 云计算管理平台时从硬件、软件、策略及管理等多个方面、多维度进行分析及设计，提出一套基于 Pacemaker＋Corosync＋HAProxy＋Ceph 的高可用性解决方案，即将 Active-Active 的双活模式、Active-Passive 的主备模式及集群技术 3 种高可用设计模式融合在一起，通过软硬件冗余和服务实例故障转移等方式实现 OpenStack 云计算管理平台的高可用性，以解决云计算管理平台在出现单点故障时无法正常提供服务的问题，保证 OpenStack 云计算管理平台的高可用性。

2 云平台的高可用性模式

2.1 高可用的工作方式

系统高可用性的构建主要是通过软硬件冗余运行每个服务实例并通过服务实例故障转移实现方式防止系统宕机和数据丢失[11]。如果一个节点上运行的实例出现故障，系统使用故障转移服务将实例转移到另一个节点运行以保证实例持续提供服务。许多服务提供保证服务水平协议(SLA)[12]，包括计算服务正常运行的百分比——基于可用时间和系统宕机时间进行计算，不包括计划维护时间。高可用性的一个关键方面是消除了单点故障（SPOF）[13-14]。实现系统高可用性的常用 3 种设计模式如下。

主主(Active-Active)：两台主机同时运行相同服务实例工作并相互监测服务状态，当任一台主机宕机时，另一台主机立即接管它的一切工作，保证工作实时可用。Active-Actvie 模式可以在所有服务实例前端增加一个 HAProxy 进行负载均衡服务以实现对客户端的请求进行负载均衡。

主备(Active-Passive)：主机工作，备机处于监控准备状况；当主机宕机时，备机接管主机的一切工作，待主机恢复正常后，按使用者的设定以自动或手动方式将服务切换到主机上运行。Active-Passive 模式可以通过 Pacemaker＋Corosync 的方案来实现。

集群(Cluster)：多台具有相同能力的主机同时对外提供透明服务，所有服务之间都是 Active-Active 关系，并分担处理服务请求，一般通过总控节点或集群软件进行高可用的控制。

2.2 OpenStack 的高可用模式

OpenStack 相关服务组件同时支持多种高可用设计模式，根据 OpenStack 组件服务状态可以选择不同的高可用设计模式来实现 OpenStack 云平台的高可用性。OpenStack 相关服务组件存在两种状态：无状态服务和有状态服务。

无状态服务指多个服务之间的请求没有依赖关系，是完全独立的。无状态服务意味着多个服务实例可以同时运行在不同的节点上，通过监控实例的可用性转发服务请求到可用实例。如果选择的服务实例无法处理请求，则其会在其他服务实例上重试相同的请求。OpenStack 无状态服务包括 nova-api，nova-conductor，nova-scheduler，cinder-api，cinder-scheduler，glance-api，keystone-api 和 neutron-api 等。前端使用 HAProxy 代理和 Active-Active 模式实现 OpenStack 无状态服务的高可用性。

有状态服务是指后续服务实例的请求依赖于之前服务实例的请求结果，在任何时候仅有一个服务实例可以被激活。通常采用 Active-Passive 设计模式来实现有状态系统服务的高可用性，OpenStack 云平台中所用到的 HAProxy 和 Neutron-LBaaS 等类似服务实例组件的高可用性采用 Pacemaker＋Corosync方式实现。OpenStack 所用到的后端服务如数据库(MariaDB)和消息队列(RabbitMQ)则通过构建自身集群来达到高可用性，使用 Pacemaker 创建虚拟 IP(VIP)向 OpenStack 服务组件提供服务访问。OpenStack 相关服务组件的具体高可用实现方式如表 1 所列。

表 1 OpenStack 相关服务组件的高可用实现方式

服务	组件	模式	HA 策略
Support services	HAProxy	A/P	Corosync/Pacemaker
Support services	rabbitMQ	A/A	Application Cluster
Support services	mariaDB-Galera	A/A	Galera Cluster
Support services	memcached	A/A	Service Configuration
Keystone	openstack-keystone	A/A	HAProxy
Glance	openstack-glance-*	A/A	HAProxy
Nova	openstack-nova-*	A/A	HAProxy
Neutron	neutron-server-*	A/A	HAProxy
Dashboard	httpd(horizon)	A/A	HAProxy
Ceilometer	openstack-ceilometer	A/A	HAProxy
Sahara	openstack-sahara	A/A	HAProxy
Neutron	neutron-LbaaS-agent	A/P	Corosync/Pacemaker
Neutron	neutron-dhcp-agent	A/A	Multiple DHCP agents
Neutron	neutron-l3-agent	A/A	DVR

3 OpenStack 云平台的系统架构

云计算有很多种不同类型的部署方式和运营方式，OpenStack 也具有多种功能服务组件。云平台具有多种业务的需求，其中的基本服务包括计算、网络、存储，每种服务都有不同的资源需求，所以云平台架构选择的是通用性云。对于计算、网络、存储等组件来说，云平台的设计须考虑必要的公平权衡。OpenStack 云平台架构主要设计 5 种类型节点：管理节点、控制节点、网络节点、计算节点及存储节点。

(1)管理节点的设计

管理节点的主要功能是监控监测 OpenStack 物理节点的数据，使用 Zabbix 作为生产环境的告警和监控工具，管理节点为 Zabbix Server，OpenStack 组件节点为 Zabbix Agent；使用 LogStash 作为日志聚合工具，管理节点为 LogStash Server，Openstack 组件节点为 LogStash Agent。

(2)控制节点设计

控制节点的主要功能是为云用户和管理员提供图形化管理界面，对计算、网络、存储等节点集群进行控制。控制器节

点主要运行身份认证服务、镜像服务、计算管理 API、网络服务 API 和仪表盘相关服务组件。

(3)网络节点设计

网络节点的主要功能是提供租户访问网络的权限，并且管理和创建网络资源，由 Neutron 组件实现。基于已有的网络基础设施来封装通信路径，以实现给租户的流量分段。这些方法依赖于特殊的实现方式，由 Neutron 服务组件实现的网络 L2 层和 L3 层代理及插件，为租户提供基于 GRE 隧道、VLan 标签、VxLan 封装的高级组网技术，使虚拟机实例连通 WAN 的物理网络基础设施。

(4)计算节点设计

计算节点的主要功能是控制整个云平台的计算资源的调度和管理，实现 OpenStack Compute 的相关服务。运行 Nova-compute 服务组件来操作虚拟机实例。计算节点也通过运行一些网络服务代理组件使虚拟机实例连接到虚拟网络，同时也提供防火墙服务。

(5)存储节点设计

存储节点的主要功能是存储云平台环境所需的数据，包括镜像服务库的磁盘镜像、计算服务的实例以及由块存储服务器创建的持久性存储卷和对象存储提供的分布式对象存储服务。存储后端采用 Ceph 分布式存储来保证存储服务的高可用性和扩展性。OpenStack 云平台架构如图 1 所示。

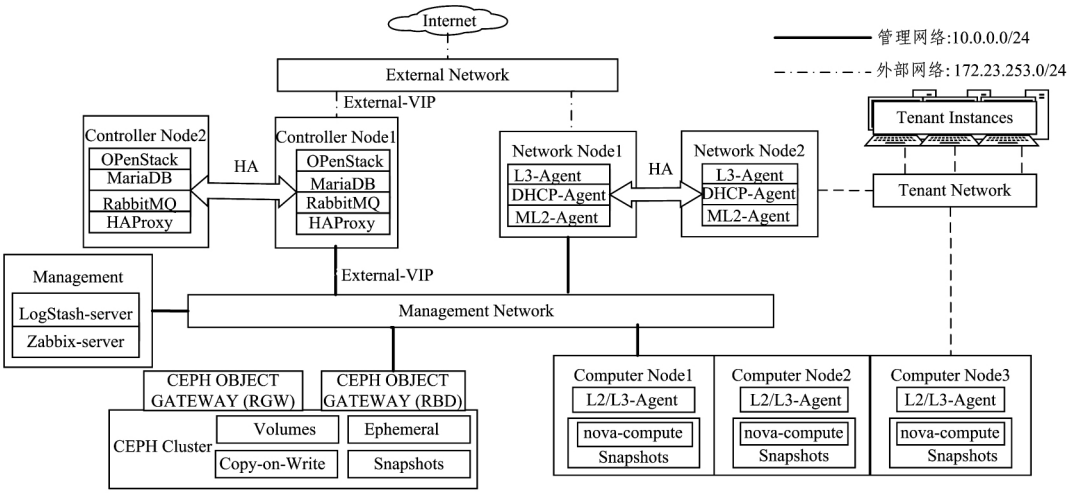

图 1　OpenStack 云平台架构图

4　OpenStack 云平台高可用性的验证

4.1　验证环境

校园云平台采用了曙光天阔 A620r-G 和天阔 A420r-G 服务器，具体型号如表 2 所列。

表 2　服务器型号及数量表

名称	数量	机型	标识
管理节点	1	天阔 A420r-G	Manager
控制节点	2	天阔 A620r-G	Controller01/02
网络节点	2	天阔 A620r-G	Neutron01/02
计算节点	14	天阔 A620r-G	Compute01/…/14
存储节点	14	天阔 A620r-G	Ceph01/…/14

表 3　服务器配置表

节点	配置
管理节点	1 个 AMD Opteron(tm) Processor 4334，1 条 8GB 内存，2 块 300GB SAS 磁盘，1 块 2 端口千兆网卡
控制节点	2 个 AMD Opteron(tm) Processor 6344，2 条 16 GB 内存，4 块 300GB SAS 磁盘(RAID-5)，1 块 2 端口千兆网卡
网络节点	2 个 AMD Opteron(tm) Processor 6344，4 条 16 GB 内存，4 块 300GB SAS 磁盘(RAID-5)，2 块 2 端口千兆网卡
计算节点	2 个 AMD Opteron(tm) Processor 6344，8 条 16 GB 内存，4 块 300GB SAS 磁盘，1 块 2 端口千兆网卡
存储节点	2 个 AMD Opteron(tm) Processor 6344，1 条 16 GB 内存，4 块 300GB SAS 磁盘，1 块 2 端口千兆网卡

测试环境使用 20 台曙光服务器，其中 1 台作为管理节点，2 台作为高可用控制节点，2 台作为高可用网络节点，14 台作为计算节点，14 台作为 Ceph 分布式存储节点。服务器配置 2 个 AMD OPTERON OS6344 CPU，每个 CPU 含有 12 核；内存相应配置为 1600MHz；以太网卡选取 2 端口千兆网卡，如表 3 所列。

4.2　OpenStack 云平台高可用性的实现

该云平台主要实现 OpenStack 云平台的控制节点、计算节点、网络节点和存储节点的高可用性，具体实现方式如下。

4.2.1　Controller 节点的 HA 实现

OpenStack Service API 是无状态的 REST 服务，如 httpd，*-api，neutron-server，glance-registry，nova-novncproxy，keystone 等。OpenStack Service API 由 HAProxy 提供负载均衡，将请求按照一定的算法转到某个节点上，由 Pacemaker 为此服务提供 VIP。因此，OpenStack 控制节点集群可配置为 HAProxy 的后端，实现双活模式。

在 OpenStack 中，HAProxy 程序为 OpenStack Service API 和 MariaDB Galera 服务提供负载均衡，其自身由 Pacemaker＋Corosync 实现 Active-Passive 模式的高可用，由 Pacemaker 为 HAProxy 提供 VIP，在任何时刻只有一个 HAProxy 处于 Active 状态并提供服务。具体实现操作如下：

$ pcs resource create vip ocf:heartbeat:IPaddr2 params ip="Server-VIP" cidr_netmask="24" op monitor interval="30s"

$ pcs resource create lb-haproxy systemd:haproxy-clone

$ pcs constraint order start vip then lb-haproxy-clone kind=Optional

$ pcs constraint colocation add vip with lb-haproxy-clone

Controller 节点高可用架构图如图 2 所示。

OpenStack 支持 Galera 集群作为数据库的同步多主

(Mutli-Master)集群工具,Galera 使用底层并行机制对数据库集群进行读写,实现同步复制。OpenStack 支持多节点写入 Galera 节点,但目前无法在生产环境中稳定有效地运行,因此只采用一个激活的 Galera 节点来配置 HAProxy,Galera 内部支持主/主拓扑结构和写入到任何集群节点。在这样的配置中,只有一个节点是活动的,其余的节点是 standby 节点。Galera 使用同步复制并确保每个集群节点是相同的。

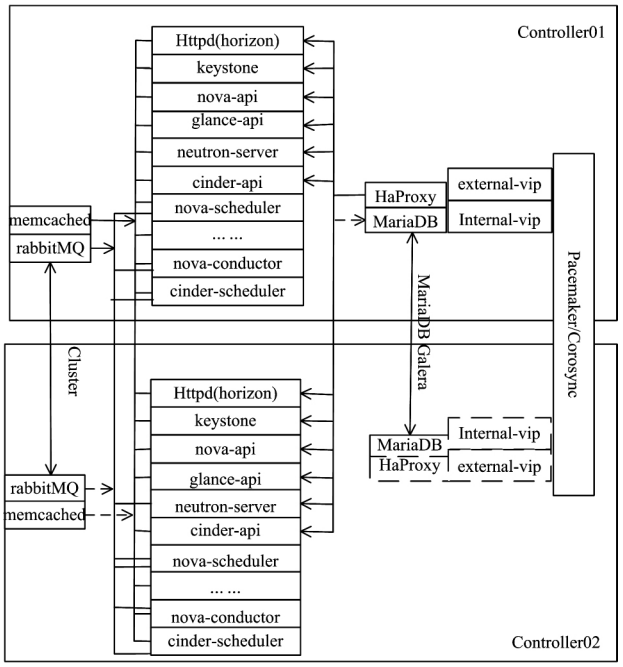

图 2 Controller 节点高可用架构图

RabbitMQ 组件为 OpenStack 提供消息队列,支持 RabbitMQ Clustering 和 RabbitMQ Mirrored Queue 高可用技术,集群内任意一个节点生产的消息都能同步到其他节点的镜像队列,当源节点出现故障时消息由同步节点接管。将 RabbitMQ 的主机列表配置给 OpenStack 相关服务组件,具体配置如下:

rabbit_hosts=controller01:5672,controller02:5672

rabbit_ha_queues=true

Memcached 组件原生支持 Active-Active,只需要在 OpenStack 中配置它的所有节点的名称即可,比如 memcached_servers=controller01:11211,controller02:11211。当 controller01:11211 失效时,OpenStack 服务组件会自动使用 controller02:11211。

4.2.2 Compute 节点的 HA 实现

本文通过使用 Ceph 分布式存储的方式来实现虚拟机实例的热迁移,采用 Ceph 作为分布式文件系统来达到计算节点的高可用性。将 Ceph Monitor 安装配置在 Controller 节点,将 Ceph-Client 安装配置在 Compute 节点。具体的高可用架构图如图 3 所示,配置语句如下:

live_migration_flag = VIR_MIGRATE_UNDEFINE_SOURCE,VIR_MIGRATE_PEER2PEER,VIR_MIGRATE_LIVE,VIR_MIGRATE_TUNNELLED

4.2.3 Network 节点的 HA 实现

Neutron 的高可用采用 OpenStack 原生自带的高可用模式,对不同服务采用不同的高可用实现方式,网络节点的高可用性架构如图 3 所示,具体实现方式如下。

L2 Agent:L2 agent 只在所在的网络或者计算节点上提供服务,因此它是不需要 HA 的。

DHCP Agent:OpenStack 网络服务是一个调度程序,因此可以允许在多个节点之间运行代理,DHCP 协议自身就支持多个 DHCP 服务器,因此只需要在多个网卡控制节点上通过配置每个网络的 DHCP 数量的方式为每个租户网络创建多个 DHCP Agent,修改/etc/neutron/ neutron. conf 文件中的 dhcp_agents_per_network 为 2 或者更大。

LBaaS 高可用性:目前 Neutron LBaaS 代理服务是无法通过其自带的 HAProxy 插件实现高可用的。实现 HAProxy 高可用的常见方案是使用虚拟路由冗余协议(Virtual Router Redundancy Protocol,VRRP),然而 LBaaS HAProxy 插件目前还不支持该协议。因此,只能使用 Pacemaker+DRBD(放置/var/lib/neutron/lbaas/目录)的方式来部署 Active-Passive 方式的 LBaas Agent HA。

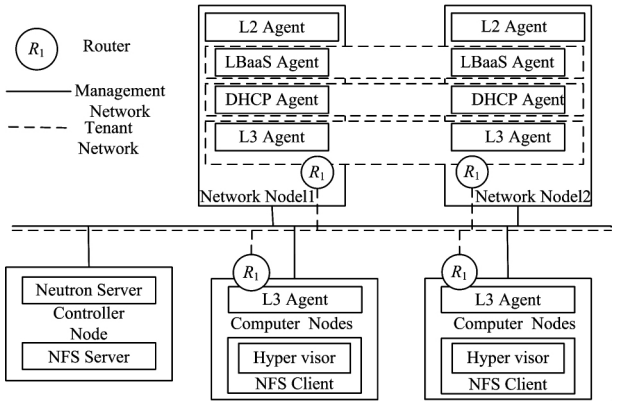

图 3 Compute 和 Network 节点高可用架构图

目前有 3 种方式可以实现 Neutron L3 高可用:1)使用 Pacemaker+corosync 实现 Active-Passive 高可用性;2)为了兼容高可用性、扩展性,使用 VRRP 的 Active-Active 高可用性;3)使用从 Juno 版发布的分布式虚拟路由(DVR)实现 Neutron L3 高可用性。本文则采用分布式虚拟路由(DVR),该方案在网络控制节点上只部署 DHCP 和 SNAT,将 NAT 和 L3 Agent 部署到虚机所在的计算节点,同时解决了 L3 Agent 和 Metadata Agent 的高可用性问题,具体配置如表 4 所列。

表 4 分布式虚拟路由(DVR)配置表

控制节点	/etc/neutron/neutron. conf	router_distributed=True
网络节点	etc/neutron/l3_agent. ini/etc/neutron/plugins/ml2/ml2_conf. ini	agent_mode=dvr_snat enable_distributed_routing=True
计算节点	/etc/neutron/l3_agent. ini/etc/neutron/plugins/ml2/ml2_conf. ini	agent_mode=dvr enable_distributed_routing=True

4.3 OpenStack 云平台高可用性测试

在云平台部署完成后,对该云平台的高可用性性能进行测试,具体测试步骤及结果如下。

1)断开 OpenStack 云平台相关节点的任一条网线,此时出现警告,OpenStack 云平台能正常工作。恢复网络,OpenStack 云平台警告消失,平台恢复正常。

2)断开任一负载均衡节点,OpenStack 云平台能正常提

(下转第 590 页)

一种允许在无共享的系统中部署“内存中”数据库的 Cluster 的技术。通过无共享体系结构，系统能够使用廉价的硬件，而且对软硬件无特殊要求。此外，每个组件有自己的内存和磁盘，不存在单点故障。MySQL Cluster 由计算机集群构成，存储的数据包括：1)馆藏图书期刊目录的订购数据、借阅数据；2)电子期刊、数据库的订购数据；3)文献传递、馆际互借的图书馆数据；4)参考咨询的咨询人员数据。

结束语 本文分析了现有网站相关技术，在分析影响性能、可用性、伸缩性、扩展性和安全性等架构因素的基础上，提出了一套网站架构解决方案，并将其应用于实际系统中。网站的架构会随着性能、可用性、伸缩性、扩展性的提升变得越来越复杂，并且没有唯一性，只有不断地摸索寻找适合自身的拓扑结构。

参 考 文 献

[1] 李钢. 大型公共服务网站架构设计初探[J]. 环球市场信息导报，2015(39)：72.

[2] 胡华海，王新宇. 浅析大型网站的基础架构[J]. 科技风，2009(4)：167.

[3] 林昊. 大型网站架构演变和知识体系[J]. 程序员，2008(11)：66-69.

[4] 韩树河. 大型网站应用技术架构演变的研究[J]. 吉林化工学院学报，2015(1)：53-56.

[5] 杨舟. 浅析大型网站的性能优化[J]. 软件工程师，2010(12)：38-40.

[6] 房辉，常盛. 大型网站高性能架构研究[J]. 信息系统工程，2015(12)：76-77.

[7] 俞华锋. Memcached 在大型网站中的应用[J]. 科技信息(科学教研)，2008(1)：70.

[8] 周建儒. Memcached 在大型网站建设中的应用[J]. 电脑知识与技术，2016(1)：57-59.

[9] 赵丽荣. 大型门户网站运行维护服务模式探讨[J]. 中国高新技术企业，2011(21)：71-73.

[10] 章文嵩. 可伸缩网络服务的研究与实现[D]. 长沙：中国人民解放军国防科学技术大学，2000.

(上接第 566 页)

供服务，说明无单点故障(SPoF)产生。连接断开节点，可以快速重置负载均衡并提供服务。

3)断开任一 OpenStack 云平台控制节点，测试确认 OpenStack 控制节点相关服务组件、数据库和消息队列集群均能提供正常服务，然后依次测试网络节点、计算节点、存储节点。

4)断开 Ceph 集群的少量节点，OpenStack 云平台中的虚拟机实例仍能够正常运行，Ceph 出现警告，但能够正常调用 Ceph 存储集群。修复错误，Ceph 警告消失，丢失节点快速加入 Ceph 集群并提供正常服务。

通过以上 4 个实验得出，此高可用性方案可以对单点故障及服务实例故障进行检测和处理，可自动切换各个服务节点，实现了 OpenStack 云平台的高可用服务。可以将服务实例资源在高可用节点之间进行快速切换，实现不间断地提供云平台服务，保证虚拟机实例稳定服务。

结束语 本文首先简要介绍了云计算和系统高可用性的相关概念以及 OpenStack 云计算管理平台的相关服务组件实现高可用性的方式。其次，结合现有的计算资源环境，提出一种基于 Pacemake＋Corosync＋HAProxy＋Ceph 的解决方案，实现 OpenStack 云计算管理平台的高可用性并实现实验云平台的建设。该方案将 Active-Active 的双活模式、Active-Passive 的主备模式及集群技术 3 种高可用设计模式融合在一起，通过软硬件冗余和服务实例故障转移方式等实现 OpenStack 云计算管理平台的高可用性。最后，利用现有的计算资源，结合该高可用性解决方案，构建 OpenStack 云计算管理平台。通过实践证明，该高可用性实验云平台方案具有有效性和可行性。

参 考 文 献

[1] 程宏兵，赵紫星，叶长河. 基于体系架构的云计算安全研究进展[J]. 计算机科学，2016，43(7)：19-27.

[2] 宋俊锋. 基于 MILP 的云计算数据中心扩张策略优化模型[J]. 湘潭大学自科学报，2015，37(4)：105-110.

[3] CORRADI A，FANELLI M，FOSCHINI L. VM consolidation：A real case based on OpenStack Cloud[J]. Future Generation Computer Systems，2014，32(1)：118-127.

[4] 张帆，李磊，杨成胡，等. 基于 Eucalyptus 构建私有云计算平台[J]. 电信科学，2011，27(11)：57-61.

[5] KESSACI Y，MELAB N，TALBI E G. A multi-start local search heuristic for an energy efficient VMs assignment on top of the OpenNebula cloud manager[J]. Future Generation Computer Systems，2014，36(3)：237-256.

[6] MILOJIČIĆ D，LLORENTE I M，MONTERO R S. OpenNebula：A Cloud Management Tool [J]. Internet Computing IEEE，2011，15(2)：11-14.

[7] 于飞. 基于 openNebula 云平台实验及性能评估[D]. 北京：北京邮电大学，2013

[8] PARADOWSKI A，LIU L，YUAN B. Benchmarking the Performance of OpenStack and CloudSack[C]// IEEE，International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing. IEEE，2014：405-412.

[9] 陈星，张颖，张晓东，等. 基于运行时模型的多样化云资源管理方法[J]. 软件学报，2014，25(7)：1476-1491.

[10] 马友忠，慈祥，孟小峰. 海量高维向量的并行 Top-k 连接查询[J]. 计算机学报，2015，38(1)：86-98.

[11] 马友忠，孟小峰. 云数据管理索引技术研究[J]. 软件学报，2015，26(1)：145-166.

[12] ANGLANO C，CANONICO M，GUAZZONE M. FC2Q：exploiting fuzzy control in server consolidation for cloud applications with SLA constraints[J]. Concurrency and Computation：Practice and Experience，2015，27(17)：910-915.

[13] TROPE R L，RESSLER E K. Mettle Fatigue：VW's Single-Point-of-Failure Ethics[J]. IEEE Security & Privacy Magazine，2016，14(1)：12-30.

[14] 杨祥. 无线传感器网络无标度容错拓扑的连锁故障诊断算法[J]. 计算机应用研究，2016，33(2)：549-551.