基于 Xen 虚拟化技术的弹性云架构①

李欣琪，吴 杰

(复旦大学 计算机科学技术学院，上海 200433)

摘 要：云计算通过虚拟化技术为用户提供基础架构即服务（IaaS），IaaS 平台上应用和服务的负载是动态变化的，这就导致其对虚拟资源的需求也是动态变化的。因此收集和分析云平台内部虚拟资源的占用量，根据需求对其进行弹性调度就成为提高整个云计算平台服务性能和资源利用率的关键。从负载均衡和降低云平台使用者成本的角度出发，根据云平台内部虚拟机的负载提出了一种弹性云架构。仿真实验结果表明，该方法能够提高虚拟资源的使用率以及降低用户的使用成本。

关键词：云计算；IaaS；虚拟机；负载均衡；弹性调度

Flexible Structure of Cloud Computing on Xen Virtualization Technology

LI Xin-Qi, WU Jie

(School of Computer Science, Fudan University, Shanghai 200433, China)

Abstract: Cloud Computing provides infrastructure as a service (IaaS) for users based on virtualization technology. The load of applications and services on IaaS platform is dynamic, thus its demand of virtual resource is also dynamic. Therefore, to collect and analyze the virtual resources footprint of in the platform, and flexibly scheduling them on demand has become the key point to improve overall service performance and resource utilization ratio of the cloud computing platform. This paper, with the concern of load balancing and reducing the user cost of the cloud platform, brings forward an elastic cloud structure based on the load of virtual machine in the platform. Simulation results show that the method can improve the utilization ratio of virtual resources and reduce user cost.

Key words: cloud computing; IaaS; virtual machine; load balance; flexibly scheduling

随着计算机技术的快速发展，移动网络和开放平台的急剧增长，云计算作为一种新兴的网络共享商业计算模型出现。云计算可以将计算任务分布在大量计算机构成的资源池上，使各种应用系统能够根据需要获取计算力、存储空间和各种软件服务。“云”中的这些资源在使用者看来是可以无限扩展的，并且可以随时获取，按需使用，随时扩展，按使用付费。云计算包含三种不同服务类型[1]: SaaS（Software as a Service，软件即服务）、PaaS（Platform as a Service，平台即服务）和 IaaS（Infrastructure as a Service，基础架构即服务），其中最重要也是最核心的技术就是 IaaS。

IaaS 指的是以服务形式为使用者提供服务器、存储和网络硬件，在 IaaS 服务中，资源是共享的，并根据用户的请求进行预留。但用户对资源的实际需求往往是不断变化的，如果资源预留的过多或过少都会造成资源的不合理分配；另外，云计算平台还需要根据应用和服务的实际负载，对用户请求的资源进行调度。

为了提供 IaaS 服务的基础架构，一般是利用虚拟化技术[2]搭建云平台的内部架构。虚拟化技术为云计算模型中的资源管理提供一种有效的解决办法。虚拟化技术可以在一台物理主机上划分并创建出不同的虚拟机，虚拟机之间相互隔离。通过将应用和服务封装在虚拟机中并根据负载的变化进行虚拟机和物理资源的调度来实现整个云平台的管理。

① 收稿时间:2011-03-09;收到修改稿时间:2011-04-18

在目前已有的云计算平台中，服务商提供了不同处理能力的虚拟机实例供用户选择。例如亚马逊的EC2 平台[3]，用户可以根据自身需求选择 Small、Large、Extra Large 等若干种不同级别的虚拟机实例，能够独立地对所租用虚拟机的状态进行启动、停止、关闭等操作。但对于初级用户而言，对于选择何种级别的虚拟机实例缺乏经验，选择过低或过高势必都造成云平台资源的浪费和用户使用成本的增加；而对于中级或高级用户来说，用户的实际需求也随着应用和服务的不断发展而迅速变化，他们需要对租用的云计算平台提供的虚拟机资源享有更高的灵活度和更多的控制权。这就导致了云计算平台下的虚拟化管理的两个问题：虚拟机所分配的资源一般都为预先定义，但应用程序信息的不确定性以及物理主机处理能力的差异性容易导致云计算环境中的负载失衡。同时，当应用程序对于资源的需求增加或减少时，还需要选择合适的物理主机对虚拟机进行迁移或释放不必要的资源。这对云计算平台的服务商而言可以将剩余资源回收并有效地分配给其他用户；对虚拟机的使用者而言可以避免占有不必要的资源而增加云的使用成本。有研究表明[4]，考虑虚拟机的放置策略能够有效协调不同物理主机的负载、维持高效的资源使用率。

因此，文章提出一种对用户和服务商而言更自动、可控、灵活的弹性云计算平台架构，能够根据应用程序和服务的负载使资源弹性分配，实现云计算环境中资源的合理优化和使用。该架构的主要优点包括：(1)能够实时监控云计算环境中资源的使用情况，根据应用和服务对资源的占用情况实现可伸缩性的动态虚拟机的调度；(2)为使用者提供更灵活的管理方式；(3)有效降低使用成本，能够对虚拟机所占用的资源量实时记录，计费方式更加合理、透明。

1　Xen虚拟化技术

1.1 Xen 虚拟化结构

弹性云计算平台的计算资源、存储资源、网络资源需要虚拟化技术的支持。虚拟化技术能够将不同结构的物理资源整合为逻辑资源池来供整个云计算平台使用。目前主流的开源虚拟化实现包括 Xen 和 KVM[5]。其中 Xen 可以工作在半虚拟化（Paravirtualization）和完全虚拟化两种模式之下。半虚拟化又叫做超虚拟化技术，该技术通过对客户操作系统做一些修改便可以在不支持虚拟化的硬件之上运行，无硬件依赖的特性使 Xen 的应用范围更加广泛。同时由于直接运行在硬件之上，虚拟机的性能更接近真实硬件环境，因此 Xen 更容易达到高性能。Amazon EC2、GoGridXen 和 Citrix 的云平台都采用 Xen 虚拟化技术，文章的弹性云架构也采用 Xen 虚拟化技术来实现。

Xen 的主要结构如下图所示：

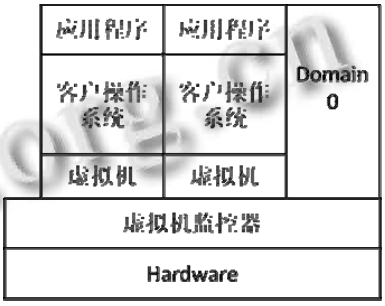

图 1　Xen 虚拟化结构

1.2 Xen Hypervisor

Xen Hypervisor 又叫做虚拟机监控器（VMM），实际上是一个软件层，介于硬件和操作系统之间，直接运行在机器硬件上。Xen Hypervisor 对硬件层进行虚拟，对系统中的所有虚拟机（VM）执行调度和分配资源，并且驱动硬件外设控制虚拟机的运行。而 Xen Hypervisor 的这些调度和分配工作对于客户操作系统而言是透明的。

1.3 Domain 0

Domain 0 是一个经过修改的 Linux 内核，也是唯一直接运行在 Xen Hypervisor 之上的虚拟机。Domain 0 在其它 Domain 启动前启动，其他虚拟机需要和 Domain 0 进行交互，通过 Domain 0 来和物理网络硬件通信、访问物理 I/O 资源等。

1.4 Domain U

在 Xen Hypervisor 之上运行的所有虚拟机都位于 Domain U，这些虚拟机获取到的只是虚拟的硬件资源。客户操作系统(Guest OS)安装在虚拟机（VM）上，并通过 Hypervisor 设定的特权等级运行在独立的地址空间以相互隔离。

2　系统架构与设计

下图描述了基于 Xen 虚拟化的弹性云整体架构，包括虚拟化集群、管理控制中心和用户三部分。用户

从客户端连接管理控制中心获取租用的虚拟机的运行状态，并可以对虚拟机进行启动、关闭等操作。管理控制中心将用户和云内部的虚拟化集群隔离开，所有的管理任务都由管理控制中心的相应模块执行。管理控制中心是整个弹性云平台的核心，云计算平台中所有的物理资源和虚拟资源的使用情况都在这里进行汇总，并按照用户预先订阅的条件，触发预设策略按需进行虚拟机动态调度，保证云平台的弹性。管理控制中心主要由监控模块、调度模块、日志模块、计费模块和用户模块五个部分组成。

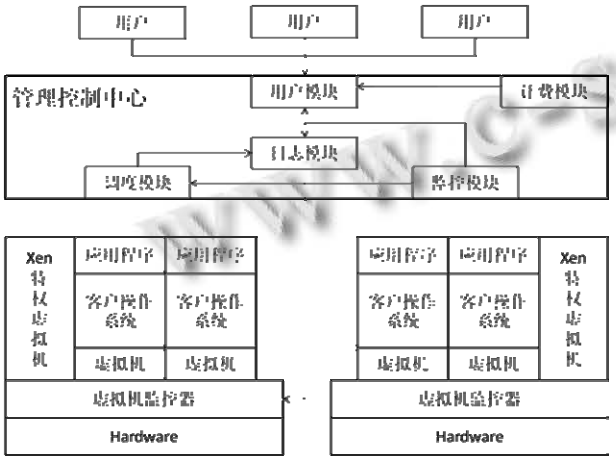

图 2 基于 Xen 虚拟化技术的弹性云架构

2.1 监控模块

监控模块的主要作用是和位于云平台内部的虚拟机集群中的虚拟机监控器进行交互，获得物理主机和虚拟机对于 CPU、内存和网络的使用情况，并将数据汇总至管理控制中心以供其它模块使用。

监控模块包括两部分，即位于每台物理主机上的服务端和位于管理控制中心的客户端。其中监控模块的服务端负责监控物理主机和运行在它之上的虚拟机的资源使用情况，监控模块的客户端定期向服务端发出请求对这些数据进行汇总，这些功能主要通过 Libvirt 库[6]实现。Libvirt 是提供虚拟化管理的一套 API 集合，它支持包括 Xen 、KVM 在内的多种虚拟机监控程序。监控模块通过 Libvirt 主要实现了以下功能：对虚拟机进行包括启动、停止、暂停、保存、恢复及迁移在内的操作；获取物理主机和虚拟机的运行状态和资源使用信息。

通过 Libvirt 进行监控的方式如下图所示：

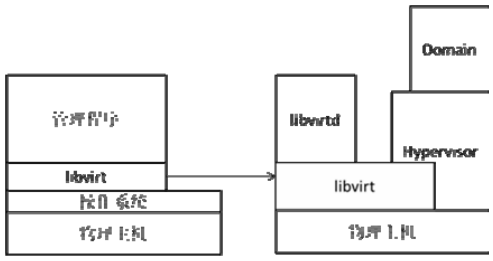

图 3 利用 Libvirt 监控示意图

在需要监控的虚拟机集群中的每台物理主机上安装 Libvirt daemon（libvirtd），Libvirt daemon 运行在虚拟机的 Domain 0 域，它负责收集物理主机及虚拟机对于 CPU、内存、网络等资源的使用情况，并与管理控制中心的 libvirt 模块通过 API 定义的通用协议进行远程通信来传递监控数据。为了获取每台物理主机和虚拟机的资源使用情况，定义了以下数据结构：

```
Struct DomainInfo
{
  unsigned char state;      //当前域的运行状态
  unsigned long maxMem; //支持的最大内存
  unsigned long memory;    //使用的内存
  unsigned short nrVirtCpu; //虚拟 CPU 数量
  unsigned long long cpuTime; //虚拟 CPU 运行时间
  unsigned long network; //网络带宽
}
```

每个 Domain 中虚拟机的资源占用情况获取流程如下所示：

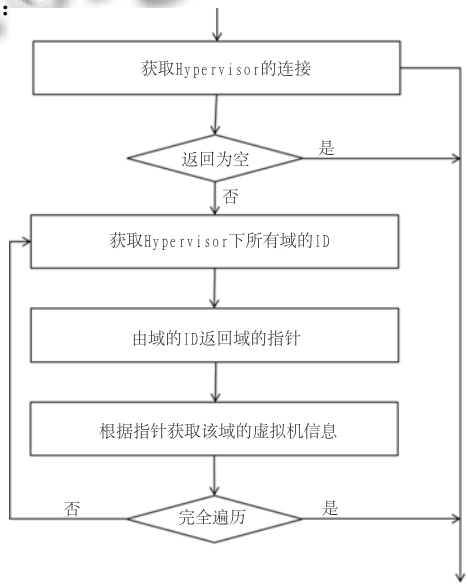

图 4 获取虚拟机资源占用流程

其中 Domain 0 反映的是物理主机的资源占用情况，其余 Domain 的 CPU 运行时间可以通过在某个时间段 Δt 内的两次 DomainInfo 中虚拟 CPU 运行时间的差来得到，若记 Δt 前后 Domain U 的 DomainInfo 分别为 D_a，D_b，那么虚拟 CPU 运行时间 VirtCPURuntime 可以表示为：

VirtCPURuntime = D_b. cpuTime – D_a. cpuTime

Domain U 的物理 CPU 的占用率 VirtCPUUsage 可以表示为：

$$VirtCPUUsage = \frac{Db.cpuTime - Da.cpuTime}{\Delta t}$$

2.2 调度模块

调度模块根据监控模块获取的资源使用情况，根据预先设定的策略，对虚拟机是否迁移以及迁移对象进行选择。考虑到云平台中虚拟机上运行的应用程序信息的差异，物理主机和虚拟机的负载有可能出现时高时低的跳跃，因此如果只根据某个时刻监控得到的峰值来进行资源重新分配或动态迁移则有可能造成整个虚拟机集群内的频繁迁移，增加不必要的开销，无法达到负载均衡的目的。为了避免这种情况，做出以下改进：当出现监控得到的负载值超过阈值时，进一步观察接下来的 N 个监控值，当出现多于 M 个值超过阈值时再进行资源重新分配或动态迁移。其中 N 的大小由云平台自身决定，M 的大小以 M/N 的比例形式由用户通过客户端预先给出，用户也可以选择将调度策略托管给云平台来自动完成。如果用户选择的比例值越接近 0，则说明用户采用较为激进的调度策略，在出现峰值时就进行虚拟机调度；反之，如果用户选择的比例值越接近 100%，则说明用户采用较为保守的调度策略，在出现若干个峰值后才进行虚拟机调度。

在做出虚拟机调度的决策之后，调度模块首先根据虚拟机所在物理主机上资源的使用情况判断能否为虚拟机重新分配相应的资源，通过这种方式能够减少虚拟机迁移所带来网络开销。如果所在物理主机无法满足虚拟机的资源请求，则需要进行虚拟机动态迁移。

迁移目的主机的选取考虑两方面的因素：响应速度均衡和处理能力均衡。根据监控模块获取的集群内各物理主机对 ping 请求的响应时间，以及物理主机 CPU、内存的使用量进行加权。选择权重值最高的物理主机作为目的主机。这种算法能较好地反映各物理主机的运行状态，当触发虚拟机迁移后，权重值最大的物理主机相应的网络负载与可提供的处理能力都会有相应的下降，从而权重值也会变化。在这个过程中集群中处理能力与网络负载次之的主机也会逐渐被选为接受虚拟机迁移的对象，相应的资源占用率也将提高，在整体上而言集群的负载也达到均衡。

2.3 计费模块

计费模块根据监控模块获取的资源使用数据，统计不同用户对于各物理资源的占用情况。按照预先设定的资源使用价格以及用户预先指定的阈值和调度策略计算出各用户的使用费用，并将数据发送到用户模块。若每单位计算、存储和网络的费用为 C_i（i=1,2,3），用户 k 对于物理主机 j（j=1,2,3,…,n）的使用量为 X_{kij}，用户为指定的阈值下界 U_1、上界 U_2 和调度策略 P 应支付的相应费用为 f(U_1,U_2, P)（具体计费策略可由云平台预先定义），那么该用户应支付的总费用 C_k 为：

$$C_k = \sum_{i=1}^{3} \sum_{j=1}^{n} C_i * + f(U_1, U_2, P)$$

2.4 日志模块

日志模块主要根据监控模块获取的数据，记录云平台内各物理主机和虚拟机运行状态、资源使用情况以及用户对虚拟机进行的操作。

2.5 用户模块

用户模块包括服务端和客户端两部分。用户可以通过客户端向服务端发送请求，包括创建、启动、停止、暂停、保存和恢复虚拟机，对虚拟机调度的阈值及策略（保守或激进）进行订阅，同时可以获取已租用的虚拟机的运行状态及应支付的费用等；服务端收到关于虚拟机的操作请求后将其传递给监控模块处理，并将用户订阅的阈值及策略传递给调度中心，同时将虚拟机的运行状态及费用反馈给用户。

3 实验和分析

根据图 2 所描述的系统架构，使用 3 台 PC 搭建实验平台，其中一台作为管理控制中心安装各模块，另外两台 PC 作为物理主机，每台分别安装 Centos 5.3 操作系统，Xen 3.0 和虚拟机控制器，并在每台物理主机上创建 2 个虚拟机，安装 Centos 5.3 操作系统。每台 PC 的 CPU 为四核 2.13GHz，硬盘大小为 160G,转速 7200RPM，内存 2G，每台虚拟机分配两个 CPU，512M 内存和 20G 硬盘。本节对提出的弹性云架构的原型系统进行性能评测，主要测试系统在不同负载状况下的

调度情况和性能表现。实验首先测试虚拟机的调度触发条件，设定 N=10，M=7，CPU 的阈值下界为 15%，上界为 80%，测试结果如图 5 所示。可以看到在最初的时刻虽然虚拟机的 CPU 负载超过了 80%，但是还未达到调度策略的条件，因此并没有引发调度。在接下来的 10 个监测结果中出现 7 次以上超过负载的情况，因此引发对虚拟机的资源调度。在 t=26 这个时刻触发调度策略之后，虚拟机所在物理主机为其增加一个 CPU 单元，之后的虚拟机 CPU 的负载有较明显的下降。

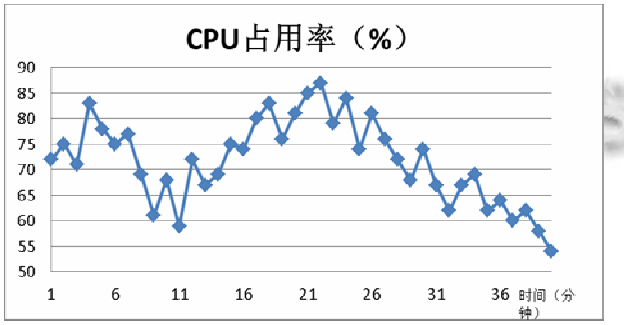

图 5 虚拟机调度监控图

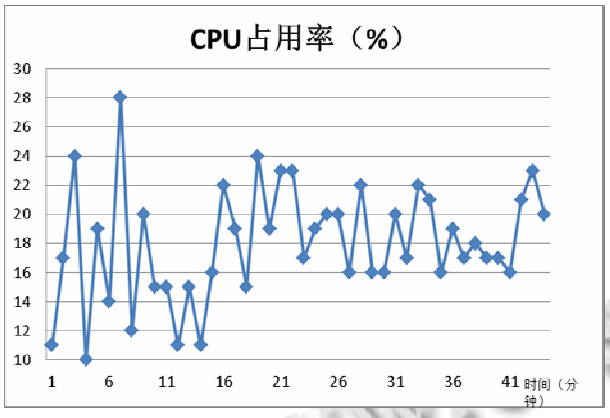

图 6 物理主机负载监控图

如图 6 所示，当虚拟机的负载有明显下降，达到阈值下界并触发调度策略时，将对虚拟机进行调度，减少其所占用的资源量。在 t=14 这个时刻，虚拟机的 CPU 占用率在最近的 10 次监控中超过 7 次低于阈值下界，从而引发调度，物理主机将减少该虚拟机的物理 CPU 资源，回收可用的 CPU 资源。在 t=14 之后的阶段，由于虚拟机所占有的物理资源减少，因此虚拟 CPU 的占用率有了一定的增加。

从图 5、图 6 可以看出，采用弹性架构的云平台原型系统，能够对虚拟机的资源请求做出精确的响应，有效地利用物理资源。当虚拟机进入繁忙状态时及时为其增加物理 CPU 资源，保证了高可用性；当虚拟机进入空闲状态时，降低物理 CPU 的预分配，从而可将回收的 CPU 资源分配给其他虚拟机，提高了物理资源的利用率。

4 结语

文章描述一种基于 Xen 虚拟化技术的弹性云架构，能够在一定程度上解决云计算平台中不同用户对租用虚拟机的弹性需求，适应应用和服务的不同负载，对其进行动态管理。实验结果表明，该架构能够较好地实现资源的动态分配，使应用和服务的负载得到均衡。

参考文献

1 Foster I, Zhao Y, Raicu I, Lu SY. Cloud Computing and Grid Computing 360-Degree Compared. Grid Computing Environments Workshop, 2008,3–4.

2 Barham P, Dragovic B, Fraser K, eds. Xen and the art of virtualization. SOSP’03 proceedings of the nineteenth ACM symposium on Operating systems principles, 2003, 164–177.

3 Amazon EC2. http://aws.amazon.com/ec2/.

4 Meng XQ, Pappas V, Zhang L. Improving the Scalability of Data Center Networks with Traffic-aware Virtual Machine Placement. INFOCOM, 2010,1–2.

5 Deshane T, Shepherd Z, Matthews JN, eds. Quantitative Comparison of Xen and KVM. Xen Summit, 2008, June (23-24): 1–2.

6 Bolte M, Sievers M, Birkenheuer G, eds. Non-intrusive Virtualization Management using libvirt. Design, Automation & Test in Europe Conference & Exhibition (DATE), 2010, 574–579.