

面向 IaaS 的云计算系统可用性评估

李阿妮¹ 张 晓^{1,2} 赵晓南¹ 张伯阳¹ 柳春懿¹

(西北工业大学计算机学院 西安 710129)¹

(高性能服务器和存储技术国家重点实验室 济南 250101)²

摘 要 随着云计算应用范围的扩展,人们对云计算在安全性、可靠性、可用性等方面的担忧与日俱增。云计算服务商承诺提供的服务满足服务水平协议,主要包括性能和可用性。但是,因可用性的内在复杂性而缺乏定量评估手段,对于云计算系统用户而言更是如此。为此,提出面向最终用户和面向云计算服务提供商的云计算系统可用性评估方法,这两种方法都采用可用性的定义建立模型,并且采用虚拟机的启动时间代替平均故障修复时间(Mean Time To Repair, MTTR)。第一种方法的平均失效前时间(Mean Time To Failure, MTTF)采用云计算服务提供商提供的参数;第二种方法通过统计得到虚拟机的 MTTF,并计算云计算系统之上的集群可用性。由于难以对后者进行实验验证,因此只对第一种方法进行验证,通过实验测量公有云和私有云提供的虚拟机的启动时间,结合云计算服务提供商提供的 MTTF,计算了两种不同云平台服务的可用性,该方法方便用户快速计算可用性,为用户选择不同的云服务提供建议,并且可以判断云计算服务提供商的承诺是否达到标准以及是否可以将一些关键行业应用服务迁移至云平台。

关键词 云计算, 可用性, 平均失效前时间, 平均故障修复时间, 定量计算

中图分类号 TP301 文献标识码 A DOI 10.11896/j.issn.1002-137X.2016.10.005

Cloud Computing System Availability Evaluation for IaaS

LI A-ni¹ ZHANG Xiao^{1,2} ZHAO Xiao-nan¹ ZHANG Bo-yang¹ LIU Chun-yi¹

(Department of Computer Science, Northwestern Polytechnical University, Xi'an 710129, China)¹

(State Key Laboratory of High-end Server and Storage Technology, Jinan 250101, China)²

Abstract With the expansion of the cloud computing application, people are getting worried about the security, reliability and availability of cloud computing. The cloud computing system promises to provide the services that meet the service level agreements which commonly include performance and availability. However, because of the inherent complexity of the availability, the evaluation of availability lacks quantitative calculation method, especially for the cloud computing system users. To solve these issues, we proposed an end-user oriented and a cloud computing service provider oriented cloud computing system availability evaluation method. These two methods adopt the definition of the availability to build model and take the boot time of the virtual machine as mean time to repair (MTTR). The mean time to failure (MTTF) of the first method adopts the parameter which is provided by cloud computing service providers. The virtual machine's MTTF of the second method is statistically obtained, meanwhile, the cluster availability of the cloud computing system is counted. Because it is difficult to carry out experimental verification for the latter, therefore, we only verified the first method. We measured the boot time of the virtual machine in public clouds and private clouds, and combined the MTTF which is provided by cloud computing service provider. Finally, we calculated the availability of two different kinds of cloud computing services. This method facilitates users to calculate the availability quickly and provides recommendations for users to choose different cloud services. In addition, users can judge whether the cloud computing provider's promise reaches the standard and whether some important industry application services can migrate to the cloud platform.

Keywords Cloud computing, Availability, Mean time to failure, Mean time to repair, Quantitative calculation

1 引言

云计算^[1]是一种新型的应用模式,云计算提供商管理大

量计算、存储资源,通过虚拟化等技术向用户提供计算、存储等资源。这些资源是按需分配、按量付费的。Infrastructure as a Service(IaaS)¹⁾即基础设施即服务,消费者使用处理、存

¹⁾ <http://zh.wikipedia.org/wiki/基础设施即服务>

到稿日期:2015-09-14 返修日期:2015-11-16 本文受国家 863 重大项目(2013AA01A215),国家自然科学基金面上项目(61472323),西北工业大学基础研究基金(3102015JSJ0009),高效能服务器和存储技术国家重点实验室开放基金项目(2014HSSA11)资助。

李阿妮(1992—),女,硕士生,主要研究方向为云存储,E-mail:liani_1028@163.com;张 晓(1978—),男,博士,副教授,硕士生导师,主要研究方向为云存储、海量存储,E-mail:zhangxiao@nwpu.edu.cn;赵晓南 女,博士,讲师,主要研究方向为海量存储,E-mail:zhaoxn@nwpu.edu.cn;张伯阳(1990—),男,硕士生,主要研究方向为云存储,E-mail:zhangboyang@mail.nwpu.edu.cn;柳春懿 男,硕士生,主要研究方向为云存储,E-mail:18729580703@163.com。

储、网络以及各种基础运算资源,部署与执行操作系统或应用程序等各种软件。客户端无须购买服务器、软件等网络设备,即可任意部署和运行处理、存储、网络和其它基本的计算资源,不能控制底层的基础设施,但是可以控制操作系统、存储装置和已部署的应用程序。

随着云计算平台的广泛应用,越来越多的企业和个人将数据迁移至云计算平台,相应地,对云计算平台的可用性也提出质疑。UCBerkeley 的 Michael Armbrust 等人提出的云计算必须克服的十大障碍中,保证服务的可用位列第一^[2]。许多著名的云计算数据中心也曾发生过重大故障,造成巨大损失,如^[1]:2012年2月29日和7月26日,微软的 Azure 发生事故,时间分别长达9个小时和2.5个小时,许多北美和欧洲的用户无法正常管理和使用其公司正常业务,有的彻底丢失了他们的最新数据。在2012年6月14日、6月29日、10月22日和圣诞节期间的12月24日,亚马逊 AWS 发生了因云服务缓慢和崩溃而无法访问的严重问题,影响的租户包括许多重要的互联网公司,例如 Netflix, pInterest, twitter, Instagram 等,每次事故导致用户无法正常使用服务的时间长达9个小时以上。2013年8月19日亚马逊网站发生故障宕机事故,波及北美等全球多地用户,持续40分钟左右才全面恢复,媒体预测,此次事故导致亚马逊至少损失472万美元^[2]。

云计算提供商声称其服务具有高可用性,但是用户无法对云计算系统的可用性进行评估。云计算系统常常由成千上万个廉价服务器构成,其中软硬件故障频繁发生。传统的集群系统可用性评估方法依次评估物理节点的故障率,然后根据各个节点在系统中所处的位置建立可用性分析模型,最后计算出系统整体的可用性。系统可用性与故障发生频率、容错机制和故障恢复时间等因素相关。

虚拟化技术为用户提供了统一的操作界面,屏蔽了底层的硬件差异和资源分布情况。由于计算资源的使用者和所有者分离,用户无法得知底层硬件的构成、故障率及恢复时间等参数,因此无法使用传统的方法评估云计算系统的可用性。

针对该问题,本文提出了面向最终用户和面向云计算服务提供商的云计算系统可用性评估方法。这两种方法通过测量虚拟机启动时间来评估云计算系统可用性,采用可用性的定义建立模型。用户通过使用面向最终用户的云计算系统可用性评估方法,可快速评估云计算服务的可用性,评估云计算服务提供商是否达到承诺标准及选择云计算服务提供商,以及评估将关键应用服务迁移至云平台的可行性。系统的可用性是通过 MTTF 和 MTTR 计算得到的。

本文第2节主要介绍在云计算可用性研究方面前人的一些相关研究;第3节详细介绍两种云计算系统可用性评估方法及虚拟机启动时间的度量方法;第4节是云计算系统可用性评估方法验证,通过实验对第3节提出的面向最终用户的云计算系统可用性评估模型进行验证;最后总结全文。

2 相关研究

文献[3]采用两层方法构建虚拟化系统和非虚拟化系统可用性模型,其中上层使用故障树,下层使用齐次连续时间的

马尔科夫链(CTMC)。该模型的优点在于不仅考虑了硬件故障,如 CPU、内存等,而且考虑了软件故障,如虚拟机监视器、虚拟机、应用程序故障等。另外其还在虚拟化系统中提供高可用性服务和虚拟机动态迁移。但是该模型的缺点是根据作者提供的模型和参数,从用户角度很难具体计算可用性,不能为用户选择云服务提供参考意见。文献[4]提出基于马尔科夫链的分层可靠性模型(HAM)。该模型不仅提高了整个数据中心的可用性,而且从成本角度增加了基础设施的可扩展性。但是与本文模型对比,很明显该模型是关于可用性的分析模型,而本文提出的模型可以分别从用户角度和云提供商角度评价云计算系统的可用性。文献[5]所提模型的优点是综合考虑了3个层次之间的可用性。该模型的缺点在于没有给出 IaaS 层云计算系统可用性的具体评估方法。文献[6]通过分析虚拟集群结点的基础架构模式特点和生存周期存活状态,给出了3种虚拟集群结点的状态转移图,并基于马尔科夫链建立了相应的可用性分析模型。该模型的优点在于通过定义各种状态之间的转移概率和虚拟集群结点处于各种状态的时间比例,能够很方便地评价具有各种可用性参数数据的虚拟集群结点的可用性;但是局限性在于其主要关注的是单台虚拟机/单台物理机模式、两台虚拟机/单台物理机模式、两台虚拟机/两台物理机模式这3种模式下的虚拟机可用性,没有研究基于整个云计算系统的可用性。文献[7,8]从云提供商的角度,提出一种基于马尔科夫链的可用性模型,将存储池分成不同类型的资源池,物理机发生故障时在不同资源池迁移,最终根据热资源池的物理机数量评价系统的可用性。该模型的缺点在于只能从云提供商的角度分析云计算系统可用性,用户完全不知云服务的底层实现细节,所以不能使用该模型对云计算系统的可用性进行评价。文献[9]提出了一种针对云服务的端到端的性能分析的一般分析模型。然而,该模型局限于每个请求只有单个任务到达及关机状态的物理机的启动延迟没有解决。同样,虚拟化对性能的影响也没有解决。文献[10]使用交互的随机模型针对云计算中心建立性能和可用性模型。文中量化了大任务到达、任务服务时间、大任务规模和物理机故障等因素的影响。文献[11]基于马尔科夫链的两个主机系统和多个主机系统模型,建立软硬件系统的可用性分析模型。该模型主要评价当系统软硬件故障、系统在不同状态之间转换时的系统可用性。文献[12]从多个角度研究云计算可用性,通过负载均衡、数据复制、自我恢复和一致性机制等来提高云计算系统的可用性。

针对以上研究工作的不足,本文提出面向最终用户和面向云计算服务提供商的云计算系统可用性评估方法。该方法的主要创新在于:1)前人的研究主要是从云提供商的角度进行可用性评估,本文提供两种评估方法,既可以从云计算服务提供商角度进行评估,也可以从用户角度进行评估;2)前人的研究工作大多关注如何提高云平台可用性或者使用不同方法对可用性进行分析,本文提出一种定量计算模型,对云计算系统可用性提供更加方便直观的评估方法;3)用户通过本文提出的面向最终用户的可用性评估方法可以快速计算云平台可用性;4)用户可以通过面向最终用户的云平台可用性评估方

^[1] <http://www.crn.com/slide-shows/cloud/240144284/the-10-biggest-cloud-outages-of-2012.htm>

^[2] <http://www.techdaily.com.my/2013/0821/5793.html>

法评估云计算服务承诺商是否达到承诺标准,为用户选择云服务提供商提供参考意见;5)根据可用性的评估结果和行业标准,可以决策将哪些关键行业应用服务迁移至云平台。

3 云计算系统可用性评估

根据可用性的定义¹⁾可知,可用性可以通过如下公式计算得到:

可用性 = $\frac{MTTF}{MTTF + MTTR}$ (1)

云计算提供商声称可以提供高可用性、高性能、低成本的服务,但是用户无法对这些服务的可用性进行评估。因此本文针对该问题提出面向最终用户的云计算系统可用性评估方法,MTTF 采用云服务提供商提供的参数,MTTR 用虚拟机的启动时间代替。

因为服务器发生故障的种类和频率随机,用户无法得知,所以本文提出一种面向云计算服务提供商的云计算系统可用性评估方法,服务器的 MTTF 一般通过长时间连续运行,收集同类服务器故障发生的种类和频率并计算得出。而云计算服务提供的虚拟机运行在异构分布式环境下,使用长时间采集的故障有以下几个问题:1)虚拟化技术屏蔽了底层的物理服务器的配置等信息;2)根据云计算平台的调度策略,为了性能优化或节能等,运行中的虚拟机可在不同的服务器之间迁移。异构服务器故障率不同,但是虚拟机在不同服务器间迁移时用户无法感知。MTTR 同样用虚拟机的启动时间代替。

3.1 虚拟机启动时间度量

本文提出的两种云计算系统可用性评估方法采用虚拟机的启动时间来代替 MTTR。MTTR 描述产品由故障状态转为工作状态时修理时间的平均值。MTTR 也包含获得配件的时间、维修团队的响应时间、记录所有任务的时间及将设备重新投入使用的时间,即指系统修复一次故障所需要的时间。虚拟机发生故障修复时需要重新申请资源,再进行修复直到正常启动。而虚拟机的启动时间也包括用户请求资源的时间、资源分配时间、正常启动时间,所以本文采用虚拟机启动时间近似代替 MTTR。

云计算以虚拟实例的形式提供云服务,虚拟机在云环境下的启动与物理机的启动过程有差别。物理机的启动过程包括 BIOS,Boot Loader,载入内核和初始化 4 个步骤。而虚拟机在用户提出申请前其各种资源并没有预先分配,所以虚拟机启动过程还包括资源的分配。

图 1 示出了 Linux 物理机的启动过程。物理机的启动过程主要分为 5 个阶段²⁾:1)加载 BIOS。当打开计算机电源后,计算机机会首先加载 BIOS 信息,BIOS 中包含 CPU 的相关信息、设备启动顺序、硬盘和内存信息等。2)读取 MBR。MBR 即主引导记录(Master Boot Record),大小为 512 字节,系统找到 BIOS 所指定的硬盘的 MBR 后,就会将其复制到 0x7c00 地址所在的物理内存中。3)Boot Loader。Boot Loader 就是操作内核运行之前运行的一小段程序。通过这段小程序,可以初始化硬件设备,建立内存空间的映射图,从而将系统的软硬件环境带到一个合适的状态,常见的 Boot Loader 有 Grub, Lilo 和 spfdisk。4)加载内核。根据 grub 设定的内核映像所

在的路径,系统读取内存映像,并进行解压缩操作。5)初始化。系统将解压缩的内核放置在内存中,调用 start_kernel() 函数启动一系列的初始化函数并初始化各种设备,完成 Linux 核心环境的建立。

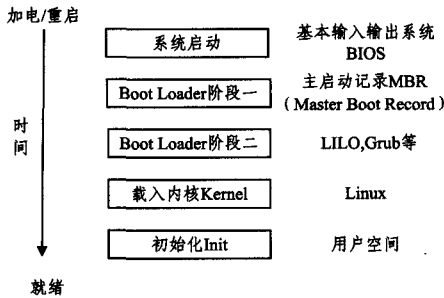


图 1 物理机的启动过程

虚拟机在用户提出申请前其各种资源并没有预先分配,所以虚拟机启动过程还包括资源的分配。而资源的分配需要云计算平台多个模块协同工作,根据平台不同启动时间也不同。以 Openstack 为例,在 Openstack Nova 中启动一个实例需要 API 服务、云控制器、调度、计算服务以及网络控制器等多个组件共同来完成。如图 2、图 3 所示,API 服务接收一个 run_instance 的命令请求并将其转发给云控制器,在云控制器中主要进行权限的认证等工作,认证通过后云控制器将接收到的命令发送给调度器,调度器会遵循一定的调度策略,指定一个计算节点来启动一个实例;该计算服务节点主要用来管理实例的启动和终止等事宜,此时虚拟机的启动还需要一个固定 IP 方可正常启动,因此计算服务组件将发一条消息给网络控制器,请求分配一个固定 IP,在这些条件具备的情况下,在计算服务组件内再执行传统系统的启动所需的过程,最终完成虚拟机的启动。

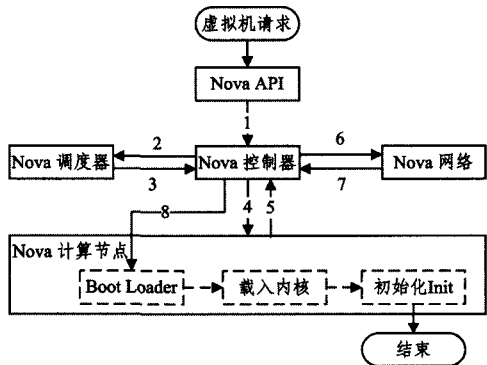


图 2 虚拟机的启动过程

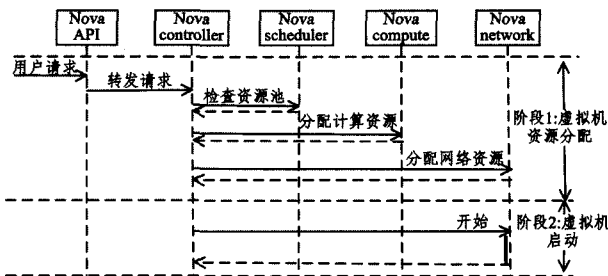


图 3 Openstack 中虚拟机的启动过程程序列图

¹⁾ <https://en.wikipedia.org/wiki/Availability>
²⁾ <http://www.cnblogs.com/QLinux/articles/2540304.html>

3.2 面向最终用户的云计算系统可用性评估方法

服务器生产商一般在其产品说明书中提供服务器的平均故障间隔时间。服务器 MTTF 通常的时间为 5 万小时至 30 万小时¹⁾。单台物理机故障会导致其所有虚拟机无法使用,所以单台虚拟机的 MTTF 与物理机虚拟化的程度有关,即与单台物理机上同时运行的虚拟机个数有关。

假设云计算平台中平均每台物理机运行 k 个虚拟机。虚拟机的 MTTF 为物理机的 MTTF 的 $\frac{1}{k}$ 倍,即:

$$MTTF_{vm} = \frac{1}{k} * MTTF_{pm} \quad (2)$$

其中, $MTTF_{pm}$ 为物理服务器的 MTTF, $MTTF_{vm}$ 为虚拟机的 MTTF, k 为单台物理机上平均启动的虚拟机数量。则单台虚拟机的可用性计算如下:

$$A = \frac{MTTF_{vm}}{MTTF_{vm} + MTTR_{vm}} \quad (3)$$

3.3 面向云计算服务提供商的云计算系统可用性评估方法

面向最终用户的云计算系统可用性评估方法基于假设集群中每台物理机上运行的虚拟机个数相同。但是,现实中为了保持负载均衡,集群中的虚拟机是动态迁移的,所以上述面向最终用户的评估方法不能很好地估计云计算系统的可用性。因此本文提出一种基于长时间观察服务器的故障种类和频率,再统计计算的面向云服务器提供商的云计算系统可用性评估模型。

如图 4 所示, MTTF 的定义为:

$$MTTF = \frac{\sum (downtime - uptime)}{failures \text{ times}} \quad (4)$$

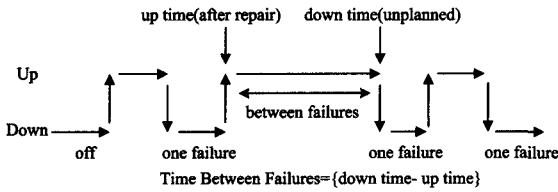


图 4 平均失效前时间的定义

失效时间是指从上一次设备恢复正常状态(uptime)起到设备此次失效那一刻(downtime)之间间隔的时间。

根据 MTTF 的定义,按如下统计计算 MTTF 的模型。

假设物理机的 MTTF 的数据采集矩阵为:

$$PM = \begin{bmatrix} n_{11} & n_{12} & \cdots & n_{1m} \\ n_{21} & n_{22} & \cdots & n_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ n_{n1} & n_{n2} & \cdots & n_{nm} \end{bmatrix} \quad (5)$$

其中, n 表示云计算平台共有 n 台物理机; m 表示采集数据时共有 m 个时间间隔,即 $t_1, t_2, \dots, t_m$; n_{ij} 表示物理机 i 在 t_j 时间段共发生故障的次数。

假设每台物理机上不同时间段启动虚拟机的个数矩阵 N 为:

$$N = \begin{bmatrix} k_{11} & k_{12} & \cdots & k_{1m} \\ k_{21} & k_{22} & \cdots & k_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ k_{n1} & k_{n2} & \cdots & k_{nm} \end{bmatrix} \quad (6)$$

其中, k_{ij} 表示物理机 i 在 t_j 时间段内启动的虚拟机数。

则虚拟机的 MTTF 为:

$$VM = PM * N$$

$$= \begin{bmatrix} n_{11} & n_{12} & \cdots & n_{1m} \\ n_{21} & n_{22} & \cdots & n_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ n_{n1} & n_{n2} & \cdots & n_{nm} \end{bmatrix} * \begin{bmatrix} k_{11} & k_{12} & \cdots & k_{1m} \\ k_{21} & k_{22} & \cdots & k_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ k_{n1} & k_{n2} & \cdots & k_{nm} \end{bmatrix} = \begin{bmatrix} n_{11}k_{11} & n_{12}k_{12} & \cdots & n_{1m}k_{1m} \\ n_{21}k_{21} & n_{22}k_{22} & \cdots & n_{2m}k_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ n_{n1}k_{n1} & n_{n2}k_{n2} & \cdots & n_{nm}k_{nm} \end{bmatrix} \quad (7)$$

其中, $A * B$ 表示矩阵的点乘运算,同维矩阵点乘等于对应的元素相乘。

则根据 MTTF 的定义即式(4)可得:单个物理机的 MTTF 为:

$$MTTF_{P_i} = \frac{\sum_{k=1}^m t_k}{\sum_{j=1}^m n_{ij}} \quad (8)$$

其中, k 表示时间段编号,从 1 到 m ; j 表示物理机 i 在时间段 j 发生故障次数的编号,从 1 到 m ; $MTTF_{P_i}$ 表示物理机 i 的 MTTF。

物理机集群的 MTTF 为:

$$MTTF_P = \frac{\sum_{k=1}^m t_k}{\sum_{i=1}^n \sum_{j=1}^m n_{ij}} \quad (9)$$

其中, i 表示物理机编号; $MTTF_P$ 表示物理机集群的 MTTF。

虚拟机集群的 MTTF 为:

$$MTTF_V = \frac{\sum_{k=1}^m t_k}{\sum_{i=1}^n \sum_{j=1}^m n_{ij} k_{ij}} \quad (10)$$

其中, $MTTF_V$ 表示虚拟机集群的 MTTF。

由式(9)可得:

$$\sum_{k=1}^m t_k = \sum_{i=1}^n \sum_{j=1}^m n_{ij} * MTTF_P \quad (11)$$

将式(11)代入式(10)可得:

$$MTTF_V = \frac{\sum_{k=1}^m t_k}{\sum_{i=1}^n \sum_{j=1}^m n_{ij} k_{ij}} = \frac{\sum_{i=1}^n \sum_{j=1}^m n_{ij} * MTTF_P}{\sum_{i=1}^n \sum_{j=1}^m n_{ij} k_{ij}} = \frac{\sum_{i=1}^n \sum_{j=1}^m n_{ij}}{\sum_{i=1}^n \sum_{j=1}^m n_{ij} k_{ij}} * MTTF_P \quad (12)$$

因此,虚拟机的可用性计算如下:

$$A = \frac{MTTF_V}{MTTF_V + MTTR_V} \quad (13)$$

4 云计算系统可用性评估方法验证

第 3 节描述了云计算系统可用性评估的定量计算模型。本节将通过实验进行云计算系统可用性评估方法的验证。因为对面向云服务器提供商的可用性评估方法的验证存在以下限制:1)需要长期观察服务器集群的故障次数及不同时间段单台服务器上虚拟机的运行个数并且进行统计;2)观察的周期长,至少需要 1 年;3)如果采用实验仿真,需要人为定期注入故障进行观察统计,这破坏了可用性的原始环境,观察的数

¹⁾ <http://www.intel.com/support/cn/motherboards/server/sb/cs-020405.htm>

据肯定不符合实际,所以本文只是针对面向最终用户的云计算系统可用性评估方法进行验证,MTTF 使用服务器提供商提供的参数,MTTR 采用测量的虚拟启动时间代替。

本文分别计算私有云和公有云的可用性,为用户在选择云服务时提供简单的参考。私有云(private cloud)^[13]是将云基础设施与软硬件资源建立在防火墙内,以供机构或企业内各部门共享数据中心内的资源。公有云通常指第三方提供商为用户提供的能够使用的云,公有云一般可通过 Internet 使用,可能是免费或成本低廉的。这种云有许多实例,可在当今整个开放的公有网络中提供服务。

4.1 评估虚拟机启动时间

本测试分别在私有云和公有云平台上进行。私有云选择本实验室搭建的 Openstack¹⁾,公有云选择亚马逊云²⁾。实验时将 Openstack 提供的云服务和亚马逊提供的云服务进行对比,Openstack 和亚马逊平台提供的不同配置虚拟机的信息如表 1 和表 2 所列。

通过查阅资料,测量虚拟机启动时间的工具,内核层面的有:Oprofile,Bootprobe,initcall_debug, kernel Function Trace³⁾等;用户层面的有:Strace,Process trace-Time's quick hack, Linux trace toolkit,Bootchart⁴⁾等;还有本实验室开发的 BootTimeScript 工具等。

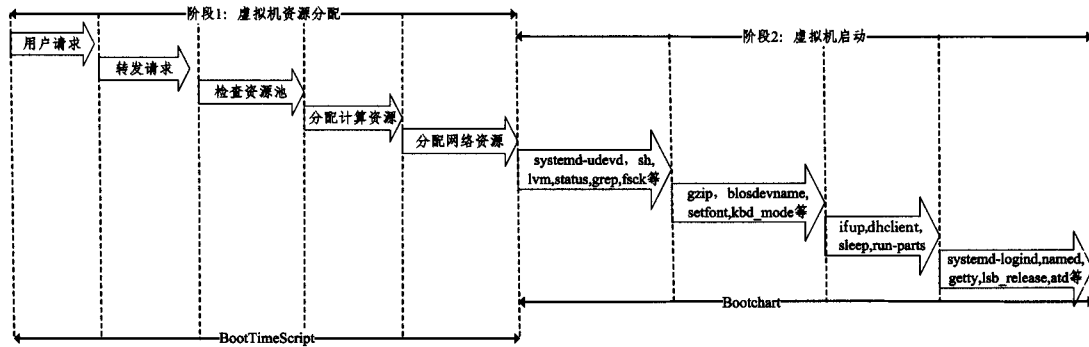


图 5 Bootchart 和 BootTimeScript 记录启动时间之间的关系

Openstack 平台提供的不同配置虚拟机的启动时间测量结果如表 3 所列。

表 3 不同配置 Openstack 虚拟机的启动时间

测试工具	Bootchart(s)				BootTimeScript(s)			
虚拟机名	osv1	osv2	osv3	osv4	osv1	osv2	osv3	osv4
测试 1	60.75	59.38	58.46	55.10	26	25	23	24
测试 2	59.59	61.77	59.81	54.48	24	26	25	25
测试 3	62.11	61.00	58.53	54.14	26	25	24	24
测试 4	58.96	59.13	58.50	53.77	24	24	24	23
测试 5	60.87	58.46	59.33	54.36	25	25	24	24
平均值	60.456	59.948	58.926	54.37	25	25	24	24

从表 3 中可以得到,随着虚拟机配置的提升,用 Bootchart 工具测得的虚拟机启动时间逐渐减少,用 BootTimeScript 工具测得的启动时间基本不变。从 Bootchart 测试的结果分析中可知,当 ifup 和 dhclient 服务启动之后虚拟机才会获得 IP,并且网卡准备好可以发送 ICMP 报文,从虚拟机

表 1 Openstack 云平台提供的虚拟机配置信息

虚拟机名	osv1	osv2	osv3	osv4
CPU	1 个单核	1 个单核	2 个单核	2 个单核
内存	994MB	2.0GB	3.9GB	7.8GB
操作系统	Ubuntu 14.04 LTS			
内核版本	3.13.0-24-generic			
处理器	Intel Core i7 9xx (Nehalem Class Core i7)			

表 2 亚马逊云平台提供的虚拟机配置信息

虚拟机名	aws1	aws2	aws3	aws4
CPU	1 个单核	1 个单核	2 个双核	2 个双核
内存	992MB	2.0GB	3.9GB	7.8GB
操作系统	Ubuntu 14.04.2 LTS			
内核版本	3.13.0-48-generic			
处理器	Intel(R) Xeon(R) CPU E5-2670 v2 @ 2.50GHz			

本验证实验使用 Bootchart 和 BootTimeScript 工具来测量虚拟机的启动时间。Bootchart 工具记录的启动时间为从虚拟机分配好资源开始到所有服务正常启动后之间的时间;而 BootTimeScript 工具记录的启动时间为从用户请求资源开始到网络服务启动后之间的时间,采用这两个工具测量虚拟机启动时间的关系,如图 5 所示。实验中用不同的方法测试 5 次,最终结果取其平均值。

准备启动到网络服务开启的平均时间为 11.5s,即 BootTimeScript 和 Bootchart 测量的重合时间为 11.5s。

亚马逊平台提供的不同配置虚拟机的启动时间测量结果如表 4 所列。

表 4 不同配置亚马逊虚拟机的启动时间

测试工具	Bootchart(s)				BootTimeScript(s)			
虚拟机名	aws1	aws2	aws3	aws4	aws1	aws2	aws3	aws4
测试 1	51.89	51.97	50.65	50.43	70	69	51	73
测试 2	52.05	51.88	50.74	50.27	64	59	56	60
测试 3	52.21	51.94	50.68	51.28	69	63	51	64
测试 4	51.95	51.96	51.05	50.44	70	69	73	75
测试 5	51.93	51.89	50.64	50.37	71	70	52	67
平均值	52.006	51.928	50.752	50.558	68.8	66	56.6	67.8

从表 4 可以得到,随着虚拟机配置提升,用 Bootchart 工具测量的虚拟机启动时间逐渐减少,用 BootTimeScript 工具测得的启动时间波动比较大,且无规律可寻,主要原因是

¹⁾ <http://www.openstack.org>

²⁾ <http://aws.amazon.com/cn>

³⁾ http://elinux.org/Boot_Time

⁴⁾ <http://www.bootchart.org>

BootTimeScript 会记录资源分配的时间,公有云某时刻的用户访问量是不定的,用户请求虚拟机时,先将请求放到队列中进行等待,所以虚拟机资源分配的时间就与队列中用户等待数相关,因此使用 BootTimeScript 工具测量的虚拟机启动时间波动比较大。分析 Bootchart 记录过程,可得网络服务开启的时间平均为 6.8s,资源分配时间波动范围为45~70s。

通过实验结果发现,私有云平台 Openstack 使用 Bootchart 测量的虚拟机启动时间比 BootTimeScript 测量的启动时间长,分析 Bootchart 记录的启动项可知,Bootchart 详细记录各个程序在启动过程中启动时间的占用情况,导致启动时间延长的主要进程有: init 进程,由内核启动的用户级进程,在操作系统加载完内核之后,会运行操作系统上的第一个进程 init; Bootchart 工具本身的 collector 进程; USB 的热插拔进程 upstart-udev-br; systemd-udev 进程,监听内核事件,对每个事件 systemd-udev 按照 udev 规则匹配指令; dhclient 进程,直接控制 eth 来进行网络操作获取 IP; 日志服务 rsyslogd; systemd-logind; 命名服务 named; 开启终端,进行终端初始化,用来设置终端的进程 getty; atd 进程,在一个指定的时间执行一个指定的任务等。而 BootTimeScript 除了进行资源分配外,只需要网络服务 ifup 和 dhclient 正常启动即可认为虚拟机已经正常启动。

公有云平台亚马逊使用 Bootchart 和 BootTimeScript 测量的虚拟机启动时间不能对比。因为使用 Bootchart 测量的虚拟机启动时间在 50~52s 之间,而 BootTimeScript 会记录资源分配时间,资源分配时间受某时刻亚马逊平台用户请求量的影响比较大,与队列中排队等待申请的用户数有关,资源分配时间在 45~70s 之间,测量的总虚拟机启动时间在 50~75s 之间。

通过实验结果知,虚拟机的启动时间与用户申请虚拟机的配置、搭建云平台服务器的性能、云平台的现有负载以及同一时刻云平台中申请虚拟机的用户数等因素有关。观察 Bootchart 工具测量私有云和公有云的虚拟机启动时间结果可知,当用户申请虚拟机的配置提升时,虚拟机启动时间随之减少。观察 BootTimeScript 工具的测量结果可知,公有云平台受用户请求数的影响比较大,虽然私有云平台测量结果显示虚拟机启动时间较少,但是当私有云平台负载过重并且某时刻同时请求虚拟机的用户较多时,虚拟机的启动时间过长,所以仅仅使用 BootTimeScript 工具测量的数据无法说明私有云和公有云平台哪个平台的可用性高。另外,表 4 中测量的私有云平台 and 公有云平台的虚拟机启动时间包括网络延迟时间的影响,通过测试可知本实验搭建的私有云平台 Openstack 虚拟机的平均网络延迟时间为 2ms,即 0.002s; 公有云平台亚马逊的平均网络延迟时间为 83ms,即 0.083s,因此网络延迟对公有云和私有云平台虚拟机启动时间的影响可以忽略。

因为 MTTR 就是当虚拟机发生故障崩溃后用户启动另一个虚拟机的时间,包括用户请求、申请资源、分配资源和启动时间。对比这两种测试工具,Bootchart 工具详细记录虚拟

机从资源分配完毕到正常启动的时间,不包含用户请求资源、分配资源的时间,而 BootTimeScript 工具记录的虚拟机启动时间,包括用户请求、申请资源、分配资源和启动时间。因此,在后文的计算时采用 BootTimeScript 工具测量的虚拟机启动时间代替 MTTR 更合理。

4.2 验证面向最终用户的云计算系统可用性评估方法

因为从用户角度采用面向云服务提供商的评估方法评估 MTTF 是不可行的,所以采用 MTTF 使用服务器提供商提供的参数的方法。

假设云平台单台服务器 MTTF 为 10 万小时,公有云平台亚马逊和本实验室搭建的私有云平台 Openstack 每台服务器分别平均运行 1000 台虚拟机,由式(2)可得:

$$MTTF_{Openstack} = \frac{1}{1000} \times 10 \text{ 万小时} = 100 \text{ 小时} \quad (14)$$

$$MTTF_{aws} = \frac{1}{1000} \times 10 \text{ 万小时} = 100 \text{ 小时}$$

由式(3)得到面向最终用户的云计算系统可用性评估模型为:

$$A = \frac{MTTF_{vm}}{MTTF_{vm} + MTTR_{vm}}$$

所以,Openstack 云平台的虚拟机可用性为:

$$A_{openstack-ovs1} = \frac{100h}{100h + 25s} = 99.99751\% \\ A_{openstack-ovs2} = \frac{100h}{100h + 25s} = 99.99751\% \quad (15)$$

$$A_{openstack-ovs3} = \frac{100h}{100h + 24s} = 99.99761\%$$

$$A_{openstack-ovs4} = \frac{100h}{100h + 24s} = 99.99761\%$$

亚马逊云平台的虚拟机可用性为:

$$A_{aws-aws1} = \frac{100h}{100h + 68.8s} = 99.99315\% \\ A_{aws-aws2} = \frac{100h}{100h + 66s} = 99.99342\% \quad (16) \\ A_{aws-aws3} = \frac{100h}{100h + 56.6s} = 99.99436\% \\ A_{aws-aws4} = \frac{100h}{100h + 67.8s} = 99.99324\%$$

根据上述两种云平台的优缺点分析及可用性计算结果,用户可以根据自己需求选择不同的云计算服务平台。

查阅资料得,Openstack 云平台的可用性可达 99.99%¹⁾, Amazon EC2 服务等级协议的承诺是为每个 Amazon EC2 区域提供 99.95% 的可用性²⁾。采用面向最终用户的评估方法评估 Openstack 平台和亚马逊云平台的可用性都超过 99.99%, 分别都高于 Openstack 云平台和亚马逊云平台的标准,因此可得 Openstack 云提供商和亚马逊云提供商的承诺达到标准。

根据云平台的可用性评估结果,可以选择将某些行业关键应用服务迁至云平台,从而同时保证可靠性和成本。例如,通过查阅资料可得银行服务的可用性必须达到 100%, 所以银行服务不能迁移至云平台³⁾; 沃尔玛电子商务选择将服务

¹⁾ <http://docs.openstack.org/high-availability-guide/content/ch-intro.html>

²⁾ <http://aws.amazon.com/cn/ec2>

³⁾ <http://www.infoq.com/cn/news/2015/02/wal-mart-choose-openstack>

迁移至 Openstack 云平台;邮件服务也可以迁移至云平台等。

结束语 通过与已有研究工作的对比,本文提出面向最终用户和面向云服务提供商的云计算系统可用性评估方法。这两种方法都采用可用性的定义建立模型,采用虚拟机的启动时间代替 MTTR。面向最终用户的云计算系统可用性评估方法中的 MTTF 采用云服务提供商提供的参数,而面向云计算服务提供商的云计算系统可用性评估方法采用连续长时间观察收集服务器的故障种类和频率,建立统计计算模型,计算得到 MTTF。由于面向云服务器提供商的云计算系统可用性评估方法实验验证的限制,本文只对面向最终用户的云计算系统可用性评估方法进行验证。本文通过实验对私有云平台 Openstack 和公有云平台亚马逊进行可用性评估,评估结果显示该模型适合用于快速评估云计算平台的可用性,同时为用户选择云服务提供了参考建议;实验结果表明云服务器提供商的承诺达到标准,可以将一些行业关键应用服务迁移至云平台,如沃尔玛的电子商务和邮件服务等。

参考文献

- [1] Luo Jun-zhou, Jin Jia-hui, Song Ai-bo, et al. Cloud computing: Architecture and Key Technology[J]. Journal on Communications, 2011, 32(7): 3-21(in Chinese)
罗军舟, 金嘉慧, 宋爱波, 等. 云计算: 体系架构与关键技术[J]. 通信学报, 2011, 32(7): 3-21
- [2] Armbrust M, Fox A, Griffith R, et al. A view of cloud computing [J]. Communications of the ACM, 2010, 53(4): 50-58
- [3] Kim D S, Machida F, Trivedi K S. Availability modeling and analysis of a virtualized system[C]//15th IEEE Pacific Rim International Symposium on Dependable Computing, 2009 (PRDC'09). IEEE, 2009: 365-371
- [4] Chuob S, Pokharel M, Park J S. Modeling and analysis of cloud computing availability based on eucalyptus platform for e-government data center[C]//2011 Fifth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS). IEEE, 2011: 289-296
- [5] Yang Zhi-ming, Zhang Jia-hui. A quantitative calculation model of the cloud computing availability[J]. Computer Software and Applications, 2014(7): 28-29(in Chinese)
杨志明, 张嘉慧. 一种云计算可用性定量计算模型[J]. 计算机软件和应用, 2014(7): 28-29
- [6] Che Jian-hua. Research on Performance and Availability Evaluation Methods of Virtualization Systems[D]. Hangzhou: Zhejiang University, 2010(in Chinese)
车建华. 虚拟计算系统性能和可用性评测方法研究[D]. 杭州: 浙江大学, 2010
- [7] Ghosh R, Longo F. Scalable Analytics for IaaS Cloud Availability[J]. IEEE Transactions on Cloud Computing, 2014, 2(1): 57-70
- [8] Longo F, Ghosh R, Naik V K, et al. A scalable availability model for infrastructure-as-a-service cloud[C]//2011 IEEE/IFIP 41st International Conference on Dependable Systems & Networks (DSN). IEEE, 2011: 335-346
- [9] Ghosh R, Trivedi K S, Naik V K, et al. End-to-end performability analysis for infrastructure-as-a-service cloud: An interacting stochastic models approach[C]//2010 IEEE 16th Pacific Rim International Symposium on Dependable Computing (PRDC). IEEE, 2010: 125-132
- [10] Khazaei H, Misic J, Misic V, et al. Availability analysis of cloud computing centers [C] // 2012 IEEE Global Communications Conference (GLOBECOM). IEEE, 2012: 1957-1962
- [11] Lai C D, Xie M, Poh K L, et al. A model for availability analysis of distributed software/hardware systems[J]. Information and Software Technology, 2002, 44(6): 343-350
- [12] Vani B, Priya R C M. A Survey on the Security Issues in Cloud Computing[J]. International Journal of P2P Network Trends and Technology, 2014(11): 16-19
- [13] Mell P, Grance T. The NIST definition of cloud computing[J]. Communication of the ACM, 2011, 53(6): 50
- [14] Bertot Y, Komendantskaya E. Inductive and Coinductive Components of Corecursive Functions in Coq[J]. Electronic Notes in Theoretical Computer Science, 2008, 203(5): 25-47
- [15] Skillicorn D B. Categorical Data Types[M]//Second Workshop on Abstract Models for Parallel Computation. Oxford University Press, 1992
- [16] Skillicorn D B. Questions and Answers about Categorical Data Types[R]. Technical Report, Queen's University, Computing and Information Science, 1994
- [17] Skillicorn D B, Cai W. A Cost Calculus for Parallel Functional Programming[J]. Journal of Parallel and Distributed Computing, 1995, 28(1): 65-83
- [18] Banger C R, Skillicorn D B. Constructing Categorical Data Types [R]. Research Reports, Canada: Queen's University, 1993
- [19] Skillicorn D B. Foundation of Parallel Programming[M]. Cambridge International Series on Parallel Computation, 1994
- [20] Orchard D A, Bolingbroke M, Mycroft A. Ypnos: Declarative, Parallel Structured Grid Programming [C] // Proc. of DAMP 2010. NY: ACM Press, 2010: 15-24
- [21] Kurz A, Petrisan D, Severi P. Nominal Coalgebraic Data Types With Applications to Lambda Calculus[J]. Logical Methods in Computer Science, 2013, 9(4): 1-51
- [22] Simon L, Mallya A, Bansal A, et al. Coinductive Logic Programming [M] // Etalle S, Truszczyński M. eds., ICLP 2006, LNCS 4079. Berlin: Springer-Verlag, 2006: 330-344
- [23] Simon L, Bansal A, Mallya A, et al. Co-Logic Programming: Extending Logic Programming with Coinduction [M] // ICALP 2007, LNCS 4596. Berlin: Springer-Verlag, 2007: 472-483
- [24] Hinze R, Jeuring J. Generic Haskell: Practice and Theory[M]//Generic Programming 2003, LNCS 2793. Berlin: Springer-Verlag, 2003: 1-56
- [25] Gibbons J. Datatypes-Generic Programming[M]//Spring School on Datatype-Generic Programming, LNCS 4719. Berlin: Springer-Verlag, 2007: 1-71
- [26] Ancona D, Lagorio G. Coinductive Type Systems for Object-Oriented Languages [M] // ECOOP 2009, LNCS 5653. Berlin: Springer-Verlag, 2009: 2-26

(上接第 18 页)

- [77] Kurz A, Petrisan D, Severi P. Nominal Coalgebraic Data Types With Applications to Lambda Calculus[J]. Logical Methods in Computer Science, 2013, 9(4): 1-51
- [78] Simon L, Mallya A, Bansal A, et al. Coinductive Logic Programming [M] // Etalle S, Truszczyński M. eds., ICLP 2006, LNCS 4079. Berlin: Springer-Verlag, 2006: 330-344
- [79] Simon L, Bansal A, Mallya A, et al. Co-Logic Programming: Extending Logic Programming with Coinduction [M] // ICALP 2007, LNCS 4596. Berlin: Springer-Verlag, 2007: 472-483
- [80] Hinze R, Jeuring J. Generic Haskell: Practice and Theory[M]//Generic Programming 2003, LNCS 2793. Berlin: Springer-Verlag, 2003: 1-56
- [81] Gibbons J. Datatypes-Generic Programming[M]//Spring School on Datatype-Generic Programming, LNCS 4719. Berlin: Springer-Verlag, 2007: 1-71
- [82] Ancona D, Lagorio G. Coinductive Type Systems for Object-Oriented Languages [M] // ECOOP 2009, LNCS 5653. Berlin: Springer-Verlag, 2009: 2-26

- [83] Bertot Y, Komendantskaya E. Inductive and Coinductive Components of Corecursive Functions in Coq[J]. Electronic Notes in Theoretical Computer Science, 2008, 203(5): 25-47
- [84] Skillicorn D B. Categorical Data Types[M]//Second Workshop on Abstract Models for Parallel Computation. Oxford University Press, 1992
- [85] Skillicorn D B. Questions and Answers about Categorical Data Types[R]. Technical Report, Queen's University, Computing and Information Science, 1994
- [86] Skillicorn D B, Cai W. A Cost Calculus for Parallel Functional Programming[J]. Journal of Parallel and Distributed Computing, 1995, 28(1): 65-83
- [87] Banger C R, Skillicorn D B. Constructing Categorical Data Types [R]. Research Reports, Canada: Queen's University, 1993
- [88] Skillicorn D B. Foundation of Parallel Programming[M]. Cambridge International Series on Parallel Computation, 1994
- [89] Orchard D A, Bolingbroke M, Mycroft A. Ypnos: Declarative, Parallel Structured Grid Programming [C] // Proc. of DAMP 2010. NY: ACM Press, 2010: 15-24