A view of cloud compting

> Cloud computing, the long-held dream of computing as a utility, has the potential to transform a large part of the IT industry, making software even more attractive as a service and shaping the way IT hardware is designed and purchased. Developers with innovative ideas for new Internet services no longer require the large capital outlays in hardware to deploy their service or the human expense to operate it. They need not be concerned about overprovisioning for a service whose popularity does not meet their predictions, thus wasting costly resources, or underprovisioning for one that becomes wildly popular, thus missing potential customers and revenue. Moreover, companies with large batch-oriented tasks can get results as quickly as their programs can scale, since using 1,000 servers for one hour costs no more than using one server for 1,000 hours. This elasticity of resources, without paying a premium for large scale, is unprecedented in the history of IT.

云计算是计算机作为一种实用工具的长期梦想，它有可能改变IT行业的大部分，使软件作为服务更具吸引力，并塑造IT硬件的设计和购买方式。对新互联网服务有创新想法的开发人员不再需要硬件上的大量资本支出来部署他们的服务或人力开支来运营它。他们不必担心过度配置的服务，其受欢迎程度不符合他们的预测，因此浪费了昂贵的资源，或者对于那些变得非常受欢迎的服务的供应不足，从而错过了潜在的客户和收入。此外，具有大批量任务的公司可以在程序可以扩展的同时尽快获得结果，因为使用1,000台服务器一小时的成本仅比使用一台服务器长达1,000小时。在没有大规模支付高价的情况下，这种资源弹性在IT历史上是前所未有的。

> As a result, cloud computing is a popular topic for blogging and white papers and has been featured in the title of workshops, conferences, and even magazines. Nevertheless, confusion remains about exactly what it is and when it’s useful, causing Oracle’s CEO Larry Ellison to vent his frustration: “The interesting thing about cloud computing is that we’ve redefined cloud computing to include everything that we already do.... I don’t understand what we would do differently in the light of cloud computing other than change the wording of some of our ads.” Our goal in this article is to reduce that confusion by clarifying terms, providing simple figures to quantify comparisons between of cloud and conventional computing, and identifying the top technical and non-technical obstacles and opportunities of cloud computing. (Armbrust et ${al}^{4}$ is a more detailed version of this article.)

因此，云计算是博客和白皮书的热门话题，并已在研讨会，会议甚至杂志的标题中出现。然而，混淆仍然是它究竟是什么以及什么时候它有用，导致甲骨文首席执行官拉里埃里森发泄他的沮丧：“云计算的有趣之处在于我们重新定义了云计算以包括我们已经做过的所有事情......除了改变我们的一些广告的措辞之外，我不明白我们会根据云计算做些什么。“我们在本文中的目标是通过澄清术语来减少这种混淆，提供简单的数字来量化之间的比较。云和传统计算，并确定云计算的顶级技术和非技术障碍和机会。（Armbrust et ${al}^{4}$ 是本文的更详细版本。）

Defining Cloud Computing

> Cloud computing refers to both the applications delivered as services over the Internet and the hardware and systems software in the data centers that provide those services. The services themselves have long been referred to as Software as a Service (SaaS). Some vendors use terms such as IaaS (Infrastructure as a Service) and PaaS (Platform as a Service) to describe their products, but we eschew these because accepted definitions for them still vary widely. The line between “low-level” infrastructure and a higher-level “platform” is not crisp. We believe the two are more alike than different, and we consider

> them together. Similarly, the related term “grid computing,” from the high-performance computing community, suggests protocols to offer shared computation and storage over long distances, but those protocols did not lead to a software environment that grew beyond its community.

云计算既指通过因特网作为服务提供的应用程序，也指提供这些服务的数据中心中的硬件和系统软件。服务本身长期以来被称为软件即服务（SaaS）。一些供应商使用诸如IaaS（基础架构即服务）和PaaS（平台即服务）之类的术语来描述他们的产品，但我们避开这些术语，因为它们的公认定义仍然存在很大差异。“低级”基础设施与更高级别“平台”之间的界限并不清晰。我们相信这两者更相似而不同，我们一起考虑它们。类似地，来自高性能计算社区的相关术语“网格计算”建议用于长距离提供共享计算和存储的协议，但是这些协议不会导致超出其社区的软件环境。

> The data center hardware and software is what we will call a cloud. When a cloud is made available in a pay-asyougo manner to the general public, we call it a public cloud; the service being sold is utility computing. We use the term private cloud to refer to internal data centers of a business or other organization, not made available to the general public, when they are large enough to benefit from the advantages of cloud computing that we discuss here. Thus, cloud computing is the sum of SaaS and utility computing,but does not include small or mediumsized data centers, even if these rely on virtualization for management. People can be users or providers of SaaS, or users or providers of utility computing.We focus on SaaS providers (cloud users) and cloud providers, which have received less attention than SaaS users.Figure 1 makes provider-user relationships clear. In some cases, the same actor can play multiple roles. For instance, a cloud provider might also host its own customer-facing services on cloud infrastructure.

数据中心硬件和软件就是我们所说的云。当云以一种付费方式向公众提供时，我们将其称为公共云; 正在销售的服务是效用计算。我们使用术语私有云来指代企业或其他组织的内部数据中心，当它们足够大以便从我们在此讨论的云计算的优势中受益时，不向公众提供。因此，云计算是SaaS和效用计算的总和，但不包括小型或中型数据中心，即使这些数据中心依赖虚拟化进行管理。人们可以是SaaS的用户或提供者，或者是公用计算的用户或提供者。我们专注于SaaS提供商（云用户）和云提供商，这些提供商受到的关注程度低于SaaS用户。图1使提供者 - 用户关系清晰。在某些情况下，同一个角色可以扮演多个角色。例如，云提供商也可能在云基础架构上托管自己面向客户的服务。

> From a hardware provisioning and pricing point of view, three aspects are new in cloud computing.
>
> The appearance of infinite computing resources available on demand, quickly enough to follow load surges, thereby eliminating the need for cloud computing users to plan far ahead for provisioning.
>
> The elimination of an up-front commitment by cloud users, thereby allowing companies to start small and increase hardware resources only when there is an increase in their needs.
>
> The ability to pay for use of computing resources on a short-term basis as needed (for example, processors by the hour and storage by the day) and release them as needed, thereby rewarding conservation by letting machines and storage go when they are no longer useful.

从硬件配置和定价的角度来看，云计算中有三个方面是新的。无限计算资源的出现可按需提供，足以快速跟踪负载激增，从而消除了云计算用户远程规划供应的需求。消除云用户的前期承诺，从而允许公司只有在需求增加时才能从小规模开始并增加硬件资源。能够根据需要在短期内支付使用计算资源的费用（例如，按小时处理的处理器和按天存储的费用）并根据需要发布它们，从而通过让机器和存储在不存在时进行保护来奖励保护 更长的用处。

> We argue that the construction and operation of extremely large-scale, commodity-computer data centers at low-cost locations was the key necessary enabler of cloud computing, for they uncovered the factors of 5 to 7 decrease in cost of electricity, network bandwidth, operations, software, and hardware available at these very large economies of scale. These factors, combined with statistical multiplexing to increase utilization compared to traditional data centers, meant that cloud computing could offer services below the costs of a medium-sized data center and yet still make a good profit.

我们认为，在低成本地区建设和运营极其大规模的商品计算机数据中心是云计算的关键必要推动因素，因为他们发现了电力成本，网络带宽，在这些非常大的规模经济中可用的操作，软件和硬件。与传统数据中心相比，这些因素与统计复用相结合，可提高利用率，这意味着云计算可以提供低于中型数据中心成本的服务，但仍能获得丰厚利润。

> Our proposed definition allows us to clearly identify certain installations as examples and non-examples of cloud computing. Consider a public-facing Internet service hosted on an ISP who can allocate more machines to the service given four hours notice. Since load surges on the public Internet can happen much more quickly than that (Animoto saw its load double every 12 hours for nearly three days), this is not cloud computing. In contrast, consider an internal enterprise data center whose applications are modified only with significant advance notice to administrators. In this scenario, large load surges on the scale of minutes are highly unlikely, so as long as allocation can track expected load increases, this scenario fulfills one of the necessary conditions for operating as a cloud. The enterprise data center may still fail to meet other conditions for being a cloud, however, such as the appearance of infinite resources or fine-grained billing. A private data center may also not benefit from the economies of scale that make public clouds financially attractive.

我们提出的定义允许我们清楚地将某些安装标识为云计算的示例和非示例。考虑一个托管在ISP上的面向公众的Internet服务，它可以在四小时通知的情况下为服务分配更多的机器。由于公共互联网上的负载激增可能比这更快（Animoto在近12天内每12小时看到其负载翻倍），这不是云计算。相比之下，考虑一个内部企业数据中心，其应用程序仅在向管理员提前通知的情况下进行修改。 在这种情况下，分钟级别的大量负载激增的可能性极小，因此只要分配可以跟踪预期的负载增加，此方案就可以满足作为云运行的必要条件之一。然而，企业数据中心可能仍然无法满足作为云的其他条件，例如无限资源的出现或细粒度计费。私人数据中心也可能无法从规模经济中受益，这使得公共云具有财务吸引力。

> Omitting private clouds from cloud computing has led to considerable debate in the blogosphere. We believe the confusion and skepticism illustrated by Larry Ellison's quote occurs when the advantages of public clouds are also claimed for medium-sized data centers. Except for extremely large data centers of hundreds of thousands of machines, such as those that might be operated by Google or Microsoft, most data centers enjoy only a subset of the potential advantages of public clouds, as Table 1 shows. We therefore believe that including traditional data centers in the definition of cloud computing will lead to exaggerated claims for smaller, so-called private clouds, which is why we exclude them. However, here we describe how so-called private clouds can get more of the benefits of public clouds through *surge computing or hybrid cloud computing*.

从云计算中省略私有云已引起博客圈的大量争论。我们认为拉里·埃里森引用的混淆和怀疑主义是在中型数据中心也声称公共云的优势时发生的。除了数十万台机器的超大型数据中心，例如可能由谷歌或微软运营的数据中心，大多数数据中心只享有公共云潜在优势的一部分，如表1所示。因此，我们认为将传统数据中心纳入云计算的定义将导致对较小的，所谓的私有云的夸大宣称，这就是我们排除它们的原因。但是，在这里我们描述了所谓的私有云如何通过浪涌计算或混合云计算获得公共云的更多好处。

Classes of Utility Computing

> Any application needs a model of computation, a model of storage, and a model of communication. The statistical multiplexing necessary to achieve elasticity and the appearance of infinite capacity available on demand requires automatic allocation and management. In practice, this is done with virtualization of some sort. Our view is that different utility computing offerings will be distinguished based on the cloud system software’s level of abstraction and the level of management of the resources.

任何应用程序都需要计算模型，存储模型和通信模型。实现弹性所需的统计多路复用和按需可用的无限容量的出现需要自动分配和管理。实际上，这是通过某种虚拟化来完成的。我们的观点是，将根据云系统软件的抽象级别和资源管理级别来区分不同的效用计算产品。

> Amazon EC2 is at one end of the spectrum. An EC2 instance looks much like physical hardware, and users can control nearly the entire software stack, from the kernel upward. This low level makes it inherently difficult for Amazon to offer automatic scalability and failover because the semantics associated with replication and other state management issues are highly application-dependent. At the other extreme of the spectrum are application domain-specific platforms such as Google AppEngine, which is targeted exclusively at traditional Web applications, enforcing an application structure of clean separation between a stateless computation tier and a stateful storage tier. AppEngine’s impressive automatic scaling and high-availability mechanisms, and the proprietary MegaStore data storage available to AppEngine applications, all rely on these constraints. Applications for Microsoft’s Azure are written using the .NET libraries, and compiled to the Common Language Runtime, a language-independent managed environment. The framework is significantly more flexible than AppEngine’s, but still constrains the user’s choice of storage model and application structure.Thus, Azure is intermediate between application frameworks like AppEngine and hardware virtual machines like EC2.

亚马逊EC2处于频谱的一端。EC2实例看起来很像物理硬件，用户可以从内核向上控制几乎整个软件堆栈。这种低级别使得亚马逊很难提供自动可扩展性和故障转移，因为与复制和其他状态管理问题相关的语义高度依赖于应用程序。另一方面，特定于应用程序领域的平台，例如Google AppEngine，专门针对传统的Web应用程序，在无状态计算层和有状态存储层之间实施清晰分离的应用程序结构。AppEngine令人印象深刻的自动扩展和高可用性机制，以及AppEngine应用程序可用的专有MegaStore数据存储都依赖于这些限制。Microsoft Azure的应用程序使用.NET库编写，并编译为Common Language Runtime，这是一种独立于语言的托管环境。该框架比AppEngine更灵活，但仍然限制了用户对存储模型和应用程序结构的选择。因此，Azure介于AppEngine等应用程序框架和EC2等硬件虚拟机之间。

Cloud Computing Economics

> We see three particularly compelling use cases that favor utility computing over conventional hosting. A first case is when demand for a service varies with time. For example, provisioning a data center for the peak load it must sustain a few days per month leads to underutilization at other times. Instead, cloud computing lets an organization pay by the hour for computing resources, potentially leading to cost savings even if the hourly rate to rent a machine from a cloud provider is higher than the rate to own one. A second case is when demand is unknown in advance. For example, a Web startup will need to support a spike in demand when it becomes popular, followed potentially by a reduction once some visitors turn away. Finally, organizations that perform batch analytics can use the “cost associativity” of cloud computing to finish computations faster: using 1,000 EC2 machines for one hour costs the same as using one machine for 1,000 hours.

我们看到三个特别引人注目的用例有利于公用计算而不是传统托管。第一种情况是对服务的需求随时间变化。例如，为峰值负载配置数据中心必须每月维持几天，导致其他时间利用不足。相反，云计算允许组织按小时计算计算资源，即使从云提供商租用计算机的小时费率高于拥有计算机的费率，也可能节省成本。第二种情况是提前需求未知。例如，一个网络创业公司需要在它变得流行时支持需求激增，之后可能会在一些访问者拒绝后减少。最后，执行批量分析的组织可以使用云计算的"成本关联性"来更快地完成计算：使用1,000台EC2机器一小时的成本与使用一台机器1000小时相同。

> Although the economic appeal of cloud computing is often described as “converting capital expenses to operating expenses” (CapEx to OpEx), we believe the phrase “pay as you go” more directly captures the economic benefit to the buyer. Hours purchased via cloud computing can be distributed non-uniformly in time (for example, use 100 server-hours today and no server-hours tomorrow, and still pay only for 100); in the networking community, this way of selling bandwidth is already known as usage-based pricing. In addition, the absence of up-front capital expense allows capital to be redirected to core business investment.

虽然云计算的经济吸引力通常被描述为"将资本支出转换为运营支出"（CapEx to OpEx），但我们认为"随用随付"这一短语更直接地抓住了买方的经济利益。通过云计算购买的小时数可以及时非均匀地分发（例如，今天使用100个服务器小时，明天不使用服务器小时，并且仍然仅支付100个）;在网络社区，这种销售带宽的方式已经被称为基于使用的定价。此外，缺乏前期资本支出可以将资金重定向到核心业务投资。

> Therefore, even if Amazon’s pay-as-you-go pricing was more expensive than buying and depreciating a comparable server over the same period, we argue that the cost is outweighed by the extremely important cloud computing economic benefits of elasticity and transference of risk, especially the risks of overprovisioning (underutilization) and underprovisioning (saturation).

因此，即使亚马逊的按需付费定价比同期购买和折旧同类服务器更昂贵，我们认为弹性和风险转移的极其重要的云计算经济效益抵消了成本，特别是过度供应（利用不足）和供应不足（饱和）的风险。

> We start with elasticity. The key observation is that cloud computing’s ability to add or remove resources at a fine grain (one server at a time with EC2) and with a lead time of minutes rather than weeks allows matching resources to workload much more closely. Real world estimates of average server utilization in data centers range from 5% to 20%. This may sound shockingly low, but it is consistent with the observation that for many services the peak workload exceeds the average by factors of 2 to 10. Since few users deliberately provision for less than the expected peak, resources are idle at nonpeak times. The more pronounced the variation, the more the waste.

我们从弹性角度开始。关键的观察结果是，云计算能够以细粒度（使用EC2一次一个服务器）添加或删除资源，并且提前几分钟而不是几周，这样可以更加紧密地将资源与工作负载相匹配。现实世界对数据中心平均服务器利用率的估计在5%到20%之间。这可能听起来令人震惊地低，但这与观察结果一致，即对于许多服务而言，峰值工作量超过平均值2到10倍。由于很少用户故意提供低于预期峰值的资源，因此资源在非峰值时间处于空闲状态。变化越明显，浪费越多。

> For example, Figure 2a assumes our service has a predictable demand where the peak requires 500 servers at noon but the trough requires only 100 servers at midnight. As long as the average utilization over a whole day is 300 servers, the actual cost per day (area under the curve) is 300 × 24 = 7,200 server hours; but since we must provision to the peak of 500 servers, we pay for 500 × 24 = 12,000 server-hours, a factor of 1.7 more. Therefore, as long as the pay-asyougo cost per server-hour over three years (typical amortization time) is less than 1.7 times the cost of buying the server, utility computing is cheaper.

例如，图2a假设我们的服务具有可预测的需求，其中峰值在中午需要500台服务器，但是低谷在午夜仅需要100台服务器。只要一整天的平均利用率为300台服务器，每天的实际成本（曲线下面积）为300×24 = 7,200服务器小时;但由于我们必须提供500台服务器的峰值，我们支付500×24 = 12,000服务器小时，相当于1.7倍。因此，只要三年（每个服务器小时）的付费使用成本（典型的摊还时间）小于购买服务器的成本的1.7倍，公用计算就更便宜。

> In fact, this example underestimates the benefits of elasticity, because in addition to simple diurnal patterns, most services also experience seasonal or other periodic demand variation (for example, e-commerce in December and photo sharing sites after holidays) as well as some unexpected demand bursts due to external events (for example, news events). Since it can take weeks to acquire and rack new equipment, to handle such spikes you must provision for them in advance. We already saw that even if service operators predict the spike sizes correctly, capacity is wasted, and if they overestimate the spike they provision for, it’s even worse.

事实上，这个例子低估了弹性的好处，因为除了简单的昼夜模式，大多数服务还会遇到季节性或其他周期性需求变化（例如，12月的电子商务和假期后的照片共享网站）以及一些意外的由于外部事件（例如，新闻事件）而导致需求突发。由于可能需要数周时间才能获得并装备新设备，因此必须提前为这些尖峰进行处理。我们已经看到，即使服务运营商正确地预测了峰值大小，也会浪费容量，如果他们高估了他们提供的峰值，那就更糟了。

> They may also underestimate the spike (Figure 2b), however, accidentally turning away excess users. While the cost of overprovisioning is easily measured, the cost of underprovisioning is more difficult to measure yet potentially equally serious: not only do rejected users generate zero revenue, they may never come back. For example, Friendster’s decline in popularity relative to competitors Facebook and MySpace is believed to have resulted partly from user dissatisfaction with slow response times (up to 40 seconds). Figure 2c aims to capture this behavior: Users will desert an underprovisioned service until the peak user load equals the data center’s usable capacity, at which point users again receive acceptable service.

他们也可能低估了尖峰（图2b），然而，意外地拒绝了多余的用户。虽然过度配置的成本很容易衡量，但是欠配置的成本更难以衡量，但可能同样严重：不仅被拒绝的用户产生零收入，他们可能永远不会回来。例如，相对于竞争对手Facebook和MySpace而言，Friendster的受欢迎程度下降被认为部分是由于用户对响应时间较慢（最多40秒）的不满.图2c旨在捕捉这种行为：用户将抛弃未充分利用的服务，直到峰值用户负载等于数据中心的可用容量，此时用户再次接收可接受的服务。

> For a simplified example, assume that users of a hypothetical site fall into two classes: active users (those who use the site regularly) and defectors (those who abandon the site or are turned away from the site due to poor performance).Further, suppose that 10% of active users who receive poor service due to underprovisioning are “permanently lost” opportunities (become defectors), that is, users who would have remained regular visitors with a better experience. The site is initially provisioned to handle an expected peak of 400,000 users (1,000 users per server × 400 servers), but unexpected positive press drives 500,000 users in the first hour. Of the 100,000 who are turned away or receive bad service, by our assumption 10,000 of them are permanently lost, leaving an active user base of 390,000. The next hour sees 250,000 new unique users. The first 10,000 do fine, but the site is still overcapacity by 240,000 users. This results in 24,000 additional defections, leaving 376,000 permanent users. If this pattern continues, after lg(500,000) or 19 hours, the number of new users will approach zero and the site will be at capacity in steady state. Clearly, the service operator has collected less than 400,000 users’ worth of steady revenue during those 19 hours, however, again illustrating the underutilization argument—to say nothing of the bad reputation from the disgruntled users.

举一个简单的例子，假设一个假设网站的用户分为两类：活跃用户（定期使用网站的用户）和叛逃者（放弃网站或由于性能不佳而拒绝网站的用户）。此外，假设由于资金不足而接受服务质量差的10%的活跃用户是“永久性失去”的机会（成为叛逃者），也就是那些本来可以保持常规访问者并获得更好体验的用户。该网站最初配置为处理400,000个用户的预期峰值（每个服务器1,000个用户×400个服务器），但意外的正压力在第一个小时内驱动500,000个用户。在被拒绝或接受不良服务的100,000人中，我们假设其中10,000人永久丢失，留下了390,000的活跃用户群。下一个小时将看到250,000个新的唯一身份用户。前10,000个很好，但该网站仍有240,000个用户的产能过剩。这导致24,000次额外叛逃，留下376,000名永久用户。如果此模式继续，在lg(500,000)或19小时之后，新用户的数量将接近零，并且站点将处于稳定状态的容量。显然，服务运营商在这19个小时内收集了不到40万用户的稳定收入，然而，再次说明了未充分利用的论点 —— 更不用说心怀不满的用户的不良声誉。

> Do such scenarios really occur in practice? When Animoto3 made its service available via Facebook, it experienced a demand surge that resulted in growing from 50 servers to 3,500 servers in three days. Even if the average utilization of each server was low, no one could have foreseen that resource needs would suddenly double every 12 hours for three days. After the peak subsided, traffic fell to a lower level. So in this real-world example, scale-up elasticity was not a cost optimization but an operational requirement, and scaledown elasticity allowed the steady-state expenditure to more closely match the steady-state workload.

这种情况真的发生在实践中吗？当Animoto3通过Facebook提供服务时，它经历了需求激增，导致在三天内从50台服务器增长到3,500台服务器。即使每台服务器的平均利用率很低，也没有人能够预见到资源需求会在12天内突然翻倍，持续3天。高峰消退后，交通量降至较低水平。因此，在这个现实世界的例子中，放大弹性不是成本优化而是操作要求，并且缩放弹性允许稳态支出更紧密地匹配稳态工作负荷。

Top 10 Obstacles and Opportunities for Cloud Computing

> Table 2 summarizes our ranked list of critical obstacles to growth of cloud computing. The first three affect adoption, the next five affect growth, and the last two are policy and business obstacles. Each obstacle is paired with an opportunity to overcome that obstacle, ranging from product development to research projects.

表2总结了我们对云计算增长的关键障碍的排名列表。前三个影响采用，后五个影响增长，后两个影响政策和业务障碍。每个障碍都与克服障碍的机会相结合，从产品开发到研究项目。

Number 1. Business Continuity and Service Availability

> Organizations worry about whether utility computing services will have adequate availability, and this makes some wary of cloud computing. Ironically, existing SaaS products have set a high standard in this regard. Google Search has a reputation for being highly available, to the point that even a small disruption is picked up by major news sources.

组织担心效用计算服务是否具有足够的可用性，这使得人们对云计算持谨慎态度。具有讽刺意味的是，现有的SaaS产品在这方面已经树立了很高的标准。谷歌搜索以其高可用性而闻名，即使主要新闻来源即使是一个小小的中断也是如此。

> Users expect similar availability from new services, which is difficult to do. Table 3 shows recorded outages for Amazon Simple Storage Service (S3), AppEngine and Gmail in 2008, and explanations for the outages. Note that despite the negative publicity due to these outages, few enterprise IT infrastructures are as good. Technical issues of availability aside, a cloud provider could suffer outages for nontechnical reasons, including going out of

> business or being the target of regulatory action (a recent example of the latter occurred last year, as we describe later).

用户希望新服务具有类似的可用性，这很难做到。表3显示了2008年亚马逊简单存储服务（S3），AppEngine和Gmail的记录中断，以及对中断的解释。请注意，尽管由于这些中断导致负面宣传，但很少有企业IT基础架构同样出色。除了可用性的技术问题之外，云提供商可能因非技术原因而遭受停机，包括停业或成为监管行动的目标（最近的一个例子发生在去年，如我们后面所述）。

> Although they have not done so, cloud vendors could offer specialized hardware and software techniques in order to deliver higher reliability, presumably at a high price. This reliability could then be sold to users as a service-level agreement. But this approach only goes so far. The high-availability computing community has long followed the mantra “no single point of failure,” yet the management of a cloud computing service by a single company is in fact a single point of failure. Even if the company has multiple data centers in different geographic regions using different network providers, it may have common software infrastructure and accounting systems, or the company may even go out of business. Large customers will be reluctant to migrate to cloud computing without a business-continuity strategy for such situations. We believe the best chance for independent software stacks is for them to be provided by different companies, as it has been difficult for one company to justify creating and maintain two stacks in the name of software dependability. Just as large Internet service providers use multiple network providers so that failure by a single company will not take them off the air, we believe the only plausible solution to very high availability is multiple cloud computing providers.

虽然他们还没有这样做，但云供应商可以提供专门的硬件和软件技术，以便提供更高的可靠性，大概是以高价格。然后，这种可靠性可以作为服务级别协议出售给用户。但这种方法只是到目前为止。高可用性计算社区长期以来一直遵循“没有单点故障”的口号，但单个公司对云计算服务的管理实际上是单点故障。即使公司在不同地理区域使用不同的网络提供商拥有多个数据中心，它也可能拥有通用的软件基础设施和会计系统，或者公司甚至可能会倒闭。如果没有针对此类情况的业务连续性策略，大客户将不愿意迁移到云计算。我们认为独立软件堆栈的最佳机会是由不同公司提供，因为一家公司很难以软件可靠性的名义创建和维护两个堆栈。正如大型互联网服务提供商使用多个网络提供商，以便单个公司的失败不会使他们无法播出，我们认为高可用性的唯一可行解决方案是多个云计算提供商。

Number 2. Data Lock-In

> Software stacks have improved interoperability among platforms, but the storage APIs for cloud computing are still essentially proprietary, or at least have not been the subject of active standardization. Thus, customers cannot easily extract their data and programs from one site to run on another. Concern about the difficulty of extracting data from the cloud is preventing some organizations from adopting cloud computing. Customer lock-in may be attractive to cloud computing providers, but their users are vulnerable to price increases, to reliability problems, or even to providers going out of business.

软件堆栈改善了平台之间的互操作性，但云计算的存储API仍然基本上是专有的，或者至少不是主动标准化的主题。因此，客户无法轻松地从一个站点提取其数据和程序以在另一个站点上运行。对从云中提取数据的难度的担忧阻碍了一些组织采用云计算。客户锁定可能对云计算提供商具有吸引力，但他们的用户很容易受到价格上涨，可靠性问题甚至服务提供商破产的影响。

> For example, an online storage service called The Linkup shut down on Aug. 8, 2008 after losing access as much as 45% of customer data. The Linkup, in turn, had relied on the online storage service Nirvanix to store customer data, which led to finger pointing between the two organizations as to why customer data was lost. Meanwhile, The Linkup’s 20,000 users were told the service was no longer available and were urged to try out another storage site.

例如，一个名为The Linkup的在线存储服务在失去了45%的客户数据后，于2008年8月8日关闭。反过来，Linkup依靠在线存储服务Nirvanix来存储客户数据，从而指导两个组织之间关于客户数据丢失的原因。同时，Linkup的20,000名用户被告知该服务已不再可用，并被敦促尝试另一个存储站点。

> One solution would be to standardize the APIsd in such a way that a SaaS developer could deploy services and data across multiple cloud computing providers so that the failure of a single company would not take all copies of customer data with it. One might worry that this would lead to a “race-to-the-bottom” of cloud pricing and flatten the profits of cloud computing providers. We offer two arguments to allay this fear.

一种解决方案是以一种SaaS开发人员可以跨多个云计算提供商部署服务和数据的方式标准化API，以便单个公司的失败不会随之获取所有客户数据副本。有人可能会担心这会导致云定价的“竞争到底”，并使云计算提供商的利润趋于平缓。我们提出两个论点来消除这种恐惧。

> First, the quality of a service matters as well as the price, so customers may not jump to the lowest-cost service. Some Internet service providers today cost a factor of 10 more than others because they are more dependable and offer extra services to improve usability.

首先，服务质量和价格都很重要，因此客户可能不会跳到成本最低的服务。如今，一些互联网服务提供商的成本比其他因素高10倍，因为它们更可靠，并提供额外的服务来提高可用性。

> Second, in addition to mitigating data lock-in concerns, standardization of APIs enables a new usage model in which the same software infrastructure can be used in an internal data center and in a public cloud. Such an option could enable hybrid cloud computing or surge computing in which the public cloud is used to capture the extra tasks that cannot be easily run in the data center (or private cloud) due to temporarily heavy workloads. This option could significantly expand the cloud computing market. Indeed, open-source reimplementations of proprietary cloud APIs, such as Eucalyptus and HyperTable, are first steps in enabling surge computing.

其次，除了减轻数据锁定问题之外，API的标准化还支持新的使用模式，其中相同的软件基础架构可用于内部数据中心和公共云。这样的选项可以实现混合云计算或浪涌计算，其中公共云用于捕获由于临时繁重的工作负载而在数据中心（或私有云）中不能轻易运行的额外任务。此选项可以显着扩展云计算市场。实际上，专有云API的开源重新实现，例如Eucalyptus和HyperTable，是启用浪涌计算的第一步。

Number 3. Data Confidentiality/Auditability

> Despite most companies outsourcing payroll and many companies using external email services to hold sensitive information, security is one of the most often-cited objections to cloud computing; analysts and skeptical companies ask “who would trust their essential data out there somewhere?” There are also requirements for auditability, in the sense of Sarbanes-Oxley and Health and Human Services Health Insurance Portability and Accountability Act (HIPAA) regulations that must be provided for corporate data to be moved to the cloud.

尽管大多数公司外包工资单，许多公司使用外部电子邮件服务来保存敏感信息，但安全性是云计算最常被提及的反对意见之一。分析师和持怀疑态度的公司会问“谁会相信他们的基本数据？”对于必须提供的“萨班斯 - 奥克斯利法案”和“健康与人类服务健康保险流通与责任法案”（HIPAA）规定，还有可审计性要求将公司数据移至云端。

> Cloud users face security threats both from outside and inside the cloud. Many of the security issues involved in protecting clouds from outside threats are similar to those already facing large data centers. In the cloud, however, this responsibility is divided among potentially many parties, including the cloud user, the cloud vendor, and any third-party vendors that users rely on for security-sensitive software or configurations.

云用户面临来自云外部和内部的安全威胁。保护云免受外部威胁所涉及的许多安全问题与已经面临大型数据中心的问题类似。但是，在云中，此责任分为潜在的多方，包括云用户，云供应商以及用户依赖安全敏感软件或配置的任何第三方供应商。

> The cloud user is responsible for application-level security. The cloud provider is responsible for physical security, and likely for enforcing external firewall policies. Security for intermediate layers of the software stack is shared between the user and the operator; the lower the level of abstraction exposed to the user, the more responsibility goes with it. Amazon EC2 users have more technical responsibility (that is, must implement or procure more of the necessary functionality themselves) for their security than do Azure users, who in turn have more responsibilities than AppEngine customers. This user responsibility, in turn, can be outsourced to third parties who sell specialty security services. The homogeneity and standardized interfaces of platforms like EC2 make it possible for a company to offer, say, configuration management or firewall rule analysis as value-added services.

云用户负责应用程序级安全性。云提供商负责物理安全，并可能实施外部防火墙策略。 软件堆栈的中间层的安全性在用户和操作员之间共享; 暴露给用户的抽象级别越低，其承担的责任就越大。 与Azure用户相比，Amazon EC2用户对其安全性负有更多的技术责任（即，必须自己实施或获取更多必要的功能），Azure用户反过来比AppEngine客户承担更多责任。反过来，这种用户责任可以外包给销售专业安全服务的第三方。EC2等平台的同质性和标准化接口使公司可以提供配置管理或防火墙规则分析作为增值服务。

> While cloud computing may make external-facing security easier, it does pose the new problem of internalfacing security. Cloud providers must guard against theft or denial-of-service attacks by users. Users need to be protected from one another.

虽然云计算可以使面向外部的安全性变得更容易，但它确实带来了内部安全性的新问题。云提供商必须防止用户遭到盗窃或拒绝服务攻击。用户需要彼此保护。

> The primary security mechanism in today’s clouds is virtualization. It is a powerful defense, and protects against most attempts by users to attack one another or the underlying cloud infrastructure. However, not all resources are virtualized and not all virtualization environments are bug-free. Virtualization software has been known to contain bugs that allow virtualized code to “break loose” to some extent. Incorrect network virtualization may allow user code access to sensitive portions of the provider’s infrastructure, or to the resources of other users. These challenges, though, are similar to those involved in managing large non-cloud data centers, where different applications need to be protected from one another. Any large Internet service will need to ensure that a single security hole doesn’t compromise everything else.

当今云中的主要安全机制是虚拟化。它是一种强大的防御，可以防止用户进行大多数互相攻击或底层云基础架构的攻击。但是，并非所有资源都是虚拟化的，并非所有虚拟化环境都没有错误。众所周知，虚拟化软件包含允许虚拟化代码在某种程度上“破坏”的错误。不正确的网络虚拟化可能允许用户代码访问提供商的基础设施的敏感部分或其他用户的资源。但是，这些挑战与管理大型非云数据中心相关的挑战类似，其中不同的应用程序需要相互保护。任何大型互联网服务都需要确保单个安全漏洞不会影响其他任何漏洞。

> One last security concern is protecting the cloud user against the provider. The provider will by definition control the “bottom layer” of the software stack, which effectively circumvents most known security techniques. Absent radical improvements in security technology, we expect that users will use contracts and courts, rather than clever security engineering, to guard against provider malfeasance. The one important exception is the risk of inadvertent data loss. It’s difficult to imagine Amazon spying on the contents of virtual machine

> memory; it’s easy to imagine a hard disk being disposed of without being wiped, or a permissions bug making data visible improperly.

最后一个安全问题是保护云用户免受提供商的侵害。根据定义，提供者将控制软件栈的“底层”，这有效地绕过了大多数已知的安全技术。如果没有安全技术的根本改进，我们希望用户将使用合同和法院，而不是聪明的安全工程，以防止提供商的渎职行为。一个重要的例外是无意中丢失数据的风险。很难想象亚马逊会窥探虚拟机内存的内容; 很容易想象一个硬盘在没有被擦除的情况下被丢弃，或者一个权限错误使得数据不正确地被看见。

> This is a problem in non-cloud contexts as well. The standard defense, user-level encryption, is also effective in the cloud. This is already common for high-value data outside the cloud, and both tools and expertise are readily available. This approach was successfully used by TC3, a health care company with access to sensitive patient records and health care claims, when moving their HIPAA-compliant application to AWS.

这也是非云环境中的问题。标准防御，用户级加密在云中也很有效。这对于云外的高价值数据已经很常见，并且工具和专业知识都很容易获得。TC3是一家医疗保健公司，在将符合HIPAA标准的应用程序移至AWS时可以访问敏感的患者记录和医疗保健索赔，成功地使用了这种方法。

> Similarly, auditability could be added as an additional layer beyond the reach of the virtualized guest OS, providing facilities arguably more secure than those built into the applications themselves and centralizing the software responsibilities related to confidentiality and auditability into a single logical layer. Such a new feature reinforces the cloud computing perspective of changing our focus from specific hardware to the virtualized capabilities being provided.

同样，可审计性可以作为虚拟客户操作系统无法覆盖的附加层添加，提供的设施可以说比内置于应用程序本身的设施更安全，并将与机密性和可审计性相关的软件职责集中到一个逻辑层中。这一新功能强化了云计算的视角，即将重点从特定硬件转变为所提供的虚拟化功能。

Number 4. Data Transfer Bottlenecks

> Applications continue to become more data-intensive. If we assume applications may be “pulled apart” across the boundaries of clouds, this may complicate data placement and transport. At $100 to$150 per terabyte transferred, these costs can quickly add up, making data transfer costs an important issue. Cloud users and cloud providers have to think about the implications of placement and traffic at every level of the system if they want to minimize costs. This kind of reasoning can be seen in Amazon’s development of its new cloudfront service.

应用程序继续变得更加数据密集。如果我们假设应用程序可能跨云层“拉开”，则可能会使数据放置和传输复杂化。每TB转移100至150美元，这些成本可以迅速增加，使数据传输成本成为一个重要问题。如果云用户和云提供商希望最大限度地降低成本，则必须考虑系统各个层面的布局和流量的影响。亚马逊开发新的云端服务可以看出这种推理。

> One opportunity to overcome the high cost of Internet transfers is to ship disks. Jim Gray found the cheapest way to send a lot of data is to ship disks or even whole computers.10 While this does not address every use case, it effectively handles the case of large delay-tolerant point-to-point transfers, such as importing large data sets.

克服互联网传输成本高的一个机会是运送磁盘。Jim Gray发现发送大量数据的最便宜方式是发送磁盘甚至是整台计算机.10虽然这并不能解决每个用例，但它有效地处理了大容量延迟容忍点对点传输的情况，例如：导入大型数据集。

> To quantify the argument, assume that we want to ship 10TB from U.C.Berkeley to Amazon in Seattle, WA. Garfinkel measured bandwidth to S3 from three sites and found an average write bandwidth of 5Mbits/sec to 18Mbits/ sec. Suppose we get 20Mbits/sec over a WAN link. It would take
>
> 10 * 1012 Bytes / (20×106 bits/second) = (8×1013)/(2×107) seconds = 4,000,000 seconds,
>
> which is more than 45 days. If we instead sent 10 1TB disks via overnight shipping, it would take less than a day to transfer 10TB, yielding an effective bandwidth of about 1,500Mbit/sec. For example, AWS8 recently started offering such a service, called Import/Export.

为了量化这个论点，假设我们想要从华盛顿州西雅图的U.C.Berkeley运送10TB到亚马逊。Garfinkel从三个站点测量到S3的带宽，发现平均写入带宽为5Mbits / sec到18Mbits / sec。假设我们通过WAN链路获得20Mbits / sec。它需要10 * 1012字节/（20×106位/秒）=（8×1013）/（2×107）秒= 4,000,000秒，这超过45天。如果我们通过隔夜发货发送10个1TB磁盘，那么转移10TB将花费不到一天的时间，产生大约1,500Mbit /秒的有效带宽。例如，AWS8最近开始提供名为Import / Export的服务。

Number 5. Performance Unpredictability

> Our experience is that multiple virtual machines (VMs) can share CPUs and main memory surprisingly well in cloud computing, but that network and disk I/O sharing is more problematic. As a result, different EC2 instances vary more in their I/O performance than in main memory performance. We measured 75 EC2 instances running the STREAM memory benchmark.14 The mean bandwidth is 1,355Mbytes/ sec., with a standard deviation across instances of just 52MBytes/sec, less than or about 4% of the mean. We also measured the average disk bandwidth for 75 EC2 instances each writing 1GB files to local disk. The mean disk write bandwidth is nearly 55Mbytes per second with a standard deviation across instances of a little over 9MBytes/sec, or about 16% of the mean. This demonstrates the problem of I/O interference between virtual machines.

我们的经验是，多个虚拟机（VM）可以在云计算中出色地共享CPU和主内存，但网络和磁盘I / O共享更成问题。 因此，不同的EC2实例的I/O性能差异大于主内存性能。我们测量了运行STREAM内存基准测试的75个EC2实例.14平均带宽为1,355Mbytes/sec。实例的标准偏差仅为52MBytes/sec，小于或约为平均值的4%。我们还测量了每个将1GB文件写入本地磁盘的75个EC2实例的平均磁盘带宽。平均磁盘写入带宽接近每秒55M字节，跨越实例的标准偏差略高于9MBytes/sec，或约为平均值的16%。这表明虚拟机之间的I/O干扰问题。

> One opportunity is to improve architectures and operating systems to efficiently virtualize interrupts and I/O channels. Note that IBM mainframes and operating systems largely overcame these problems in the 1980s, so we have successful examples from which to learn.

一个机会是改进体系结构和操作系统，以有效地虚拟化中断和I/O通道。请注意，IBM大型机和操作系统在20世纪80年代基本上克服了这些问题，因此我们有成功的示例可供学习。

> Another possibility is that flash memory will decrease I/O interference. Flash is semiconductor memory that preserves information when powered off like mechanical hard disks, but since it has no moving parts, it is much faster to access (microseconds vs. milliseconds) and uses less energy. Flash memory can sustain many more I/Os per second per gigabyte of storage than disks, so multiple virtual machines with conflicting random I/O workloads could coexist better on the same physical computer without the interference we see with mechanical disks.

另一种可能性是闪存会降低I/O干扰。闪存是半导体存储器，可以像机械硬盘一样在断电时保存信息，但由于它没有移动部件，因此访问速度要快得多（微秒与毫秒）并且耗电量更少。与磁盘相比，闪存每GB每秒可以承受更多的I/O，因此具有冲突的随机I/O工作负载的多个虚拟机可以在同一台物理计算机上更好地共存，而不会受到机械磁盘的干扰。

> Another unpredictability obstacle concerns the scheduling of virtual machines for some classes of batch processing programs, specifically for highperformance computing. Given that high-performance computing (HPC) is used to justify government purchases of $100M supercomputer centers with 10,000 to 1,000,000 processors, there are many tasks with parallelism that can benefit from elastic computing. Today, many of these tasks are run on small clusters, which are often poorly utilized. There could be a significant savings in running these tasks on large clusters in the cloud instead. Cost associativity means there is no cost penalty for using 20 times as much computing for 1/20th the time. Potential applications that could benefit include those with very high potential financial returns—financial analysis, petroleum exploration, movie animation—that would value a 20x speedup even if there were a cost premium.

另外，不可预测性障碍涉及某些类批处理程序的虚拟机调度，特别是高性能计算。鉴于高性能计算（HPC）用于证明政府购买价值1亿至1,000,000台处理器的1亿美元超级计算机中心是合理的，因此有许多并行的任务可以从弹性计算中受益。今天，许多这些任务都是在小型集群上运行的，这些集群往往利用率很低。相反，在云中的大型集群上运行这些任务可能会有很大的节省。成本关联意味着在1/20的时间内使用20倍的计算没有成本损失。可能受益的潜在应用包括具有非常高的潜在财务回报的应用 - 财务分析，石油勘探，电影动画 - 即使存在成本溢价，也将重视20倍的加速。

> The obstacle to attracting HPC is not the use of clusters; most parallel computing today is done in large clusters using the message-passing interface MPI. The problem is that many HPC applications need to ensure that all the threads of a program are running simultaneously, and today’s virtual machines and operating systems do not provide a programmer-visible way to ensure this. Thus, the opportunity to overcome this obstacle is to offer something like “gang scheduling” for cloud computing. The relatively tight timing coordination expected in traditional gang scheduling may be challenging to achieve in a cloud computing environment due to the performance unpredictability just described.

吸引HPC的障碍不是使用集群;今天的大多数并行计算是使用消息传递接口MPI在大型集群中完成的。问题是许多HPC应用程序需要确保程序的所有线程同时运行，而今天的虚拟机和操作系统不提供程序员可见的方式来确保这一点。因此，克服这一障碍的机会是为云计算提供类似“群组调度”的东西。由于刚刚描述的性能不可预测性，在传统群组调度中预期的相对紧密的定时协调可能难以在云计算环境中实现。

Number 6: Scalable Storage

> Earlier, we identified three properties whose combination gives cloud computing its appeal: short-term usage (which implies scaling down as well as up when demand drops), no upfront cost, and infinite capacity on demand. While it’s straightforward what this means when applied to computation, it’s less clear how to apply it to persistent storage.

早些时候，我们确定了三个属性，它们的组合为云计算带来了吸引力：短期使用（这意味着在需求下降时缩小和向上），没有前期成本和无限的按需容量。虽然这在应用于计算时意味着什么，但是如何将其应用于持久存储却不太清楚。

> There have been many attempts to answer this question, varying in the richness of the query and storage API’s, the performance guarantees offered, and the resulting consistency semantics. The opportunity, which is still an open research problem, is to create a storage system that would not only meet existing programmer expectations in regard to durability, high availability, and the ability to manage and query data, but combine them with the cloud advantages of scaling arbitrarily up and down on demand.

已经有很多尝试回答这个问题，不同的是查询和存储API的丰富性，提供的性能保证以及由此产生的一致性语义。这个仍然是一个开放性研究问题的机会是创建一个存储系统，它不仅能够满足现有程序员对耐用性，高可用性以及管理和查询数据的能力的期望，还能将它们与云的优势结合起来。根据需要随意上下缩放。

Number 7: Bugs in LargeScale Distributed Systems

> One of the difficult challenges in cloud computing is removing errors in these very large-scale distributed systems. A common occurrence is that these bugs cannot be reproduced in smaller configurations, so the debugging must occur at scale in the production data centers.

云计算面临的一个难题是消除这些超大规模分布式系统中的错误。常见的情况是这些错误无法以较小的配置再现，因此调试必须在生产数据中心大规模进行。

> One opportunity may be the reliance on virtual machines in cloud computing. Many traditional SaaS providers developed their infrastructure without using VMs, either because they preceded the recent popularity of VMs or because they felt they could not afford the performance hit of VMs. Since VMs are de rigueur in utility computing, that level of virtualization may make it possible to capture valuable information in ways that are implausible without VMs.

一个机会可能是依赖云计算中的虚拟机。许多传统的SaaS提供商在不使用虚拟机的情况下开发了他们的基础设施，要么是因为他们先于最近流行的虚拟机，要么是因为他们认为他们无法承受虚拟机的性能损失。由于VM在公用计算中是必需的，因此该级别的虚拟化可以以不具有VM的方式难以接收的方式捕获有价值的信息。

Number 8: Scaling Quickly

> Pay-as-you-go certainly applies to storage and to network bandwidth, both of which count bytes used. Computation is slightly different, depending on the virtualization level. Google AppEngine automatically scales in response to load increases and decreases, and users are charged by the cycles used. AWS charges by the hour for the number of instances you occupy, even if your machine is idle.

即用即付肯定适用于存储和网络带宽，两者都计算使用的字节数。计算略有不同，具体取决于虚拟化级别。Google AppEngine会根据负载的增加和减少自动进行扩展，并根据使用的周期向用户收费。即使您的计算机处于空闲状态，AWS也会按小时收取您占用的实例数。

> The opportunity is then to automatically scale quickly up and down in response to load in order to save money, but without violating servicelevel agreements. Indeed, one focus of the UC Berkeley Reliable Adaptive Distributed Systems Laboratory is the pervasive and aggressive use of statistical machine learning as a diagnostic and predictive tool to allow dynamic scaling, automatic reaction to performance and correctness problems, and automatically managing many other aspects of these systems.

然后，有机会根据负载自动快速上下调整以节省资金，但不违反服务级别协议。实际上，加州大学伯克利分校可靠自适应分布式系统实验室的一个重点是普及和积极地使用统计机器学习作为诊断和预测工具，以允许动态扩展，自动响应性能和正确性问题，并自动管理这些的许多其他方面系统。

> Another reason for scaling is to conserve resources as well as money. Since an idle computer uses about two-thirds of the power of a busy computer, careful use of resources could reduce the impact of data centers on the environment, which is currently receiving a great deal of negative attention. Cloud computing providers already perform careful and low-overhead accounting of resource consumption. By imposing fine-grained costs, utility computing encourages programmers to pay attention to efficiency (that is, releasing and

> acquiring resources only when necessary), and allows more direct measurement of operational and development inefficiencies.

扩展的另一个原因是节省资源和资金。由于空闲计算机使用繁忙计算机大约三分之二的功率，因此小心使用资源可以减少数据中心对环境的影响，而环境目前正受到很大的负面关注。云计算提供商已经对资源消耗进行了谨慎且低开销的计算。通过实施细粒度成本，效用计算鼓励程序员关注效率（即仅在必要时释放和获取资源），并允许更直接地测量操作和开发效率低下。

> Being aware of costs is the first step to conservation, but configuration hassles make it tempting to leave machines idle overnight so that startup time is zero when developers return to work the next day. A fast and easy-to-use snapshot/restart tool might further encourage conservation of computing resources.

了解成本是保护的第一步，但是配置麻烦使得机器一夜之间保持闲置，以便当开发人员第二天重返工作岗位时，启动时间为零。快速且易于使用的快照/重启工具可能进一步鼓励保护计算资源。

Number 9: Reputation Fate Sharing

> One customer’s bad behavior can affect the reputation of others using the same cloud. For instance, blacklisting of EC2 IP addresses by spam-prevention services may limit which applications can be effectively hosted. An opportunity would be to create reputation-guarding services similar to the “trusted email” services currently offered (for a fee) to services hosted on smaller ISP’s, which experience a microcosm of this problem.

一个客户的不良行为可能会影响使用同一个云的其他人的声誉。例如，通过垃圾邮件防护服务将EC2 IP地址列入黑名单可能会限制哪些应用程序可以有效托管。一个机会是创建声誉保护服务，类似于当前提供的“可信电子邮件”服务（收费）到小型ISP托管的服务，这些服务经历了这个问题的一个缩影。

> Another legal issue is the question of transfer of legal liability—cloud computing providers would want customers to be liable and not them (such as, the company sending the spam should be held liable, not Amazon). In March 2009, the FBI raided a Dallas data center because a company whose services were hosted there was being investigated for possible criminal activity, but a number of “innocent bystander” companies hosted in the same facility suffered days of unexpected downtime, and some went out of business.

另一个法律问题是转移法律责任的问题 - 云计算提供商希望客户承担责任而不是他们（例如，发送垃圾邮件的公司应该承担责任，而不是亚马逊）。2009年3月，联邦调查局袭击了达拉斯数据中心，因为在那里托管服务的公司正在接受可能的犯罪活动调查，但在同一设施中托管的一些“无辜的旁观者”公司遭遇了数天的意外停机，有些人破产。

Number 10: Software Licensing

> Current software licenses commonly restrict the computers on which the software can run. Users pay for the software and then pay an annual maintenance fee. Indeed, SAP announced that it would increase its annual maintenance fee to at least 22% of the purchase price of the software, which is close to Oracle’s pricing. Hence, many cloud computing providers originally relied on open source software in part because the licensing model for commercial software is not a good match to utility computing.

当前的软件许可证通常会限制运行该软件的计算机。用户支付软件费用，然后支付年度维护费。事实上，SAP宣布将其年度维护费用增加至至少22%的软件购买价格，这接近甲骨文的定价。因此，许多云计算提供商最初依赖于开源软件，部分原因是商业软件的许可模式与效用计算不匹配。

> The primary opportunity is either for open source to remain popular or simply for commercial software companies to change their licensing structure to better fit cloud computing. For example, Microsoft and Amazon now offer pay-as-you-go software licensing for Windows Server and Windows SQL Server on EC2. An EC2 instance running Microsoft Windows costs $0.15perhourinsteadof$0.10 per hour for the open source alternative. IBM also announced pay-as-you-go pricing for hosted IBM software in conjunction with EC2, at prices ranging from $0.38perhourforDB2Expressto$6.39 per hour for IBM WebSphere with Lotus Web Content Management Server.

主要机会是开源保持流行，或者仅仅是商业软件公司改变其许可结构以更好地适应云计算。例如，微软和亚马逊现在为EC2上的Windows Server和Windows SQL Server提供即用即付软件许可。运行Microsoft Windows的EC2实例每小时收费0.15美元，而开源替代方案每小时收费0.10美元。IBM还宣布与EC2一起托管IBM软件的按需付费定价，价格从DB2 Express每小时0.38美元到IBM WebSphere with Lotus Web Content Management Server每小时6.39美元不等。

Conclusion

> We predict cloud computing will grow, so developers should take it into account. Regardless of whether a cloud provider sells services at a low level of abstraction like EC2 or a higher level like AppEngine, we believe computing, storage, and networking must all focus on horizontal scalability of virtualized resources rather than on single node performance. Moreover:

我们预测云计算将会增长，因此开发人员应将其考虑在内。无论云提供商是以低级抽象（如EC2）还是更高级别（如AppEngine）销售服务，我们都认为计算，存储和网络都必须关注虚拟化资源的水平可扩展性，而不是单节点性能。此外：

> 1. Applications software needs to both scale down rapidly as well as scale up, which is a new requirement. Such software also needs a pay-for-use licensing model to match needs of cloud computing.
> 2. Infrastructure software must be aware that it is no longer running on bare metal but on VMs. Moreover, metering and billing need to be built in from the start.
> 3. Hardware systems should be designed at the scale of a container (at least a dozen racks), which will be the minimum purchase size. Cost of operation will match performance and cost of purchase in importance, rewarding energy proportionality5 by putting idle portions of the memory, disk, and network into low-power mode. Processors should work well with VMs and flash memory should be added to the memory hierarchy, and LAN switches and WAN routers must improve in bandwidth and cost.

1.应用软件需要快速缩小规模并扩大规模，这是一项新要求。此类软件还需要付费使用许可模式以满足云计算的需求。

2.基础设施软件必须意识到它不再在裸机上运行，而是在VM上运行。此外，计量和计费需要从一开始就建立起来。

3.硬件系统应按容器（至少十几个机架）的规模设计，这将是最小购买尺寸。运营成本将重要地匹配性能和购买成本，通过将存储器，磁盘和网络的空闲部分置于低功耗模式来奖励能量比例5。处理器应与VM配合使用，并且应将闪存添加到内存层次结构中，LAN交换机和WAN路由器必须提高带宽和成本。