INTRODUCTION TO CLOUD COMPUTING ARCHITECTURE

White Paper
1st Edition, June 2009

Abstract

Cloud computing promises to increase the velocity with which applications are deployed, increase innovation, and lower costs, all while increasing business agility. Sun takes an inclusive view of cloud computing that allows it to support every facet, including the server, storage, network, and virtualization technology that drives cloud computing environments to the software that runs in virtual appliances that can be used to assemble applications in minimal time. This white paper discusses how cloud computing transforms the way we design, build, and deliver applications, and the architectural considerations that enterprises must make when adopting and using cloud computing technology.

This page intentionally left blank.

Table of Contents

Chapter 1

Introduction

Everyone has an opinion on what is cloud computing. It can be the ability to rent a server or a thousand servers and run a geophysical modeling application on the most powerful systems available anywhere. It can be the ability to rent a virtual server, load software on it, turn it on and off at will, or clone it ten times to meet a sudden workload demand. It can be storing and securing immense amounts of data that is accessible only by authorized applications and users. It can be supported by a cloud provider that sets up a platform that includes the OS, Apache, a MySQL™ database, Perl, Python, and PHP with the ability to scale automatically in response to changing workloads. Cloud computing can be the ability to use applications on the Internet that store and protect data while providing a service — anything including email, sales force automation and tax preparation. It can be using a storage cloud to hold application, business, and personal data. And it can be the ability to use a handful of Web services to integrate photos, maps, and GPS information to create a mashup in customer Web browsers.

Sun's perspective

Sun takes an inclusive view that there are many different types of clouds, and many different applications that can be built using them. To the extent that cloud computing helps to increase the velocity at which applications are deployed, helping to increase the pace of innovation, cloud computing may yet take forms that we still cannot imagine today. What remains constant, however, is that Sun is an experienced provider of server, storage, networking, and software technology that is ready to support cloud computing. As the company that coined the phrase "The Network is the Computer™," we believe that cloud computing is the next generation of network computing.

What distinguishes cloud computing from previous models? Boiled down to a phrase, it's using information technology as a service over the network. We define it as services that are encapsulated, have an API, and are available over the network. This definition encompasses using both compute and storage resources as services. Cloud computing is based on the principle of efficiency above all — efficiency that produces high-level tools for handling 80% of use cases so that applications can be created and deployed at an astonishing rate.

Cloud computing can be provided using an enterprise datacenter's own servers, or it can be provided by a cloud provider that takes all of the capital risk of owning the infrastructure. The illusion is that resources are infinite. While the field is in its infancy, the model is taking the information technology (IT) world by storm. The

predominant model for cloud computing today is called *infrastructure as a service*, or IaaS, and because of its prominence, the IaaS model is the focus of the first edition of this white paper.

This paper discusses the nature of cloud computing and how it builds on established trends while transforming the way that enterprises everywhere build and deploy applications. It proceeds to discuss the architectural considerations that cloud architects must make when designing cloud-based applications, concluding with a discussion of Sun's technologies that support cloud computing.

Chapter 2

The Nature of Cloud Computing

Building on established trends

Cloud computing builds on established trends for driving the cost out of the delivery of services while increasing the speed and agility with which services are deployed. It shortens the time from sketching out an application architecture to actual deployment. Cloud computing incorporates virtualization, on-demand deployment, Internet delivery of services, and open source software. From one perspective, cloud computing is nothing new because it uses approaches, concepts, and best practices that have already been established. From another perspective, everything is new because cloud computing changes how we invent, develop, deploy, scale, update, maintain, and pay for applications and the infrastructure on which they run. In this chapter, we examine the trends and how they have become core to what cloud computing is all about.

Virtual machines as the standard deployment object

Over the last several years, virtual machines have become a standard deployment object. Virtualization further enhances flexibility because it abstracts the hardware to the point where software stacks can be deployed and redeployed without being tied to a specific physical server. Virtualization enables a dynamic datacenter where servers provide a pool of resources that are harnessed as needed, and where the relationship of applications to compute, storage, and network resources changes dynamically in order to meet both workload and business demands. With application deployment decoupled from server deployment, applications can be deployed and scaled rapidly, without having to first procure physical servers.

Virtual machines have become the prevalent abstraction — and unit of deployment — because they are the least-common denominator interface between service providers and developers. Using virtual machines as deployment objects is sufficient for 80 percent of usage, and it helps to satisfy the need to rapidly deploy and scale applications.

Virtual appliances, virtual machines that include software that is partially or fully configured to perform a specific task such as a Web or database server, further enhance the ability to create and deploy applications rapidly. The combination of virtual machines and appliances as standard deployment objects is one of the key features of cloud computing.

Compute clouds are usually complemented by storage clouds that provide virtualized storage through APIs that facilitate storing virtual machine images, source files for components such as Web servers, application state data, and general business data.

The on-demand, self-service, pay-by-use model

The on-demand, self-service, pay-by-use nature of cloud computing is also an extension of established trends. From an enterprise perspective, the on-demand nature of cloud computing helps to support the performance and capacity aspects of service-level objectives. The self-service nature of cloud computing allows organizations to create elastic environments that expand and contract based on the workload and target performance parameters. And the pay-by-use nature of cloud computing may take the form of equipment leases that guarantee a minimum level of service from a cloud provider.

Virtualization is a key feature of this model. IT organizations have understood for years that virtualization allows them to quickly and easily create copies of existing environments —sometimes involving multiple virtual machines — to support test, development, and staging activities. The cost of these environments is minimal because they can coexist on the same servers as production environments because they use few resources.

Likewise, new applications can be developed and deployed in new virtual machines on existing servers, opened up for use on the Internet, and scaled if the application is successful in the marketplace. This lightweight deployment model has already led to a “Darwinistic” approach to business development where beta versions of software are made public and the market decides which applications deserve to be scaled and developed further or quietly retired.

Cloud computing extends this trend through automation. Instead of negotiating with an IT organization for resources on which to deploy an application, a compute cloud is a self-service proposition where a credit card can purchase compute cycles, and a Web interface or API is used to create virtual machines and establish network relationships between them. Instead of requiring a long-term contract for services with an IT organization or a service provider, clouds work on a pay-by-use, or pay-by-the-sip model where an application may exist to run a job for a few minutes or hours, or it may exist to provide services to customers on a long-term basis. Compute clouds are built as if applications are temporary, and billing is based on resource consumption: CPU hours used, volumes of data moved, or gigabytes of data stored.

The ability to use and pay for only the resources used shifts the risk of how much infrastructure to purchase from the organization developing the application to the cloud provider. It also shifts the responsibility for architectural decisions from

application architects to developers. This shift can increase risk, risk that must be managed by enterprises that have processes in place for a reason, and of system, network, and storage architects that needs to factor in to cloud computing designs.

Infrastructure is programmable

This shift of architectural responsibility has significant consequences. In the past, architects would determine how the various components of an application would be laid out onto a set of servers, how they would be interconnected, secured, managed, and scaled. Now, a developer can use a cloud provider's API to create not only an application's initial composition onto virtual machines, but also how it scales and evolves to accommodate workload changes.

Consider this analogy: historically, a developer writing software using the Java™ programming language determines when it's appropriate to create new threads to allow multiple activities to progress in parallel. Today, a developer can discover and attach to a service with the same ease, allowing them to scale an application to the point where it might engage thousands of virtual machines in order to accommodate a huge spike in demand.

The ability to program an application architecture dynamically puts enormous power in the hands of developers with a commensurate amount of responsibility. To use cloud computing most effectively, a developer must also be an architect, and that architect needs to be able to create a self-monitoring and self-expanding application. The developer/architect needs to understand when it's appropriate to create a new thread versus create a new virtual machine, along with the architectural patterns for how they are interconnected.

When this power is well understood and harnessed, the results can be spectacular. A story that is already becoming legendary is Animoto's mashup tool that creates a video from a set of images and music. The company's application scaled from 50 to 3,500 servers in just three days due in part to an architecture that allowed it to scale easily. For this to work, the application had to be built to be horizontal scaled, have limited state, and manage its own deployment through cloud APIs. For every success story such as this, there will likely be a similar story where the application is not capable of self-scaling and where it fails to meet consumer demand. The importance of this shift from developer to developer/architect cannot be understated.

Consider whether your enterprise datacenter could scale an application this rapidly to accommodate such a rapidly growing workload, and whether cloud computing could augment your current capabilities.

Applications are composed and are built to be composable

Another consequence of the self-service, pay-by-use model is that applications are composed by assembling and configuring appliances and open-source software as much as they are programmed. Applications and architectures that can be refactored in order to make the most use of standard components are those that will be the most successful in leveraging the benefits of cloud computing. Likewise, application components should be designed to be composable by building them so they can be consumed easily. This requires having simple, clear functions, and well-documented APIs. Building large, monolithic applications is a thing of the past as the library of existing tools that can be used directly or tailored for a specific use becomes ever larger.

For a description of how this feat was accomplished, please visit: http://open.blogs.nytimes.com/2007/11/01/self-service-prorated-super-computing-fun/

For example, tools such as Hadoop, an open-source MapReduce implementation, can be used in a wide range of contexts in which a problem and its data can be refactored so that many parts of it can execute in parallel. When The New York Times wished to convert 11 million articles and images in its archive to PDF format, their internal IT organization said that it would take seven weeks. In the mean time, one developer using 100 Amazon EC2 simple Web service interface instances running Hadoop completed the job in 24 hours for less than $300. (This did not include the time required to upload the data or the cost of the storage.)

Even large corporations can use cloud computing in ways that solve significant problems in less time and at a lower cost than with traditional enterprise computing.

Example Web application deployment

As an example of how the combination of virtualization and self service facilitate application deployment, consider a two-tier Web application deployment into a cloud (Figure 1):

1. A developer might choose a load balancer, Web server, and database server appliances from a library of preconfigured virtual machine images.
2. The developer would configure each component to make a custom image. The load balancer would be configured, the Web server populated with its static content by uploading it to the storage cloud, and the database server appliances populated with dynamic content for the site.
3. The developer layers custom code into the new architecture, making the components meet specific application requirements.
4. The developer chooses a pattern that takes the images for each layer and deploys them, handling networking, security, and scalability issues.

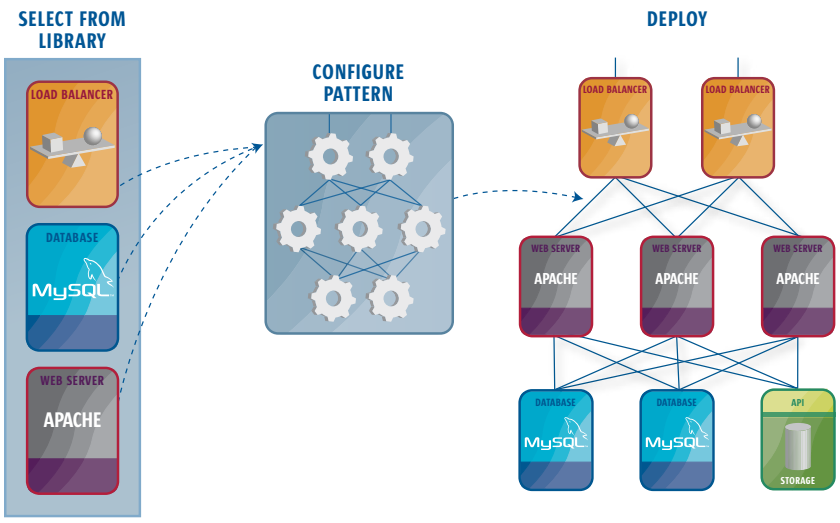

Figure 1. Example cloud-based deployment of an application onto a two-tier Web server architectural pattern.

5. The secure, high-availability Web application is up and running. When the application needs to be updated, the virtual machine images can be updated, versioned, copied across the development-test-production chain, and the entire infrastructure redeployed. Cloud computing assumes that everything is temporary, and it's just as easy to redeploy an entire application than it is to manually patch a set of individual virtual machines.

In this example, the abstract nature of virtual machine images supports a composition-based approach to application development. By refactoring the problem, a standard set of components can be used to quickly deploy an application. With this model, enterprise business needs can be met quickly, without the need for the time-consuming, manual purchase, installation, cabling, and configuration of servers, storage, and network infrastructure.

Services are delivered over the network

It almost goes without saying that cloud computing extends the existing trend of making services available over the network. Virtually every business organization has recognized the value of Web-based interfaces to their applications, whether they are made available to customers over the Internet, or whether they are internal applications that are made available to authorized employees, partners, suppliers, and consultants. The beauty of Internet-based service delivery, of course, is that applications can be made available anywhere, and at any time.

While enterprises are well aware of the ability to secure communications using Secure Socket Layer (SSL) encryption along with strong authentication, bootstrapping trust in a cloud computing environment requires carefully considering

the differences between enterprise computing and cloud computing. When properly architected, Internet service delivery can provide the flexibility and security required by enterprises of all sizes.

The role of open source software

Open source software plays an important role in cloud computing by allowing its basic software elements — virtual machine images and appliances — to be created from easily accessible components. This has an amplifying effect:

- Developers, for example, can create a database appliance by layering MySQL software onto an instance of the OpenSolaris™ Operating System and performing customizations (Figure 2). Appliances such as these enable cloud computing applications to be created, deployed, and dynamically scaled on demand. Consider, for example, how open source software allows an application such as that created by Animoto to scale to 3,500 instances in a matter of days.

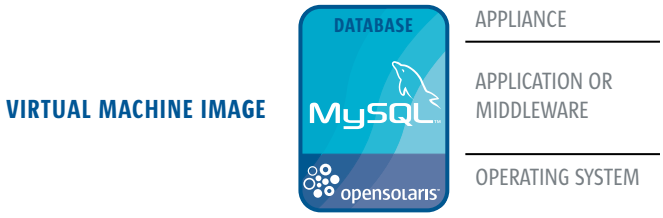

Figure 2. Appliances can be created by layering open source software into a virtual machine image and performing customizations that simplify their deployment. In this example, a database appliance is created by layering MySQL software on top of the OpenSolaris Operating System.

- The ease with which open source components can be used to assemble large applications generates more open source components. This, in turn, makes the role of open source software even more important. The need, for example, to have a MapReduce algorithm that can run in a cloud-computing environment, was one of the factors stimulating its development. Now that the tool has been created, it is being used to further raise the level at which developers 'program' cloud computing applications.

Cloud computing infrastructure models

There are many considerations for cloud computing architects to make when moving from a standard enterprise application deployment model to one based on cloud computing. There are public and private clouds that offer complementary benefits, there are three basic service models to consider, and there is the value of open APIs versus proprietary ones.

Public, private, and hybrid clouds

IT organizations can choose to deploy applications on public, private, or hybrid clouds, each of which has its trade-offs. The terms *public*, *private*, and *hybrid* do not dictate location. While public clouds are typically "out there" on the Internet and private clouds are typically located on premises, a private cloud might be hosted at a colocation facility as well.

Companies may make a number of considerations with regard to which cloud computing model they choose to employ, and they might use more than one model to solve different problems. An application needed on a temporary basis might be best suited for deployment in a public cloud because it helps to avoid the need to purchase additional equipment to solve a temporary need. Likewise, a permanent application, or one that has specific requirements on quality of service or location of data, might best be deployed in a private or hybrid cloud.

Public clouds

Public clouds are run by third parties, and applications from different customers are likely to be mixed together on the cloud's servers, storage systems, and networks (Figure 3). Public clouds are most often hosted away from customer premises, and they provide a way to reduce customer risk and cost by providing a flexible, even temporary extension to enterprise infrastructure.

If a public cloud is implemented with performance, security, and data locality in mind, the existence of other applications running in the cloud should be transparent to both cloud architects and end users. Indeed, one of the benefits of public clouds is that they can be much larger than a company's private cloud might be, offering the ability to scale up and down on demand, and shifting infrastructure risks from the enterprise to the cloud provider, if even just temporarily.

Portions of a public cloud can be carved out for the exclusive use of a single client, creating a virtual private datacenter. Rather than being limited to deploying virtual machine images in a public cloud, a virtual private datacenter gives customers greater visibility into its infrastructure. Now customers can manipulate not just virtual machine images, but also servers, storage systems, network devices, and

network topology. Creating a virtual private datacenter with all components located in the same facility helps to lessen the issue of data locality because bandwidth is abundant and typically free when connecting resources within the same facility.

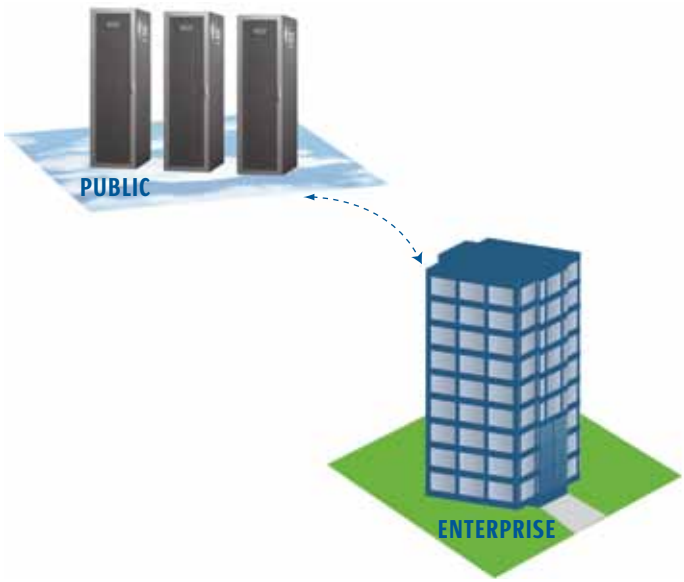

Figure 3. A public cloud provides services to multiple customers, and is typically deployed at a colocation facility.

Private clouds

Private clouds are built for the exclusive use of one client, providing the utmost control over data, security, and quality of service (Figure 4). The company owns the infrastructure and has control over how applications are deployed on it. Private clouds may be deployed in an enterprise datacenter, and they also may be deployed at a colocation facility.

Private clouds can be built and managed by a company's own IT organization or by a cloud provider. In this "hosted private" model, a company such as Sun can install, configure, and operate the infrastructure to support a private cloud within a company's enterprise datacenter. This model gives companies a high level of control over the use of cloud resources while bringing in the expertise needed to establish and operate the environment.

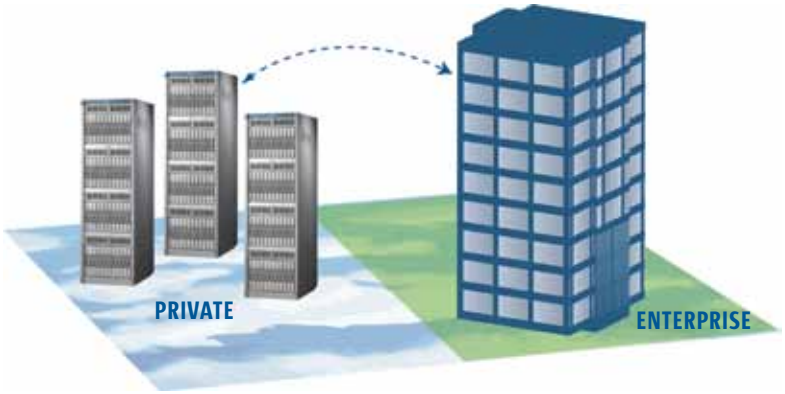

Figure 4. Private clouds may be hosted at a colocation facility or in an enterprise datacenter. They may be supported by the company, by a cloud provider, or by a third party such as an outsourcing firm.

Hybrid clouds

Hybrid clouds combine both public and private cloud models (Figure 5). They can help to provide on-demand, externally provisioned scale. The ability to augment a private cloud with the resources of a public cloud can be used to maintain service levels in the face of rapid workload fluctuations. This is most often seen with the use of storage clouds to support Web 2.0 applications. A hybrid cloud also can be used to handle planned workload spikes. Sometimes called "surge computing," a public cloud can be used to perform periodic tasks that can be deployed easily on a public cloud.

Hybrid clouds introduce the complexity of determining how to distribute applications across both a public and private cloud. Among the issues that need to be considered is the relationship between data and processing resources. If the data is small, or the application is stateless, a hybrid cloud can be much more successful than if large amounts of data must be transferred into a public cloud for a small amount of processing.

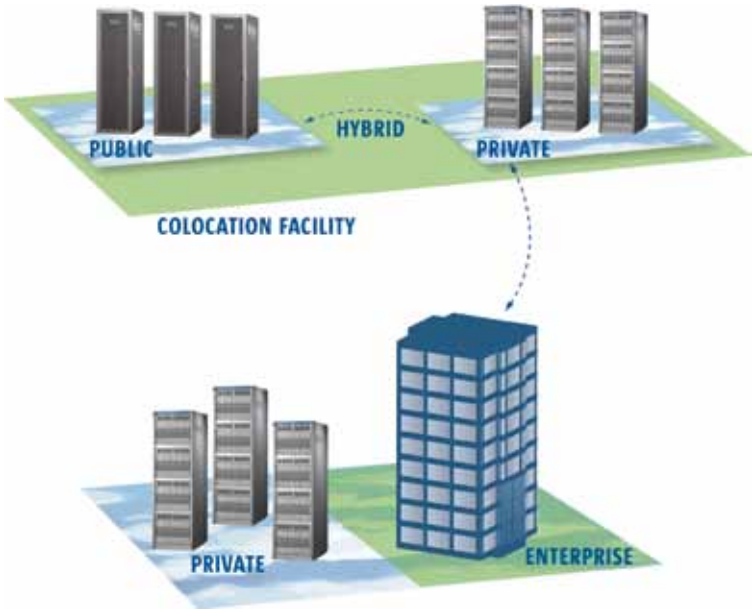

Figure 5. Hybrid clouds combine both public and private cloud models, and they can be particularly effective when both types of cloud are located in the same facility.

Architectural layers of cloud computing

Sun's view of cloud computing is an inclusive one: cloud computing can describe services being provided at any of the traditional layers from hardware to applications (Figure 6). In practice, cloud service providers tend to offer services that can be grouped into three categories: software as a service, platform as a service, and infrastructure as a service. These categories group together the various layers illustrated in Figure 6, with some overlap.

Software as a service (SaaS)

Software as a service features a complete application offered as a service on demand. A single instance of the software runs on the cloud and services multiple end users or client organizations.

The most widely known example of SaaS is salesforce.com, though many other examples have come to market, including the Google Apps offering of basic business services including email and word processing.

Although salesforce.com preceded the definition of cloud computing by a few years, it now operates by leveraging its companion force.com, which can be defined as a platform as a service.

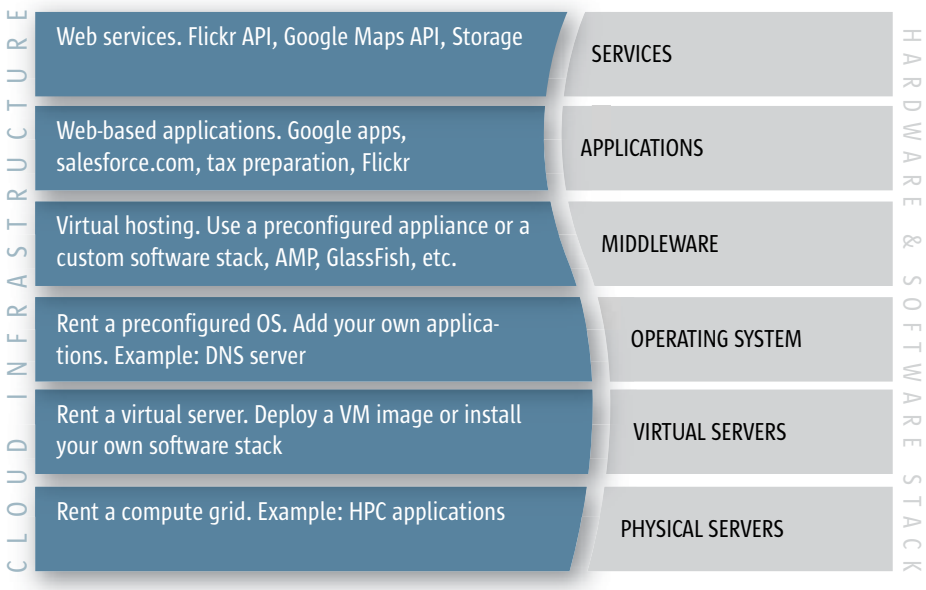

Figure 6. Cloud computing means using IT infrastructure as a service — and that service may be anything from renting raw hardware to using third-party APIs.

Platform as a service (PaaS)

Platform as a service encapsulates a layer of software and provides it as a service that can be used to build higher-level services. There are at least two perspectives on PaaS depending on the perspective of the producer or consumer of the services:

- Someone *producing* PaaS might produce a platform by integrating an OS, middleware, application software, and even a development environment that is then provided to a customer as a service. For example, someone developing a PaaS offering might base it on a set of Sun™ xVM hypervisor virtual machines that include a NetBeans™ integrated development environment, a Sun GlassFish™ Web stack and support for additional programming languages such as Perl or Ruby.
- Someone *using* PaaS would see an encapsulated service that is presented to them through an API. The customer interacts with the platform through the API, and the platform does what is necessary to manage and scale itself to provide a given level of service. Virtual appliances can be classified as instances of PaaS. A content switch appliance, for example, would have all of its component software hidden from the customer, and only an API or GUI for configuring and deploying the service provided to them.

PaaS offerings can provide for every phase of software development and testing, or they can be specialized around a particular area such as content management. Commercial examples of PaaS include the Google Apps Engine, which serves

applications on Google's infrastructure. PaaS services such as these can provide a powerful basis on which to deploy applications, however they may be constrained by the capabilities that the cloud provider chooses to deliver.

Infrastructure as a service (IaaS)

Infrastructure as a service delivers basic storage and compute capabilities as standardized services over the network. Servers, storage systems, switches, routers, and other systems are pooled and made available to handle workloads that range from application components to high-performance computing applications.

Commercial examples of IaaS include Joyent, whose main product is a line of virtualized servers that provide a highly available on-demand infrastructure.

Cloud application programming interfaces

One of the key characteristics that distinguishes cloud computing from standard enterprise computing is that the infrastructure itself is programmable. Instead of physically deploying servers, storage, and network resources to support applications, developers specify how the same virtual components are configured and interconnected, including how virtual machine images and application data are stored and retrieved from a storage cloud. They specify how and when components are deployed through an API that is specified by the cloud provider.

An analogy is the way in which File Transfer Protocol (FTP) works: FTP servers maintain a control connection with the client that is kept open for the duration of the session. When files are to be transferred, the control connection is used to provide a source or destination file name to the server, and to negotiate a source and destination port for the file transfer itself. In a sense, a cloud computing API is like an FTP control channel: it is open for the duration of the cloud's use, and it controls how the cloud is harnessed to provide the end services envisioned by the developer.

The use of APIs to control how cloud infrastructure is harnessed has a pitfall: unlike the FTP protocol, cloud APIs are not yet standardized, so each cloud provider has its own specific APIs for managing its services. This is the typical state of an industry in its infancy, where each vendor has its own proprietary technology that tends to lock in customers to their services because proprietary APIs make it difficult to change providers.

Look for providers that use standard APIs wherever possible. Standard APIs can be used today for access to storage; APIs for deploying and scaling applications are likely to be standardized over time. Also look for cloud providers that understand their own market and provide, for example, ways to archive and deploy libraries of virtual machine images and preconfigured appliances.

Cloud computing benefits

In order to benefit the most from cloud computing, developers must be able to refactor their applications so that they can best use the architectural and deployment paradigms that cloud computing supports. The benefits of deploying applications using cloud computing include reducing run time and response time, minimizing the risk of deploying physical infrastructure, lowering the cost of entry, and increasing the pace of innovation.

Reduce run time and response time

For applications that use the cloud essentially for running batch jobs, cloud computing makes it straightforward to use 1000 servers to accomplish a task in 1/1000 the time that a single server would require. The New York Times example cited previously is the perfect example of what is essentially a batch job whose run time was shortened considerably using the cloud.

For applications that need to offer good response time to their customers, refactoring applications so that any CPU-intensive tasks are farmed out to 'worker' virtual machines can help to optimize response time while scaling on demand to meet customer demands. The Animoto application cited previously is a good example of how the cloud can be used to scale applications and maintain quality of service levels.

Minimize infrastructure risk

IT organizations can use the cloud to reduce the risk inherent in purchasing physical servers. Will a new application be successful? If so, how many servers are needed and can they be deployed as quickly as the workload increases? If not, will a large investment in servers go to waste? If the application's success is short-lived, will the IT organization invest in a large amount of infrastructure that is idle most of the time?

When pushing an application out to the cloud, scalability and the risk of purchasing too much or too little infrastructure becomes the cloud provider's issue. In a growing number of cases, the cloud provider has such a massive amount of infrastructure that it can absorb the growth and workload spikes of individual customers, reducing the financial risk they face.

Another way in which cloud computing minimizes infrastructure risk is by enabling surge computing, where an enterprise datacenter (perhaps one that implements a private cloud) augments its ability to handle workload spikes by a design that allows it to send overflow work to a public cloud. Application lifecycle management can be handled better in an environment where resources are no longer scarce, and where resources can be better matched to immediate needs, and at lower cost.

Lower cost of entry

There are a number of attributes of cloud computing that help to reduce the cost to enter new markets:

- Because infrastructure is rented, not purchased, the cost is controlled, and the capital investment can be zero. In addition to the lower costs of purchasing compute cycles and storage "by the sip," the massive scale of cloud providers helps to minimize cost, helping to further reduce the cost of entry.
- Applications are developed more by assembly than programming. This rapid application development is the norm, helping to reduce the time to market, potentially giving organizations deploying applications in a cloud environment a head start against the competition.

Increased pace of innovation

Cloud computing can help to increase the pace of innovation. The low cost of entry to new markets helps to level the playing field, allowing start-up companies to deploy new products quickly and at low cost. This allows small companies to compete more effectively with traditional organizations whose deployment process in enterprise datacenters can be significantly longer. Increased competition helps to increase the pace of innovation — and with many innovations being realized through the use of open source software, the entire industry serves to benefit from the increased pace of innovation that cloud computing promotes.

Chapter 3

Architectural Considerations for IaaS

Evolving application architectures

Just as we have shown that cloud computing is a natural extension of current trends and best practices, the same is true when viewing cloud computing from an architectural perspective. Again, cloud computing is nothing new, yet in its implementation, it changes everything that we do.

Changing approaches to architecture

In the 1990s, the conversation was on how to decompose an application into its various components and then how to deploy those components on separate servers in order to optimize non-functional requirements including scalability, availability, manageability, and security. Today, we are maintaining a decomposed application architecture while actually deploying onto a consolidated architecture that uses virtualization.

Cloud computing continues this trend by providing a way to programmatically deploy application architectures, finally delivering on the promise of a dynamic datacenter. With cloud computing, efficiency is highly valued; if it can't be done quickly and programmatically, it probably isn't an application that is suited to the model.

Changing application designs

In the past, applications were built to handle larger workloads through vertical scaling. Put more processors and memory on a mail server to handle a larger volume of traffic. Scale up a database server to increase throughput. Run high-performance computing jobs on a supercomputer.

The movement away from highly scalable symmetric multiprocessors and toward less expensive, but less scalable x86-architecture servers has influenced application design. Rather than expecting applications to run on highly scalable servers, developers have been refactoring their applications so that they can scale horizontally across a number of servers. This application refactoring is not always easy, as both applications and their data must be designed so that both processing and data can be factored into smaller chunks. This existing architectural trend has been a key factor propelling the adoption of cloud computing. Examples of this trend include:

High-performance computing

HPC workloads have been running on bare-metal compute grids for some time now, enabled by application refactoring. For example, scientists have found ways to chunk down data for applications such as 3D climate modeling so that it can be spread across a large number of servers. Grid computing is a predecessor to cloud computing in that it uses tools to provision and manage multiple racks of physical servers so that they all can work together to solve a problem. With its high compute, interprocess communication, and I/O demands, HPC workloads are good candidates for clouds that provide infrastructure as a service, specifically bare-metal servers or Type I virtual machines that provide more direct access to I/O devices.

Database management systems

Database management systems have adapted to run in cloud environments by horizontally scaling database servers and partitioning tables across them. This technique, known as sharding, allows multiple instances of database software — often MySQL software — to scale performance in a cloud environment. Rather than accessing a single, central database, applications now access one of many database instances depending on which shard contains the desired data (Figure 7)

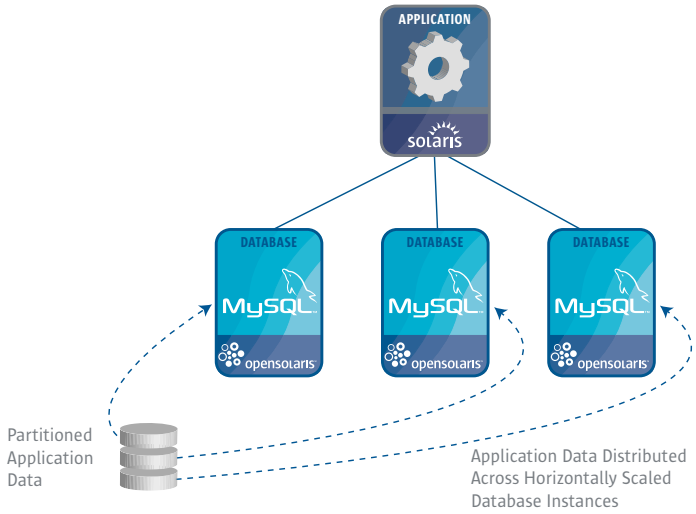

Figure 7. Database sharding partitions database tables across multiple database management system instances, supporting large databases through horizontal scaling.

CPU-intensive processing

Applications that perform activities such as frame rendering have been designed so that, rather than creating a new thread for each frame, they create a separate virtual machine to render each frame, increasing performance through horizontal scaling.

Data-intensive processing

Generalized tools are being developed by the open source community that assist in the processing of large amounts of data and then coalesce the results up to a coordinating process. Hadoop, for example, is an open source implementation of the MapReduce problem that integrates the deployment of 'worker' virtual machines with the data they need.

The goals remain the same

Numerous advances in application architecture have helped to promote the adoption of cloud computing. These advances help to support the goal of efficient application development while helping applications to be elastic and scale gracefully and automatically. The overriding objective of good application architectures, however, has not changed at all: it is to support the same characteristics that have always been important:

- *Scalability*. This characteristic is just as important as it has ever been. Applications designed for cloud computing need to scale with workload demands so that performance and compliance with service levels remain on target. In order to achieve this, applications and their data must be loosely coupled to maximize scalability. The term *elastic* often applies to scaling cloud applications because they must not only be ready to scale up, but also scale down as workloads diminish in order to not run up the cost of deploying in the cloud.
- *Availability*. Whether the application serves the users of social networking sites, or it manages the supply chain for a large manufacturing company, users of Internet applications expect them to be up and running every minute of every day. Sun has been an industry leader in this area establishing early on its SunTone^{SM} certification program that helped customers to certify that its applications and services would stand up to required availability levels.
- *Reliability*. The emphasis on reliability has shifted over time. When large applications meant large symmetric multiprocessing systems, reliability meant that system components rarely fail and can be replaced without disruption when they do. Today, reliability means that applications do not fail and most importantly they do not lose data. The way that architecture addresses this characteristic today is to design applications so that they continue to operate and their data remains intact despite the failure of one or more of the servers or virtual machines onto which they are decomposed. Where we once worried about the failure of individual server components, now we build applications so that entire servers can fail and not cause disruption.
- *Security*. Applications need to provide access only to authorized, authenticated users, and those users need to be able to trust that their data is secure. This is true whether the application helps individual users on the Internet prepare

their tax returns, or whether the application exchanges confidential information between a company and its suppliers. Security in today's environments is established using strong authentication, authorization, and accounting procedures, establishing security of data at rest and in transit, locking down networks, and hardening operating systems, middleware, and application software. It is such a systemic property that we no longer call it out as its own principle — security must be integrated into every aspect of an application and its deployment and operational architecture and processes.

- *Flexibility and agility*. These characteristics are increasingly important, as business organizations find themselves having to adapt even more rapidly to changing business conditions by increasing the velocity at which applications are delivered into customer hands. Cloud computing stresses getting applications to market very quickly by using the most appropriate building blocks to get the job done rapidly.
- *Serviceability*. Once an application is deployed, it needs to be maintained. In the past this meant using servers that could be repaired without, or with minimal, downtime. Today it means that an application's underlying infrastructure components can be updated or even replaced without disrupting its characteristics including availability and security.
- *Efficiency*. This is the new characteristic on the list, and it is perhaps one that most differentiates the cloud computing style from others. Efficiency is the point of cloud computing, and if an application can't be deployed in the cloud quickly and easily, while benefitting from the pay-by-the-sip model, it may not be a good candidate. Enterprise resource planning applications, for example, may be best suited to vertically scaled systems and provided through SaaS in the near term. Applications that extract, manipulate, and present data derived from these systems, however, may be well suited to deployment in the cloud.

Consistent and stable abstraction layer

Cloud computing raises the level of abstraction so that all components are abstracted or virtualized, and can be used to quickly compose higher-level applications or platforms. If a component does not provide a consistent and stable abstraction layer to its clients or peers, it's not appropriate for cloud computing.

The standard deployment unit is a virtual machine, which by its very nature is designed to run on an abstract hardware platform. It's easy to over focus on building virtual machine images and forget about the model that was used to create them. In cloud computing, it's important to maintain the model, not the image itself. The model is maintained; the image is produced from the model.

Virtual machine images will always change because the layers of software within them will always need to be patched, upgraded, or reconfigured. What doesn't change is the process of creating the virtual machine image, and this is what developers should focus on. A developer might build a virtual machine image by layering a Web server, application server, and MySQL database server onto an operating system image, applying patches, configuration changes, and interconnecting components at each layer. Focusing on the model, rather than the virtual machine image, allows the images themselves to be updated as needed by re-applying the model to a new set of components.

With this standard deployment unit, cloud architects can use appliances that help to speed deployment with lower costs. A developer might use an appliance that is preconfigured to run Hadoop on the OpenSolaris OS by interacting with the appliance's API. Architects can use content switches that are deployed not as physical devices, but as virtual appliances. All that needs to be done to deploy it is interact with its API or GUI. Even companies producing licensed, commercial software are adapting to cloud computing with more flexible, use-based licensing models.

Whether invoking a model that creates a virtual machine image, or customizing an appliance, the resulting virtual machine images need to be stored in a library of images that the enterprise versions and supports.

Standards help to address complexity

Cloud computing emphasizes efficiency above all, so adopting a small number of standards and standard configurations helps to reduce maintenance and deployment costs. Having standards that make deployment easy is more important than having the perfect environment for the job. The 80/20 rule comes into play here: cloud computing focuses on the few standards that can support 80% of the use cases. This shifts the economics from costly, one-off implementations to choosing the building blocks that can be used in the largest volume. There will continue to be specialization, however the starting point should be with a standard.

For an enterprise shifting to cloud computing, standards may include the type of virtual machine, the operating system in standard virtual machine images, tools, and programming languages supported:

- *Virtual machine types*. Consider the impact of virtual machine choice on the application to be supported. For a social networking application, isolation for security, and a high level of abstraction for portability, would suggest using Type II virtual machines. For a high-performance computing or visualization applications, the need to access hardware directly to achieve the utmost performance would suggest using Type I virtual machines.

- *Preinstalled, preconfigured systems*. The software on virtual machines must be maintained just as it does on a physical server. Operating systems still need to be hardened, patched, and upgraded. Having a small, standard set of supported configurations allows developers to use the current supported virtual machine. When the supported configuration is updated, the model dictating customizations should be designed so that it's easy to re-apply changes to a new virtual machine image. The same is true for appliances, where the current version can be configured through their standard APIs.
- *Tools and languages*. Enterprises might standardize on using the Java programming language and Ruby on Rails; small businesses might standardize on PHP as their preferred tools for building applications. As these standards mature in the context of cloud computing, they start to form the next layer, platform as a service.

Virtualization and encapsulation supports refactoring

When applications are refactored and created by combining and configuring a set of virtual machine images and appliances, the emphasis is on what a particular virtual machine *does*, not how it's *implemented*. Virtualization and encapsulation hides implementation details and refocuses developers on the interfaces and interactions between components. These components should provide standard interfaces so that developers can build applications quickly and easily — as well as use alternate components with similar functionality as performance or cost dictates.

Application deployment is done programmatically, and even the programs that deploy applications can be encapsulated so that they can be used and re-used. A program that deploys a three-tier Web infrastructure could be encapsulated so that its parameters would include pointers to virtual machine images for the Web server, business logic, and database tiers (Figure 8). This design pattern could then be executed to deploy standard applications without having to re-invent or even re-consider, for example, the network architecture required to support each tier.

The cloud computing philosophy for application maintenance is not to patch, but redeploy. Managing the model that created a virtual machine image, not the image itself, simplifies this redeployment. It's relatively easy to solve problems discovered after deployment, or release new versions of the application by updating the component virtual machines and invoking the design pattern to redeploy. When a developer patches a virtual machine, only one virtual machine image needs to be created — the rest should be replicated and deployed programmatically. Virtual machines should be versioned to facilitate rollback when necessary.

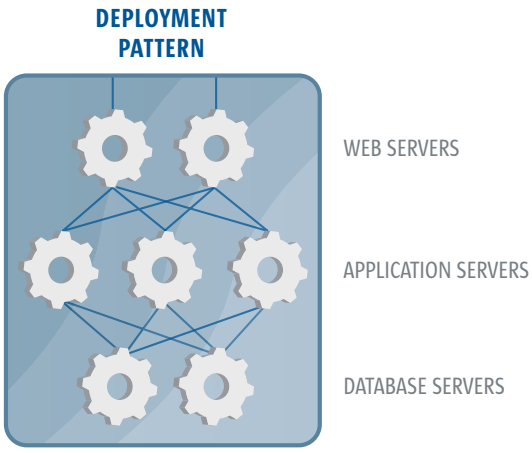

Figure 8. A deployment pattern can be encapsulated for re-use. In this example a pattern specifies Web, application, and database server tiers, and all that is needed for it to deploy an instance of itself is pointers to virtual machines for each of the three layers.

Loose-coupled, stateless, fail-in-place computing

For years, Web-based applications have been moving toward being loose-coupled and stateless. In cloud computing, these characteristics are even more important because of cloud computing's even more dynamic nature. Application images are not patched, they are throwaway objects and thus need to be stateless. If a virtual machine fails, the application must continue to run interrupted. Coupling between application components needs to be loose so that a failure of any component does not affect overall application availability. A component should be able to "fail in place" with little or no impact on the application.

As application components become increasingly transient, they cannot contain data that must persist beyond any application instance. Applications should be made as stateless as possible by pushing the state out of the software, separating processing and data as much as possible. Techniques for doing this include:

- Push state out to the user in the form of cookies or state coded into URLs
- Push state down to a back-end database
- Maintain additional copies of data, a strategy used by Hadoop
- Use network-based persistence, for example Terracotta or Shoal in a GlassFish application server

The impact that fail-in-place computing has on operations is that even the hardware should be stateless for the cloud to function properly. Hardware configurations should be stored in metadata so that configurations can be restored in the event of a failure.

Horizontal scaling

Cloud computing makes a massive amount of horizontal scalability available to applications that can take advantage of it. The trend toward designing and refactoring applications to work well in horizontally scaled environments means that an increasing number of applications are well suited to cloud computing.

OpenSolaris Dynamic Service Containers being produced by Project Kenai provide a lightweight provisioning system that can be used to horizontally scale Solaris Zones. Please see http://kenai.com/projects/dsc/.

Applications taking advantage of horizontal scaling should focus on overall application availability with the assumption that individual components will fail. Most cloud platforms are built on a virtual pool of server resources where, if any one physical server fails, the virtual machines that it was hosting are simply restarted on a different physical server. The combination of stateless and loose-coupled application components with horizontal scaling promotes a fail-in-place strategy that does not depend on the reliability of any one component.

Horizontal scaling does not have to be limited to a single cloud. Depending on the size and location of application data, “surge computing” can be used to extend a cloud’s capability to accommodate temporary increases in workload. In surge computing, an application running in a private cloud might recruit additional resources from a public cloud as the need arises.

Surge computing depends heavily on the amount and locality of data. In the case of a private cloud located in an enterprise datacenter expanding to use a public cloud located somewhere else on the Internet, the amount of data that needs to be moved onto the public cloud needs to be factored in to the equation (see “data physics” below). In the case of a private cloud hosted at the same colocation facility as a public cloud provider, the data locality issue is significantly diminished because virtually unlimited, free bandwidth can connect the two clouds.

Parallelization

Horizontal scaling and parallelization go hand in hand, however today the scale and implementation has changed. On a microscopic scale, software can use vertical scaling on symmetric multiprocessors to spawn multiple threads where parallelization can speed operations or increase response time. But with today’s compute environments shifting toward x86-architecture servers with two and four sockets, vertical scaling only has as much parallel processing capability as the server has cores (or as many cores have been purchased and allocated to a particular

virtual machine). On a macroscopic scale, software that can use parallelization across many servers can scale to thousands of servers, offering more scalability than was possible with symmetric multiprocessing.

In a physical world, parallelization is often implemented with load balancers or content switches that distribute incoming requests across a number of servers. In a cloud computing world, parallelization can be implemented with a load balancing appliance or a content switch that distributes incoming requests across a number of virtual machines. In both cases, applications can be designed to recruit additional resources to accommodate workload spikes.

The classic example of the parallelization with load balancing is a number of stateless Web servers all accessing the same data, where the incoming workload is distributed across the pool of servers (Figure 9).

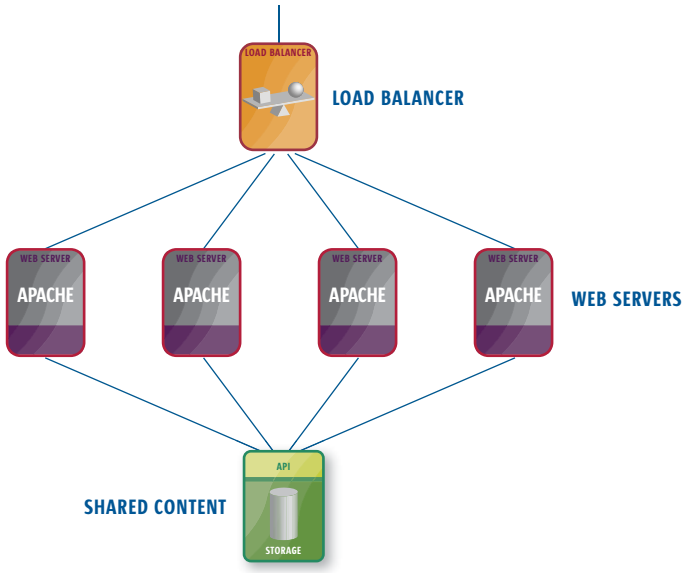

Figure 9. A very common use of parallelization and load balancing is horizontally scaled Web servers.

There are many other ways to use parallelization in cloud computing environments. An application that uses a significant amount of CPU time to process user data might use the model illustrated in Figure 10. A scheduler receives jobs from users, places the data into a repository, then starts a new virtual machine for each job, handing the virtual machine a token that allows it to retrieve the data from the repository. When the virtual machine has completed its task, it passes a token back to the scheduler that allows it to pass the completed project back to the user, and the virtual machine terminates.

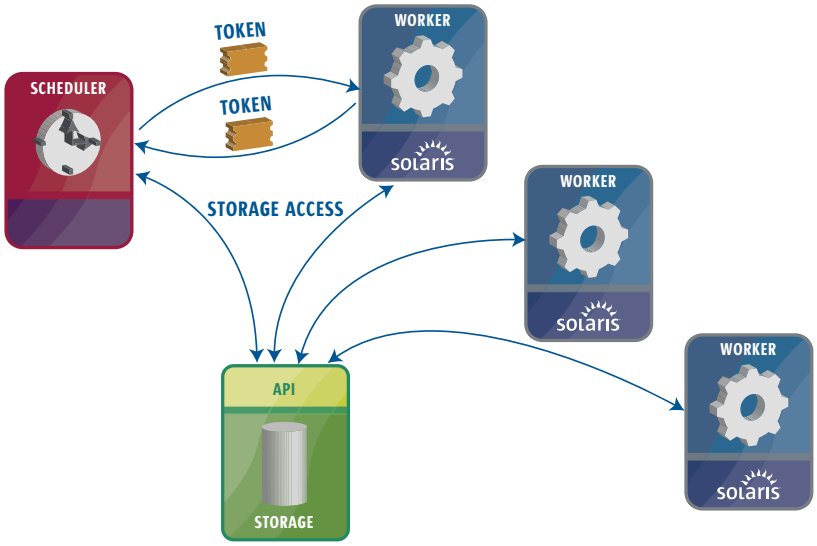

Figure 10. Another example of parallelization is the batch performance of resource-intensive tasks on behalf of users.

Divide and conquer

Applications can be parallelized only to the extent that their data can be partitioned so that independent systems can operate on it in parallel. A good application architecture includes a plan for dividing and conquering data, and a variety of real-world examples illustrate the wide range of approaches:

- Hadoop is an implementation of the MapReduce pattern which is an implementation of the master/worker parallelization pattern.
- Database sharding can be accomplished through a range of partitioning techniques, including vertical partitioning, range-based partitioning, or directory-based partitioning. The approach used depends entirely on how the data is to be used.
- Major financial institutions have refactored their fraud detection algorithms so that what was once more of a batch data-mining operation now runs on a large number of systems in parallel, providing real-time analysis of incoming data.
- Some high-performance computing applications that deal with three-dimensional data have been designed so that state of one cubic volume (of gas, liquid, or solid) can be calculated for time *t* by one process. Then the state of the one cube is passed onto the processes representing the eight adjoining cubes, and the state is calculated for time *t+1*.

The partitioning of data has a significant impact on the volume of data transferred over networks, making data physics the next in the list of considerations.

Data physics

Data physics considers the relationship between processing elements and the data on which they operate. Since most compute clouds store data in the cloud, not on a physical server's local disks, it takes time to bring data to a server to be processed. Data physics is governed by the simple equation that describes how long it takes to move an amount of data between where it is generated, stored, processed, and archived. Clouds are good at storing data, not necessarily at archiving it and destroying it on a predefined schedule. Large amounts of data, or low-bandwidth pipes, lengthen the time it takes to move data:

$$\text{time} = \frac{\text{bytes} * 8}{\text{bandwidth}}$$

This equation is relevant for both the moment-by-moment processing of data and for long-term planning. It can help determine whether it makes sense, for example, to implement a surge computing strategy where it might take longer to move the data to a public cloud than it does to process it. It can also help determine the cost of moving operations from one cloud provider to another: whatever data has accumulated in one cloud provider's datacenter must be moved to another, and this process may take time.

The cost of moving data can be expressed both in time and bandwidth charges. The hybrid model illustrated in Figure 5, where a company's private cloud is collocated with its cloud provider's public cloud, can help to reduce costs significantly. Bandwidth within a colocation facility generally is both plentiful and free, making this strategy a win-win proposition for both the time and dollar cost of moving data around.

The relationship between data and processing

Data physics is a reminder to consider the relationship between data and processing, and that moving data from storage to processing can take both time and money. Some aspects of this relationship to consider include:

- Data stored without compute power nearby has limited value, and cloud providers should be transparent regarding the network relationship between these two components. What is the size of their pipes? What is the latency? What is the reliability of the connection? Cloud providers should be forthcoming with answers to all of these questions.
- Cloud architects should be able to specify the locality of virtual components and services so that there is a well-defined relationship between virtual machines and the storage they access.
- Cloud providers may optimize this relationship automatically for customers, but consider whether their optimization is right for the application at hand.

- In a networked environment, it is sometimes more efficient (faster, less latency) to calculate a value than it is to retrieve it from networked storage. Consider the trade-off between using compute cycles and moving data around.

Programming strategies

Cloud computing calls for using programming strategies that consider the movement of data:

- Moving pointers around is usually better than moving the actual data. Note how the scheduler/worker model illustrated in Figure 10 uses a central storage service and passes tokens between application components rather than the actual data.
- Pointers should be treated as a capability, with care taken to ensure that they are difficult to forge.
- Tools such as representational state transfer (REST) or Simple Object Access Protocol (SOAP) help to reduce application state while managing the transfer of state data.

Compliance and data physics

Maintaining compliance with governmental regulations and industry requirements adds another layer of considerations to the management of data. A cloud architect needs to be able to specify both topological and geographical constraints on data storage. A cloud provider should make it easy to specify the relationship between data and the virtual machines that process it, and also where the data is stored physically:

- Companies handling personal data may be required to adhere to governmental regulations regarding the handling of the data. For example, those doing business in the European Union may violate local laws if they store their data in the United States because of the difference in how the law protects their data. In cases like this, cloud providers should provide the capability to specify constraints on how and where data can be moved.
- Companies subject to industry standards, such as those imposed by credit card processing authorities, may face restrictions on where data is stored and how and when it is destroyed. In cases like this, freed disk blocks cannot be allowed to be incorporated into another customer's block of storage. They must be securely erased before reuse.

When choosing a cloud provider for data storage, consider not just whether the provider is trustworthy. Consider whether the cloud provider is certified according to standards appropriate for the application.

Security and data physics

Data is often the most valuable of a company's assets, and it must be protected with as much vigilance than any other asset. It is easy to argue that more vigilance is needed to protect data because of how an intruder can potentially reach a company's data from anywhere on the Internet. Some steps to take include:

- Encrypt data at rest so that if any intruder is able to penetrate a cloud provider's security, or if a configuration error makes that data accessible to unauthorized parties, that the data cannot be interpreted.
- Encrypt data in transit. Assume that the data will pass over public infrastructure and could be observed by any party in between.
- Require strong authentication between application components so that data is transmitted only to known parties.
- Pay attention to cryptography and how algorithms are cracked and are replaced by new ones over time. For example, now that MD5 has been proven vulnerable to attack, use a stronger technique such as SHA-256.
- Manage who has access to the application and how:
  - Consider using strong, token-based authentication for administrator roles.
  - For customer login/password access, consider who manages the authentication server and whether it is under the company or the cloud provider's control.
  - For anonymous access to storage, for example anonymous FTP, consider whether a customer would have to register with the cloud provider for access or whether the cloud provider could federate with the company's authentication server.

Network security practices

Good security practices permeate every aspect of system design, implementation, and deployment. Applications must be secure by design, with interfaces that present only the appropriate data to authorized users. During implementation, developers must take care to avoid coding practices that could result in vulnerability to techniques such as buffer overflow or SQL injection. When deployed, operating systems should be hardened and every layer of software kept up to date with the most recent security patches.

In cloud computing, applications are deployed in a shared network environment, and very straightforward security techniques such as VLANs and port filtering are used to segment and protect various layers of an application deployment architecture as well as isolating customers from each other. Some approaches to network security include:

- Use security domains to group virtual machines together, and then control access to the domain through the cloud provider's port filtering capabilities. For example, create a security domain for front-end Web servers, open only the HTTP or HTTPS ports to the outside world, and filter traffic from the Web server security domain to the one containing back-end databases (Figure 11).

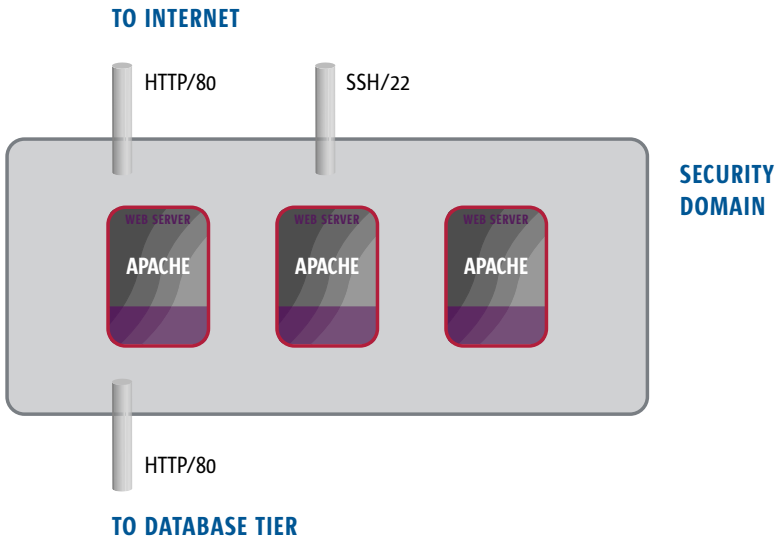

Figure 11. Cloud providers should offer mechanisms, such as security domains, to secure a group of virtual machines and control traffic flow in and out of the group.

For more information on ISCs, please visit: http://wikis.sun.com/display/ISC/Home/

- Control traffic using the cloud provider's port-based filtering, or utilize more stateful packet filtering by interposing content switches or firewall appliances where appropriate. For even more fine-grained control over traffic, the concept of Immutable Service Containers (ISCs) allow multiple layers of software to be deployed in a single virtual machine, with pre-plumbed networking that is kept internal to the virtual machine. This technology uses Solaris™ Zones to support multiple secure virtual environments on a shared OS platform, and is available with both the Solaris and OpenSolaris Operating Systems.

Chapter 4

Sun and Cloud Computing

For more than a decade, Sun has been advancing the state of the art of large-scale computing infrastructure that has formed the foundation of cloud computing. Beginning in the 1990s, Sun has been a leader in helping service providers implement their large-scale networks so that they can serve up to millions of customers. Large numbers of Sun servers are employed by financial institutions and stock exchanges to handle tasks ranging from handling transactions to real-time fraud detection. Sun has been a leader in high-performance computing by developing rack-at-a-time deployment models, automatic provisioning from the bare metal to the application, and large scale virtual networking with one petabit per second throughput. Indeed, Sun pioneered the cloud's predecessor, grid computing, by selling physical server time by the hour and helping customers implement internal grids to support their own operations.

Just as much as these large-scale computing capabilities help Sun develop cloud computing solutions, they are done in the context of advancing the systemic qualities that these solutions require: scalability, availability, reliability, manageability, and security.

Innovations from the Sun community

Sun has developed foundational technologies for cloud computing and has been a central player in the community development processes they have promoted. While Sun has long maintained the Solaris Operating System's industry leadership, it has also spawned a corresponding open source movement around the OpenSolaris Operating System. The MySQL database is the Web application database of choice, and the Java programming language powers Web sites and enterprise datacenters worldwide. The community-based, open source GlassFish application server provides a Java software execution container that has been extended to support Ruby applications and the Drupal content management system. OpenSolaris Project Crossbow has helped to expand the multi-tenancy support in Sun xVM hypervisor.

Beneath the rich, community-supported Software that Sun helps to foster comes the powerful server, storage, and networking products that make them perform — including standard, scalable x86-architecture servers, Sun's UltraSPARC® processor-powered server product line, and servers that incorporate Sun's energy-efficient, chip-multithreaded (CMT) UltraSPARC T1, T2, and T2 Plus processors. Sun's CMT processors process high-throughput workloads so efficiently that they are harnessed in content load balancing and application delivery products such as the Zeus Extensible Traffic Manager. Sun's Open Storage products combine open source software with industry-standard hardware to help reduce reliance on high-priced,

purpose-built systems. Indeed, Sun's ground-breaking Sun Fire X4500 Server helped the industry see the benefits of combining server and storage technology in the same system. Sun delivers virtual networking for large-scale computing through InfiniBand to massive-scale compute grids with the Sun Datacenter Switch 3456, scaling up to 13,834 nodes.

Community and open standards

Learn about how the community is defining the Sun cloud APIs by visiting http://kenai.com/projects/suncloudapis/

Working with a community breeds open standards-based products and helps to provide investment protection. In an emerging and rapidly changing market such as cloud computing, it's easy to create applications that are locked in to one vendor's cloud because of the use of proprietary APIs and formats. Using open standards and open source software is the best insurance that the applications you create today will still be useful tomorrow and will give you the needed flexibility to change cloud providers.

The open source communities with which Sun is involved develop to open standards where they exist, and establish new open standards as new products are developed. Open source, open standards, and open APIs lead to applications that have more portability and longevity. Sun's credentials in the open source community are impeccable, with projects including the: OpenSolaris OS, Linux OS, StarOffice™ software, NetBeans™ platform application framework, OpenSPARC™ technology, Java programming language, Sun xVM hypervisor, Sun xVM VirtualBox, Sun Open Storage Solutions, MySQL database management system, and the Solaris ZFS™ File System.

The importance of choice

Sun's hardware and software product line is synonymous with choice. Sun offers the choice of servers based on the x86 architecture, that are powered by powerful SPARC® and UltraSPARC processors, and those with CoolThreads™ technology. Sun offers all of these choices in form factors including rack-mount and blade systems, allowing customers a range of densities and I/O capacities to choose from. Sun offers virtualization solutions for every one of its server products, including support on its x86-architecture servers for Sun xVM hypervisor, VMware vSphere, and Microsoft Hyper-V. And of course your choice of operating system, including the Solaris OS, Linux, and Microsoft Windows.

Choosing a cloud computing provider

Sun innovations are the foundational technologies for cloud computing environments that are open, standards-based, and are the fruit of a community effort. Joining the Sun cloud computing community means having the choice of server, storage, and networking technologies that work at maximum scale. It means using software stacks, APIs, and standards that aren't owned by a cloud

provider, they're owned by the companies that build their cloud applications to have lasting value. Sun offers choice —not just in using the right hardware and software components to get the job done —but in leveraging cloud computing for the greatest benefit.

Learn about how Sun can help build your cloud at http://www.sun.com/cloud/.

Those joining the Sun community for cloud computing have a range of options. Sun can help organizations build their own private, local clouds as a way to transition enterprise datacenters toward this new computing model while retaining the utmost control over business-critical data. Sun can help companies build their own private, non-local clouds as a way to leverage the low cost of new large, energy-efficient colocation facilities such as Switch Communications' SuperNAP facility in Las Vegas, Nevada. Sun can help those wishing to become cloud providers with the hardware, software, and management capabilities that are required. And, starting now, organizations everywhere can augment their private clouds with Sun's public cloud offering —either by collocating with Sun at the SuperNAP site and enjoying the benefits of high-speed, local infrastructure, or by using Sun's services over the Internet. Whether you are looking to cloud computing for development and testing, experimenting with hosting applications in the cloud, offloading specific functions, or using the cloud for surge computing, Sun is in a unique position to help enterprises build and use cloud computing.

Acknowledgments

This paper was made possible through the efforts of Jason Carolan and Steve Gaede. Additional contributors include James Baty, Glenn Brunette, Art Licht, Jim Remmell, Lew Tucker, and Joel Weise. Thanks also to Benoit Chaffanjon, David Douglas, Mikael Lofstrand, Ken Pepple, Scott Mattoon, and John Stanford for their review comments.

This page intentionally left blank.

This page intentionally left blank.

Sun Microsystems, Inc. 4150 Network Circle, Santa Clara, CA 95054 USA Phone 1-650-960-1300 or 1-800-555-9SUN (9786) Web sun.com

SunWIN #564162 Lit. #GNWP14947-0 06/09