Cloud computing – key concepts

Overview

Some organizations are moving to cloud computing but many are learning the cloud computing model, understanding pros and cons, doing ROI analysis and evaluating various cloud providers. This paper is meant for people who are in evaluation mode, they could be head of engineering, head of operations or somebody who is influencing or deciding which cloud technology to adopt. We will explain important cloud computing concepts in the context of criteria that should be used for evaluation. The criteria are:

- Flexibility: This is the capability to have control over the hardware, software or other parts of the cloud.
- Engineering Tools: There are many languages, IDEs, frameworks for developing cloud applications. How do you select the right tools?
- Manageability: This includes deployment, monitoring and repair of applications.
- Security: This is about keeping your application and data secure and private.
- Cost: What is the cost of moving to cloud model? How do you estimate it?
- Application Infrastructure services: This is about services such as auto scaling, caching and high availability which are needed to build world class applications
- Performance: How do you ensure that your application performs well in the cloud?
- Ecosystem and mindshare: How broadly liked and adopted is a certain cloud solution?

We will provide examples of technology and services provided by vendors for each of the criteria. We will provide specific data gathered as of Fall 2011 to have a meaningful dialog with the reader. But cloud computing is a very fast changing world. The reader is encouraged to look at the vendor's site to get the latest data.

Flexibility

An organization should decide how much control they need on the application, the underlying languages and services used to develop the application or the underlying hardware infrastructure on which the services and languages are running. There are basically 3 service models of cloud computing:

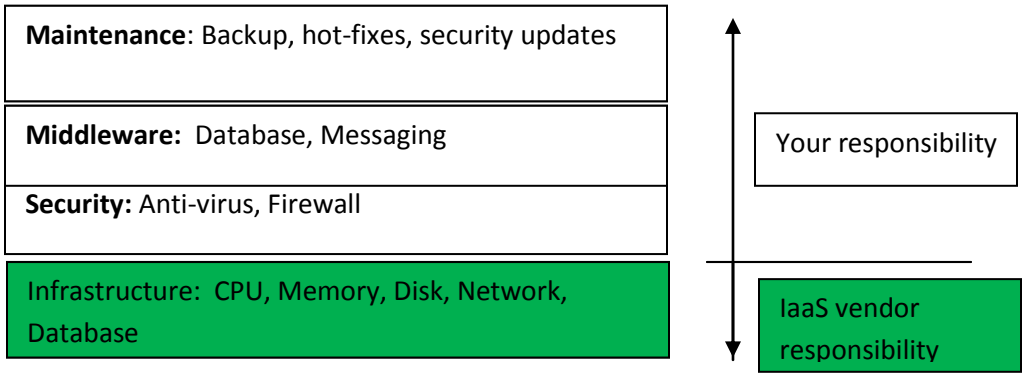
SaaS: In this model the capability provided is to simply run the application. You can access the application through a browser. You may be able to change some settings relevant to the application, but other than that you cannot choose operating system, services or the hardware. Some examples are Google's gmail, Salesforce CRM.

PaaS: In this model, you can build an application using the frameworks, languages and services supported by the provider. You cannot control the underlying hardware. Some examples are Google App Engine (GAE), Vmware CloudFoundry.com and Heroku.

IaaS: In this model, you can choose type of processor, amount of RAM, storage and operating systems. You might have control of select network hardware too. Some

examples are Amazon AWS, Vmware and Microsoft Azure.

As you go from SaaS to PaaS to IaaS, you have more flexibility and control. But it also means you have to know how to make the correct choices on software and hardware. See the diagram below to understand your responsibility if you go with IaaS. You are responsible for all the boxes other than the Infrastructure box. If you do not want to take on so much responsibility then you can go with a PaaS service model.



In IaaS model, you can combine resources from different vendors to build your own stack of components or you can get all components from one vendor such as Microsoft to build a homogeneous stack. In a heterogeneous system there is more choice since you can pick components from different vendors or open source community. But it may be difficult to manage such a system. You may not get comprehensive tools to monitor and manage a heterogeneous system. In a homogeneous system, manageability will be easier, and different components may be tuned well to work with each other having the potential of higher performance.

Some vendors have great offerings that reduce your management and operations overhead. But this requires you to choose their development patterns and APIs. You have to consider this carefully because you could get locked in with a specific vendor. This is an operational long term lock-in. This means that you are impacted by the vendor hardware/software's reliability, security and performance, as well as the vendor's business viability. You are also at risk of price increases, for example Google significantly increased the price of its App Engine in Fall 2011. In order to address the issue of lock-in some innovative companies have created different solutions. Orangescape is a cross platform PaaS solution. You can develop applications in their development studio and deploy to different cloud providers (such as Amazon, Google, IBM, Microsoft) and on premise. Another company RightScale lets you manage cloud resources of multiple providers such as Amazon, Rackspace, Eucalyptus, CloudStack and others. There is less lock-in with IaaS service model because in this model you can choose the type of hardware, operating system, database and other things. But in PaaS or SaaS there is more locking because they are built upon an underlying IaaS or PaaS respectively. If you are going with IaaS model, you can ask the provider about their system architecture; understand how they are taking care of single points of failure and other reliability issues before you decide to bet on

a provider. Remember that even providers such as Amazon, Google and Microsoft have had outages which have impacted the services running on their platforms.

Engineering Tools

Choosing a language and framework is a strategic decision that can make the difference between success and failure. Over the last few years there is a huge proliferation of tools for building applications and services. :

- Languages: There are languages such as C, C++, Java and loosely typed script languages such as Javascript, Python, Ruby, PHP and others.
- IDEs: For open source platforms - Eclipse, Netbeans, Codelite, MonkeyStudio just to name a few. For Microsoft stack, Visual Studio and others.
- Build tools: Maven, Ant/Ivy, Gradle, Buildr and many more tools.
- Frameworks for building web applications: A framework is a collection of packages which enable developers to write applications or services without having to deal with details such as protocols, sockets or process/thread management. Ruby has Rails, Java has node.JS, Python has Django, Java has Spring, Groovy has Grails. There are many other frameworks for these languages, I'm just listing a few for brevity.

There are following business considerations in choosing tools:

1. Productivity: Your developers, testers and operations folks need to be highly productive. If a certain tool enables them to develop application faster than another tool then it gives your business a unique advantage.
2. Talent availability: Certain tools/platforms are very popular but it may be difficult to find talent for them.

There is an index that tracks the usage of languages. It is called the Tiobe index and you can view it at http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html . Java, C and C++ are the top 3 languages. Python showed the most gain in 2010.

There are a number of engineering considerations in choosing tools:

1. Functionality: You have to look at the functionality and features offered by the tool. Is it comprehensive? Is the tool high quality? If you are evaluating build tools you would look at versioning, dependency management, reporting, ability to run quality check tools, language support, cross platform support and integration with other tools such as IDE. You also have to see the suitability of tool for the type of application. E.g. C language wouldn't be a good choice for a web application.
2. Stability: Some popular tools in open source community evolve very fast. Many developers are contributing code and plugins to it. If you are using these tools you have to stay up to date and sometimes the latest updates can be unstable.
3. Integration: Choose a toolset that offers integration between various parts of the development workflow. If you are working on Amazon AWS and use Eclipse, you can download AWS's Eclipse plugin. Once you download and install it, you can write, build, and do simple deployments all from eclipse.

Manageability

Manageability includes deployment, monitoring and repair of applications deployed in the cloud. An application consists of 3 things:

- Software: This is the actual code for your application and all its features
- Configuration: These are settings that control certain aspects of the software such as boost performance under certain conditions or increase logging to debug failures.
- Data: This is the information that your application uses or exposes to customers. For example, if your application is providing stock quotes, then all the companies and their stock prices is the data.

A broadly used cloud application cannot be directly deployed from developer's box into production otherwise it can take the service down. You will have to create multiple environments: development -> test ->staging ->production. The application will have to be verified in each environment before it goes to the next stage. But all of this can seem like it is slowing you down. Hence these design principles need to be followed:

- Data should not change the behavior of the application.
- Configuration can cause only minimal behavior changes to your application
- Application should use new settings and data without having to restart the application.

If you follow these principles then configuration and data can be deployed fast and with minimal risk of service outage. You can also deploy it much faster than software.

Check functionality provided by vendor to setup environments and tear them down. You need a quality exit criteria at each stage, check if the vendor has tools to automatically deploy, check if quality criteria is met and then move the application to next stage. This will save a lot of human effort in deployment. If you are a broadly used application you will need many production environments in multiple data centers. You will have to stage the application through one or more of these environments to reduce service outage risk to your customers. While your normal deployment path is staged, you need a path for rapid deployment to all production environments to handle emergency. For more advice on deployment please consider consultation with Pramak. If you are in the IaaS service model, you will have to deploy middleware, anti-virus, firewall, security updates and general configuration settings. Check about tools to deploy all these pieces of software. Deploying OS and middleware software updates is tricky, they may destabilize the system. A benefit v/s risk analysis needs to be done for these updates. If you decide to deploy them, you will have to verify these updates in your test or staging environment before deploying to production.

Certain metrics are important in deployment. How long does it take to provision an instance? How long does it take to stop it? How long does it take to deploy a critical update to all my compute instances in production worldwide? The table below shows a measurement for provisioning Windows and Linux image in AWS

Operating	Method	Time
Windows	Create from image	10-15 minutes
Linux	Create from image	5-10 minutes
Windows	Revive stopped instance	30 seconds
Linux	Revive stopped instance	30 seconds

Source: http://slideshare.net, "10 Things you didn't know about cloud Platforms: AWS, GAE, Azure" by Dr. Anna Liu, Dr. Hiroshi Wada, Kevin Lee. National ICT Australia, 2010

It takes a while to start an EC2 instance. Hence it is better to provision an instance in advance and stop it and then just start it. You won't pay instance cost for stopped instance but you will pay storage cost.

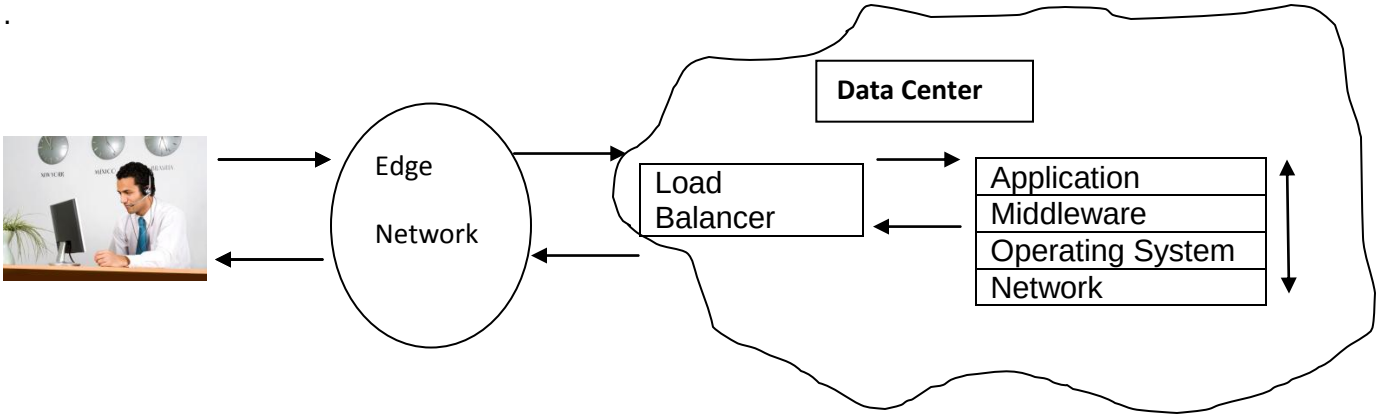
Monitoring

Following are the business requirements of monitoring:

- Ensure that Service Level Agreements (SLAs) that were promised to customers are met and there is no service degradation.
- Observe key metrics deep in the software, hardware or network to look for anomalies, that if not addressed could lead to SLA violation.
- Ensure there is optimal utilization of resources if the IaaS service model is being used, so that your operational expenses are within budget
- Watch security attacks against the application.

The above requirements are universal across any type of business running in any cloud. Cloud applications are running on complex distributed systems. A call from end user goes through an edge network into the load balancers in the data center. The load balancers forward the call to your application. Depending on the type of application it may use database or other middleware and operating system functionality. The picture below is a simple illustration. You could have a database from Oracle, Linux operating system and the routers could be from Cisco. Further your application may be integrated with some other application such as a directory or messaging solution running in the same or other cloud.

.



The following criteria should be considered in evaluating monitoring tools provided by cloud vendors or 3rd parties:

- Timely identification of issue and as close to source as possible. Knowing a failure at the top level is not enough. If the failure is due to some underlying failure deep down in the system, then knowing that failure and having a co-relation is very useful to quickly debug the problem.
- Data collection of failures across all levels from application down to the network within and across environments.
- Tools for analysis and visualization of data.
- Ability to take actions based on data. Actions could be scaling out, throttling and others.

AWS provides a Cloudwatch monitoring service. It provides metrics for the resources that the application is using. For example for Compute Instance it shows CPU utilization, disk usage, network bytes sent/received, for database it shows read/write throughput, # of connections, free storage and others. You can track 5-7 metrics at 5 minute intervals. The data is stored for 2 weeks. If you need to track more metrics, or track them at more frequent intervals then you need to pay for it. AWS also provides command line tools and API for monitoring. Azure has an API for monitoring and Microsoft has released a System Center Monitoring pack for Azure applications. It requires System Center product. There are 3rd party tools such as Azure Diagnostics Manager (ADM) for monitoring Azure.

Security

A key concern for businesses is whether their app and data are secure. Any cloud provider must provide confidentiality, integrity and availability of customer's data. Security is impacted by the type of deployment model you choose, physical and network security, system architectures and other areas. We will cover this in more detail:

Deployment model: There are 3 deployment models for the cloud: Private cloud, Public cloud and Hybrid cloud. In a Private cloud the infrastructure is provisioned for the use of a single organization. No other organization's application or data co-mingles with it. In a Public cloud, infrastructure is provisioned for the use of general public. Your application may be running on the same physical machine as some other application from a different company. In a Hybrid cloud, there are 2 or more distinct cloud infrastructures but they are bound by some technology such as cloud bursting to load balance between Private and Public cloud. Private cloud is more secure than Hybrid or Public cloud because there is physical isolation between applications.

Physical Security: Have you ever visited a data center? If not, you should try to visit. It is impressive to see huge number of machines, power equipment, cooling structures and diesel generators. What should physical security be like? First off, as you approach the area you shouldn't even feel that you are coming to a building that is housing a data center. There needs to perimeter security as you enter the campus, you need video surveillance. A visitor should not be able to enter any of the rooms without an escort. An employee should

have multiple levels of authentication before they enter and they should only have access to areas where they are supposed to work. We were able to see a Microsoft data center because of our previous relationship with Microsoft. It had all these attributes. Vendors publish details about their data center security. AWS says that their data centers are in non-descript facilities. They have access control both at perimeter and building ingres points with video surveillance, intrusion detection systems and other electronic means. They have two factor authentication at least twice for access to datacenter floors. Amazon has been operating data centers for many years for their retail business. They have applied their learnings to AWS data centers. One can presume that their physical security is at least as good as if not better than most cloud providers.

Network Security: Cloud applications are vulnerable to denial of service, man in the middle and other attacks. Amazon has developed proprietary techniques to prevent a DDOS or MITM attack. Amazon EC2 instances cannot send spoofed network traffic. Also tenants cannot sniff traffic of other tenant's on the same physical host. In the case of Azure, the hypervisor and the root OS provide network packet filters that assure that the untrusted VMs cannot generate spoofed traffic, cannot receive traffic not addressed to them, cannot direct traffic to protected infrastructure endpoints, and cannot send or receive inappropriate broadcast traffic.

Architecture, design and API: AWS has customized the Xen hypervisor for its needs. It takes advantage of paravirtualization in case of Linux guests. The hypervisor runs in ring 0 and guest operating systems run in ring 3. The firewall resides within the hypervisor between the NIC and the VM instance's interface. All packets must pass through this layer, hence a VM instance's neighbors have no more access to that instance than any other host on the Internet and can be thought of as being on separate physical hosts. The physical RAM is separated using similar mechanism. AWS's disk virtualization software automatically resets every block of storage used by the VM instance, so that one instance's data is not exposed to another. Azure provides strong isolation of the root VM from the guest VMs and the guest VMs from one another; all managed by the hypervisor and the root OS. Further, their fabric controllers are strongly isolated from fabric agents running within customer applications by controlling the initiation and direction of communication and using SSL.
If you are communicating with cloud provider using an API, there should be authentication based on public/private key pair and certificate. Azure's service management API (SMAPI) authentication is based on user generated public / private key pair and self-signed certificate that you can generate through the Azure portal.

Software development process: AWS engineering process includes formal design reviews by their Security Team, threat modeling and risk assesment. They run static analysis tools during build process. They also get industry experts to do recurring penetration testing on all deployed software.

Certifications and accreditation: There are various industry certifications for security:

- ISO27001: This standard sets out requirements and best practices for a systematic approach to managing company and customer information; that's based on periodic assessments of risks such as confidentiality, integrity and availability of customer

data. AWS has achieved ISO 27001 certification covering its data centers and services including EC2, S3 and VPC. Microsoft says that portions of their Azure data centers are ISO27001 certified as of Dec 2011.

- SOC1 report: Amazon has published a Service Organization Controls 1 (SOC 1) Type 2 report. It attests that AWS's control objectives are appropriately designed and that the individual controls designed to safeguard customer data are operating effectively. The audit for this report is conducted in accordance with the Statement on Standards for Attestation Engagements No. 16 (SSAE 16) and the International Standards for Assurance Engagements No. 3402 (ISAE 3402) professional standards.
- FISMA accreditation: The Federal Information Security Management Act (FISMA) authorization and accreditation requires a vendor to document the management, operational and technical processes to secure the physical and virtual infrastructure and the 3rd party audit of the established processes and controls. AWS has received Moderate accreditation for EC2, S3 and VPC.
- PCI standard: AWS has achieved Payment Card Industry (PCI) Data Security Standard (DSS) Level 1 service provider status. This is for EC2, S3, EBS and VPC.
- ITAR and FIPS: The AWS GovCloud supports US International Traffic in Arms (ITAR) compliance. In addition to support customers with FIPS 140-2 requirements, the Amazon VPN endpoints and SSL-terminating load balancers in GovCloud operate on FIPS 140-2 compliant hardware.
- Safe Harbor: Microsoft and Amazon are signatories to Safe Harbor framework for data privacy.

Cost model

Moving to the cloud has implications on your cost. Following considerations:

Development cost: This is the cost to develop a new application or to modify and migrate an existing application. In general cost to develop a cloud application should be less than the cost to develop a regular client/server application. The overhead incurred by a development house in setting up source control, build tools, pre-deployment testing is reduced in the cloud world if the development shop uses the right cloud engineering tools. But you have to ensure that your application has the right architecture and design. An inefficient use of resources won't impact your development cost but can significantly increase the running cost of your application.

Resource costs: This is the cost for all resources that you consume: compute, storage, caching and others. Cloud providers offer different pricing models for their resources:

- On-demand: This is a pay as you go model based on hourly usage of capacity. You sign up with a credit card. At the end of the month the vendor will charge your credit card for the services and capacity that you used.
- Reservation: In this model, you pre-reserve capacity. You pay an upfront fee and a lower per hour fee than the On-demand model.
- Spot: This is an interesting model wherein your application can "bid" a certain price. If the cloud provider's spot price goes below your "bid" price then you get the

capacity, otherwise you don't. The spot price varies based on supply and demand. Amazon AWS offers this feature. Note that if the spot price of the resource increases above your bid, then AWS automatically terminates whatever is running on that resource. If your application is designed to handle resources coming and going, then you can avail of this variable pricing. Spot pricing can lower your cost significantly. You can consult Pramak to design applications and services that can use Spot resource allocation methodology.

Besides hourly fees, certain resources such as databases may have additional fees based on amount of data transferred or number of connections. Many cloud providers offer free trials for their services for 1-3months. But some providers will have a free tier if your resource utilization falls below certain thresholds. This isn't just a teaser trial but it can go for 1 year for new customers.

The table below shows cost of compute for 3 vendors

SKU Name (AWS, Azure)	**CPU cores**	**Memory (AWS, Azure, Vmware) GB unless specified**	**Diskspace (AWS, Azure, Vmware) GB**	**Windows Price (AWS, Azure, Vmware) cents/hour**	**Linux Price (AWS, Vmware) cents/ hour**
Micro, ExtraSmall	Shared	613MB, 768 MB	Not specified, 20	3, 4	2
Standard Small, Small	1	1.7, 1.75, 1	160, 165, 50	12, 12, 9	8.5, 7
None, Medium	2	3.5, 3.5, 2	340, 340, 50	N/A, 24, 17	N/A, 13
Standard Large, Large	4	7.5, 7, 4	850, 850, 50	48, 48, 34	34, 26
Standard Extra Large, ExtraLarge	8	15 , 14, 16	1690, 1890, 50	96, 96, 112	68, 96

(Note: VmWare pricing is for their partner VirtaCore. Table data as of Fall 2011)

There are a couple of things to notice from this table:

- First, price for Windows at Microsoft, Amazon and Vmware/Virtacore is very similar. This means that price is not a distinguishing factor if you want to go with Windows SKUs. You will have to consider other factors that we discussed in the paper.
- Second, AWS doesn't have any compute SKU at the 25c/hour price point for Windows. (Note this table doesn't contain Double Extra Large, Quadruple Extra Large SKUs since there isn't anything comparable offered by Azure.

The table was meant to give you a flavor for cost variation per hour, amongst some vendors. But the cost of resources should be estimated over a period of time such as 1-3 years. You will have to estimate the load on the service during normal and peak times of the day. You will have to also think of periods of abnormal activity during the year when your load changes. For example, if you are a retailer in the US, then the load during Thanksgiving or other special holidays could be an order of magnitude more than normal load. Based on these factors you can plan resources and costs. Even though cloud computing offers elasticity and can handle your load, it won't do it for free. You will have to model your resource usage to understand your costs.

Management costs

In order to operate a world class service you need to do excellent management and monitoring of service:

- Tools costs: You may need 3rd party management tools. You may also need tools for backup, disaster recovery and maintenance and software updates.
- People costs: This is the people cost of operating a service. Some of the work is done by tools but you also need to hire some operations folks. You should strive to build a high quality service which doesn't need many people for managing the operations. If there is a service outage, you should be able to recover quickly. There should be great diagnostics built into the system to troubleshoot, debug and get to mitigation fast. The more automated the work-flows for management, monitoring and alerting, the less people cost you incur.

Application Infrastructure services

A scalable high performance application needs the help of many services. We refer to these services as application infrastructure services. We will discuss these services by looking at how an application evolves from a basic application to a high performance scalable application in an IaaS environment

In the first incarnation of a cloud app, there is a front end, a server and some storage.

- On Amazon you need to allocate 1 EC2 instance, an elastic IP and flat file storage (S3) or structured storage (SimpleDB).
- On Azure the counterpart to EC2 is Azure Compute. You will have to run 1 VM instance in a *worker* role. The storage equivalent to S3 and SimpleDB are blobs and tables. This is where you will store logs and static data. In both AWS and Azure you will have to create an endpoint to access the application. Now your application becomes popular and it has to scale.

In the second incarnation of the cloud app you add autoscaling.

- On Amazon you get autoscaling through the CloudWatch service. You need to download tools and set policies for ramping up/down resources based on criteria such as CPU utilization, network activity or disk utilization.

- On Azure you automatically get a scaling group and an elastic load balancer. You have to do is update the instance count in Azure management portal.
  Vendors tout autoscaling like it is something magic. But in reality there is work involved in making it happen. The vendor's monitoring system has to support the ability to do monitoring at a very fine granularity. It is too late to react to a traffic spike when it has already happened. The vendor also has to make sure that certain other components of the system such as load balancers themselves don't become bottlenecks. Assuming these things are taken care of, now you find that there you are getting customers from different parts of the world and some are seeing slow performance.

In the third incarnation of the cloud app you add content caching for static and streaming content. These edge caches can deliver your content with minimal latency and maximum throughput.

- On Amazon, you enable the CloudFront service. Amazon has a global network of edge servers. There are no upfront commitments; you only pay for as much data as is sent through the cache. Amazon says they have 20,000 customers using CloudFront as of Dec 2011.
- On Azure they have a Content Delivery Network (CDN) which exists in US, Europe, Asia, Australia and South America. They also have a pay as you go plan. Next you find that your backend database is not meeting the increased demands on your application.

In the fourth revision of the cloud app you add a relational database instead of using SimpleDB/Tables. (By now, we feel like calling the changes as revisions rather than incarnations)

- On Amazon you can add Relational Database Service (RDS) which gives you access to mySQL or Oracle database. RDS provides management functionality such as automatic patching, backup and replication. There are six instance types ranging from 1 ECU, 1.7GB RAM to 26 ECU and 68 GB RAM. For every instance you can have 5GB to 1TB storage capacity.
- On Azure you enable the SQL Azure service. SQL Azure is also a managed service which includes patching, backup and replication. You can setup anywhere from 5GB to 150GB database. Your app now has become a huge hit. Customers worldwide are using it, you cannot afford to have a service outage.

In the fifth and final revision of the application you make it highly available.

- On Amazon you have a couple of options: Amazon has regions and availability zones. Regions are geographic locations that have data centers such as US-East coast (Virginia) or US-West coast (Oregon). Each region has Availability zones. Availability Zones are distinct locations that are engineered to be insulated from failures in other Availability Zones and provide inexpensive, low latency network connectivity to other Availability Zones in the same Region. If you run your service in more than one availability zone you reduce the risk of site outage due to failures happening in that zone such as local network equipment in that zone. But it doesn't protect you if an entire region is out. In this case you need your service to run in multiple regions. Amazon has a SLA at a region level of 99.95% availability for compute. Hence if your service is running in 2 regions you effective availability is even higher.

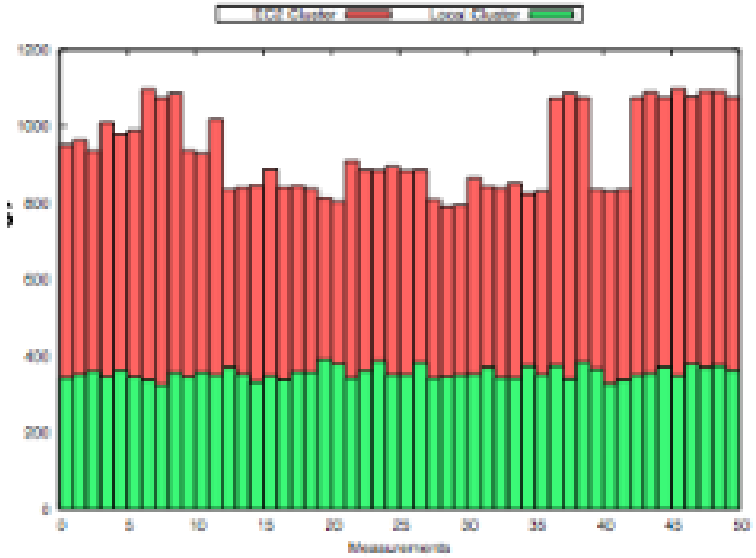
- On Azure, your compute, storage and database resources are distributed across the data center to prevent any single failure from taking out all resources. Azure has an overall SLA of 99.95% for compute, not specific to a region.

This description gave you an overview of the basic application infrastructure services. There are many other application infrastructure services such as messaging, notification, DNS and others. For a detail analysis of these application infrastructure services consider a consulting engagement with Pramak.

Performance

Moving to the cloud reduces capital expenses, decreases manageability overhead but it should not impact performance. How do you ensure that performance doesn't degrade and your customers don't suffer? We suggest the following things:

- Resources: The type of processor and clock speed, amount of RAM, network bandwidth will have huge impact on performance. If you are migrating to cloud ensure that you are provisioning resources similar to on premise resources.
- Deployment model: If you are in a private cloud, you don't have any other tenants and hence your performance will be better than a hybrid cloud shared tenant model, assuming everything else is identical. If you are in a hybrid or public cloud then you should provision more powerful resources than on premise, because you are only going to get a slice of the resource.
- Benchmarks: You can develop or acquire benchmarks for CPU, memory, disk and network. You can run these benchmarks in multiple vendor data centers while selecting the same type of resources and tenancy model. This will give you data to do a comparison.
- Workloads: Last but not the least is to carve out a portion of your application into a standalone workload. Run this workload in premise and in multiple vendor clouds and do a comparison. Here is some sample data that shows a performance comparison for a map-reduce job for a 50 node local cluster and a similar cluster in AWS.



Source: http://slideshare.net, "10 Things you didn't know about cloud Platforms: AWS, GAE, Azure" by Dr. Anna Liu, Dr. Hiroshi Wada, Kevin Lee. National ICT Australia, 2010

As you can see there is huge variation in performance for EC2 cluster. This will impact the predictability of your application or service.

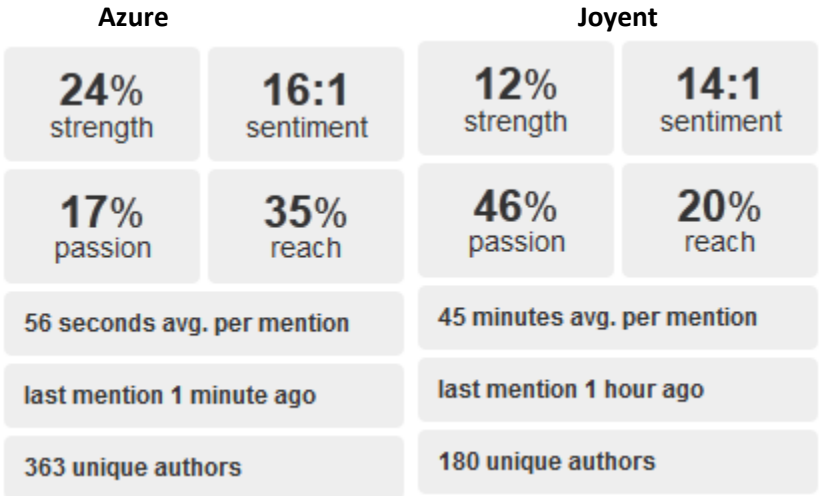
- Service choices: Your other service choices will have impact on performance. For example if you need higher security and go through additional firewall, do encryption, decryption, then this will impact performance.

Ecosystem and Mindshare

The ecosystem of apps and tools is very important for a IaaS and PaaS product to thrive in the marketplace. The more mindshare, the more apps and tools exist, the more things can be done with that IaaS and PaaS product. The analogy is an operating system, the more applications exists, the more useful it gets. If there aren't many applications then the operating system by itself is not of much use to end users. The following factors need to be considered for IaaS/PaaS:

- Number of applications, services, plugins that exist today and growth rate. Ranking/reviews of these products. Salesforce.com Appexchange has a nice review system for apps running on its platform.
- Number of vendors working on it today and growth rate
- Number of developers, consultants working on it
- Sentiment expressed by developers and consultants in newsgroups, blogs and tweets
- Number of open jobs

Take a look at chart showing comparison between two providers Azure and Joyent.



(Data gathered using Socialmention monitoring tool in Fall 2011)

You can see that Azure has more strength, positive sentiment and reach. But Joyent has more passion. Strength is the likelihood that the product is discussed in social media. Sentiment is the ratio of mentions that are generally positive to ones that are generally negative. Reach is a measure of range of influence. Passion is the likelihood that individuals talking about a product will do so repeatedly. This table implies that Joyent has a smaller group of followers but they are passionate about it. This likely could be because their cloud solution excels in certain niche areas where they have a strong following. This is a very high level comparison. A more detail comparison for the factors listed above can and should be done before making adoption decisions.

Summary

This paper was meant to give you an overview of the key concepts of cloud computing and various factors involved in deciding your cloud platform. It gave you examples of features that different vendors offer and some comparison. For a more comprehensive understanding of technologies beyond the scope of this paper or to develop your product or technical strategy for cloud, consider consulting engagement with Pramak, LLC.

For more information about Pramak, please visit www.pramak.com