Cloud Models/Architecture

- Three cloud service models
  1. Infrastructure as a Service (IaaS)
  2. Platform as a Service (PaaS)
  3. Software as a Service (SaaS)
- Provide a level of abstraction to reduce the effort required by consumer to build and deploy systems
- Cloud stack

<table>
<thead>
<tr>
<th>Service Models</th>
<th>Cloud Stack</th>
<th>Stack Components</th>
<th>Who is Responsible</th>
</tr>
</thead>
<tbody>
<tr>
<td>IaaS</td>
<td>User</td>
<td>Login, Registration, Administration</td>
<td>Customer</td>
</tr>
<tr>
<td></td>
<td>Application</td>
<td>Authentication, Authorization, User Interface, Transactions, Reports, Dashboard</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Application Stack</td>
<td>OS, Prog Lang, App Server, Middleware, Database, Monitoring</td>
<td>Vendor</td>
</tr>
<tr>
<td></td>
<td>Infrastructure</td>
<td>Data Center, Disk Storage, Servers, Firewall, Network, Load Balancer</td>
<td></td>
</tr>
</tbody>
</table>

- Bottom layer
  * Traditional data center
  * Possibly some virtualization

Infrastructure as a Service (IaaS)

- Physical assets: servers, network devices, storage disks
- Traditional model requires IT team to build and support all infrastructure requirement for the organization
  - Install servers
  - Develop/install appropriate software
  - Ensure security
  - Update security and software by applying patches
- Cloud service model provides levels of abstraction and automation for those tasks
  - Put together infrastructure demanded by user, including servers, storage, networks, and data center
  - User can deploy and run on multiple VMs, running guest OSes for specific applications
  - User does not manage the underlying cloud infrastructure but can specify when to request/release a needed resource
- NIST definition of IaaS
  - Capability provided to consumer to enable
Cloud Models/Architecture

- Processing
- Storage
- Networks
- Other fundamental computing resources
  - Consumer should be able to deploy and run arbitrary software, including OS and applications
  - Consumer does not manage or control underlying cloud infrastructure
  - Consumer has control over OS, storage, and deployed applications, and possibly limited control over networking components (host firewalls)

- Cloud Security Alliance (CSA) model of IaaS
  - Delivers computer infrastructure (platform virtualization environment) as a service, plus raw storage and networking

- Tasks for physical data center and infrastructure are abstracted and available as a collection of services
  - Services accessed from web-based management consoles
  - Developers design and code entire applications
  - Admins install and manage/patch the developed applications
  - No physical infrastructure to manage
    * No procurement cycle to evaluate and purchase physical hardware
    * No need for physical data center to host hardware
  - Virtual infrastructure available as a metered service with pay-as-you-use model
  - Consumers focus on application development and deployment rather than managing data center and infrastructure

- Major IaaS providers include Amazon AWS, Windows Azure, Google Compute Engine, Rackspace Open Cloud, IBM SmartCloud Enterprise, and HP Enterprise Converged Infrastructure

Platform as a Service (PaaS)

- Application infrastructure
  - Access to OS and associated services
  - Way to deploy applications to the cloud

- PaaS sits on top of IaaS as the next level of abstraction
  - Enables the user to deploy user-built applications on a virtualized cloud platform
  - Handles platform-level services such as caching, asynchronous messaging, and database scaling
  - Includes middleware, databases, development tools, and some runtime support such as Web 2.0 and Java
  - Platform includes both hardware and software integrated with specific programming interfaces
  - Allows developers to focus on business logic and not worry about underlying IT plumbing

- NIST definition of PaaS
  - Capability provided to consumer to
    * Deploy onto cloud consumer-created or acquired applications created using software utilities supported by the provider
  - Consumer does not manage the underlying cloud infrastructure, including networks, servers, OS or storage
  - Consumer controls deployed applications, and possibly configuration settings for the application-hosting environment
Cloud Models/Architecture

- CSA model of PaaS
  - Delivery of computing platform and solution stack as a service
  - Facilitate deployment of applications without the cost and complexity of buying underlying hardware/software
  - Services available entirely from Internet
- PaaS vendors
  - Manage the application platform
  - Provide developers with tools for development
  - Controls the amount of computing power available to developer or consumer
    * May throttle the amount of compute power to a service customer to ensure that the platform scales equally for everyone
- Developers
  - Constrained by tools provided by vendor
  - Have no control over lower-level software controls such as memory and thread allocation, or amount of cache
- Major PaaS providers include Engine Yard, Red Hat OpenShift, Google App Engine, Heroku, AppFog, Windows Azure Cloud Services, Amazon AWS, and Caspio

Software as a Service (SaaS)

- Application execution, provided on demand to user
- Consumer configures application-specific parameters and manages users
- Browser-initiated application software
- Common applications include customer relationship management (CRM), enterprise resource planning (ERP), payroll, and accounting
- Useful for non-core functions
  - No need to support application infrastructure
  - No need to provide maintenance
  - No requirement to hire staff to manage it
- On customer side, no upfront investment in servers or software licensing
- On provider side, low costs compared to conventional hosting of user applications
- NIST definition of SaaS
  - Capability provided to consumer
    * To use provider’s applications running on a cloud infrastructure
    * Applications accessible from various client devices through a thin client interface – web browser or a program
  - Consumer does not manage or control underlying cloud infrastructure (network, servers, OS, storage), or even individual application capabilities except limited user-specific application configuration settings
- Major SaaS vendors include Abiquo, AccelOps, Akamai, AppDynamics, MeghaWare, Cloud9, Oracle, Salesforce.com, and SAP

Deployment models
• NIST visual model of cloud computing

<table>
<thead>
<tr>
<th>Essential Characteristics</th>
<th>Broad Network Access</th>
<th>Rapid Elasticity</th>
<th>Measured Service</th>
<th>On-Demand Self-Service</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resource Pooling</td>
<td>SaaS</td>
<td>PaaS</td>
<td>IaaS</td>
<td></td>
</tr>
<tr>
<td>Service Models</td>
<td>Public</td>
<td>Private</td>
<td>Hybrid</td>
<td>Community</td>
</tr>
</tbody>
</table>

• Public cloud
  – Available to general public
  – Owned and provisioned by an organization selling cloud services
  – NIST definition
    * Cloud infrastructure provisioned for open use by general public
    * Maybe owned, managed, and operated by some combination of a business, academic, or government organization
  – Multitenant environment
  – Resources used from a shared grid of commodity resources
  – Users unaware of physical location of resources or data center
    * Users access resources through an abstraction layer on top of the physical hardware
    * Virtual compute resources created by APIs in the abstraction
  – Advantages of public clouds
    * Utility pricing
      · Pay for the resources consumed
      · Scale up or scale down as per need
      · No procurement of physical hardware, except for the hardware to connect to the cloud
      · No wasted compute cycles
    * Elasticity
      · Endless pool of resources
      · Configure software solutions to dynamically increase/decrease resources to handle peak loads
      · React to traffic spikes in real-time
    * Core competency
      · Outsourced data center and infrastructure management
      · More time on core competence
  – Risks of public clouds
    * Control
      · Reliance on vendor for performance and uptime
      · Outage at cloud vendor could adversely affect services
    * Regulatory issues
      · PCI DSS – Payment Card Industry Data Security Standard
      · HIPAA – Health Information Portability and Accountability Act
      · Data privacy issues
      · May be solved by leveraging certified SaaS solutions for components that are hard to audit in public cloud
    * Limited configurations
      · May not be able to access specific hardware to solve intensive computational problems
• Private cloud
  – Hosted within an organization’s firewall, managed by the organization
  – Created and controlled by the enterprise
  – NIST definition
    * Cloud infrastructure provisioned for exclusive use by a single organization with multiple consumers or business units
    * May be owned, managed, and operated by the organization, a third party, or some combination
    * May exist on or off premises
  – Deploy in a single-tenant environment and not comingled with other customers
  – Costs more than sharing in public cloud environment, but more control and security
  – Reduce regulatory risks

• Hybrid cloud
  – Combination of public and private clouds
  – Management responsibilities divided between public cloud provider and the business renting it
  – NIST definition
    * Composition of two or more distinct cloud infrastructures (private, community, or public)
    * The clouds remain unique entities but bound together by standardized or proprietary technologies to enable data/application portability
    * Cloud bursting for load balancing between clouds

Cloud computing worst practices

• Avoiding failure when moving to cloud
  – Understanding and embracing new technology
  – Necessary architecture and design of applications vs development
  – Unrealistic expectations
    * Aggressive due dates
    * Large scope
    * Human resources

• Migrating applications to the cloud
  – Will migrating existing applications to the cloud cut down costs?
  – Tightly coupled architecture (software/hardware/environment)
  – Cloud computing applications require loosely coupled architecture
  – Legacy applications
    * Not meant to be scaled up/down automatically
    * Use vertical scaling
      · Add more hardware (CPU/memory/disk) or replace existing hardware with more powerful hardware
      · Software changes achieved by changing configuration files
      · Applications tightly coupled with hardware for performance
      · Migrating an application from existing hardware requires major reengineering to remove hardware dependencies
    * System not responsive to unanticipated spikes in workload
- Hosting vs cloud
  * Hosting used when a company does not want to manage and maintain infrastructure
  * Hosting does not provide the characteristics of cloud computing
    - Broad network access
    - Elasticity
    - Measured service
    - On-demand self-service
    - Resource pooling
  * Vertical scaling but responsibility given to infrastructure provider
- Horizontal scaling in the cloud
  * Additional infrastructure running in conjunction with existing infrastructure
  * Scaling out instead of scaling up
  * Performed at multiple layers of architecture
  * Add nodes by
    - Server farm type
Stateful vs stateless system design

* Stateless service does not maintain any history of requests
  - Only aware of the transaction information
  - Maintains application state on the client and not server
  - No dependency on infrastructure
  - Loan service request to evaluate credit rating of a customer applying for loan
  - Service has no record of information on customer
  - After processing, it does not store any information within session and does not have any information on customer

Misguided expectations

* Some examples (Instagram and Netflix) overhype the situation
  - Success will require vision, talent in the team, and ability to execute
* Successfully running legacy applications complicate the issue
* Architecting vs cost saving
* Cost depreciation of assets over the year
  - Buy in advance to account for perceived surges in traffic and growth over time
  - Correct architecture to scale up/down and turn off extra cloud services
  - Align cost with revenue (pay-as-you-go)
* Moving code repository into cloud
  - $3,000 fixed cost for server vs 50 cents per hour for the cloud
  - Cloud cost comes to $4,380 per year
* Must set realistic goals and expectations
  - Proper analysis
  - Design to optimize, monitor, and audit cloud service consumption
  - Closely monitor the costs from the cloud service provider

Misinformed about cloud security

* Extreme views
  - Do not place anything on public cloud
    - Build private clouds
    - Need to develop competence in security and infrastructure
  - Security is the responsibility of cloud vendors
    - May leave big holes in deployed software
– Lack of experience personnel to build secure applications for the cloud
  * Security expertise is ever changing and evolving
– Cloud vendors may host resources and data for a large number of companies
  * Makes them a huge target for cybercriminals
– Cloud vendors provide just perimeter security
  * Application security still the responsibility of client
  * Responsibility with architect to encrypt data, manage keys, and implement good password policies
– Good security can enable public cloud to be more secure than private data centers
  * Most of the security breaches are inside jobs
  * A number of those happen due to carelessness
    - Lost, stolen, or misplaced assets – thumb drives, disks, documents, devices, laptops
– Planning for security
  * Designed into software
  * Security best practices applied in data centers must be applied in cloud as well
  * May need additional steps to pass regulatory audits such as HIPPA
– Typically, security is a core competency with cloud providers
  * Leverage security as a service from cloud providers
  * Must know security risks with a combination of technology, process, and governance

• Favorite vs appropriate vendor
  – Going with your existing biases may not be correct
  – Microsoft Azure (PaaS) for .NET applications
  – For scaling requirements, IaaS provider may be a better choice compared to a PaaS
    * PaaS providers have thresholds enforced within architecture layers to ensure that one customer does not consume too many resources
    * Fewer such limitations with IaaS

• Outages and out-of-business scenarios
  – Everything can and will fail
  – Design for failure
  – Cause and effect of lock-in with proprietary technology
  – If the cloud vendor fails, the service disappears
  – Best practices leveraging SaaS or PaaS database technology
    * Access to data outside of service provider
    * Snapshots of data backups, a daily extract, or some method to store recoverable data independent of service and provider
  – Outage within zones
    * Avoided by using multiple zones
    * AWS provides multiple zones within a region and multiple regions across the globe
    * SLA of 99.95% uptime implies a downtime of 20 minutes and 9 seconds per month, or about 4 hours per year
    * Impact of downtime on average business: $5,000 per minute or $300,000 per hour

• Impact of organizational change
  – Buying virtual services vs physical assets; is procurement ready for that?
  – Paying for on-demand service
– Forecasting usage in the future vs real-time autoscaling for capacity planning
– Securing data outside of corporate firewalls
– Proof-of-concept by storing some noncritical data with a cloud service provider
– Technological problem vs people problem
  ∗ Resistance to change
  ∗ People told to change vs nudged to change

• Skills shortage
  – Existing skills centered on applications and available hardware
    ∗ Optimization and security; on-premises virtualization
  – Stateless and loosely-coupled cloud architectures
  – Integration with multiple cloud-based solutions with other vendors, partners, and customers
  – Significant change from the perspectives of architecture, business process, and people
  – Application security skills to ensure safety of data and intellectual property outside of corporate firewalls
  – Close cooperation between system administration and development teams
    ∗ System administrators as part of release management lifecycle

• Misunderstanding customer requirements
  – Business requirements vs IT preferences
  – Security and privacy requirements; regulatory constraints

• Unexpected costs
  – Governing process of consuming cloud resources
    ∗ SaaS for storage and applications
    ∗ Different tiers of service based on number of users and amount of storage, or amount of compute power
  – SaaS services
    ∗ Free repositories vs payment based
    ∗ Charge per user or per seat (floating license)
    ∗ Charge monthly or transaction based (email campaign)
    ∗ Triggers to avoid surprises
  – PaaS services
    ∗ Allows developers to focus on business requirements while platform handles the infrastructure
    ∗ Handling scaling during peak times
    ∗ Make sure that PaaS does not consume huge amount of infrastructure, possibly using triggers as above
  – IaaS services
    ∗ Server sprawl
    ∗ Different groups creating multiple servers
    ∗ Automatic allocation of servers to groups
    ∗ May not be able to shut down the servers as they may be running some critical applications

Architecture

• Development phase vs analysis of business and technical requirements
Importance of asking questions

• Value of enterprise architecture
  – Perform the necessary discovery steps before diving headfirst into cloud
  – Do not select a vendor before due diligence
  – Answer the following questions
    * Why. What problems are we trying to solve? What are the business goals and drivers?
    * Who needs this problem solved? Who are all the actors involved (internal/external)?
    * What are the business and technical requirements? What legal and/or regulatory constraints apply? What are the risks?
    * Where will these services be consumed? Are there any location-specific requirements (regulations, taxes, usability concerns, language/locale issues)?
    * When are these services needed? What is the budget? Are there dependencies on other projects/initiatives?
    * How can the organization deliver these services? What is the readiness of the organization, the architecture, the customer?
  – Other factors
    * Is the project being built from scratch from the ground up?
    * Is the project a migration of a legacy system?
    * A combination of the two?
    * Does the cloud provider provide any migration services?
    * Types of users and data
      - Social networking site vs medical records site

Business architecture

• Business architecture diagram to show touchpoints and business functions across the enterprise
  – Work on different components
  – Level of visibility into the overall vision of the enterprise

Identify the problem statement (Why)

• What problem are we trying to solve?
• Business drivers to leverage cloud computing services within organization
  – No-brainer for a startup building new in cloud
  – Bigger problem for established companies with large investment in physical infrastructure
  – Reducing infrastructure costs?
  – May replace non-core-competency processes, such as payroll, human resources, and accounting, by SaaS
  – May leverage cloud for storage, backup/recovery, provisioning testing and development environments on demand, or integrate with external APIs such as maps

Evaluate user characteristics (Who)

• Internal and external users; people or systems
• Understand the characteristics of users such as demographics, location, type of actor (person/business/government), type of business (social media/health/manufacturing)

• Need to account for privacy, regulations, usability, risk, and more

**Identify business and technical requirements (What)**

• Drives the discovery of functional and nonfunctional requirements

• Functional requirements
  – What data the system must process
  – How the screens should operate
  – How the workflow operates
  – What are the system outputs
  – Who has access to each part of the system
  – What regulations must be adhered to

• Nonfunctional requirements
  – Usability – Requirements for end users and systems using the platform
  – Performance – Ability to respond to user and system requests
  – Flexibility – Ability to change the speed of business with minimal code change
  – Capability – Ability to perform current and future business functions
  – Security – Requirements for security, privacy, and compliance
  – Traceability – Logging, auditing, notification, and event processing
  – Reusability – Level of reuse required both internally and externally
  – Integrability – Integrate with various systems and technologies
  – Standardization – Specific industry standards
  – Scalability – Scale to meet demands
  – Portability – Deploy on various hardware and software platforms
  – Reliability – Uptime and SLAs, along with recovery mechanisms

**Visualize the service consumer experience (Where)**

• Impact of laws relative to user locale
  – Point of consumption
  – Point of data storage
  – Restrictions on data transfer into or out of a country

• Hybrid cloud solutions
  – Public IaaS or PaaS services for majority of processing needs
  – Sensitive data in a private cloud or in-house data center

• Access mechanism for the services (devices and touchpoints)

• Effects of time zones
Identify project constraints (When and with What)

- Budget and expected delivery dates
- Compromise between architecture decisions and business goals/deadlines
- Mandate to use a certain vendor
- Prototyping vs production systems

Understand current state constraints (How)

- Organizational readiness
  - Resources in house?
  - Capital expenditure (buy up-front) or operational expenditure (pay-as-you-go)
- Resistance within ranks