What is a cloud?

SLAs

Web Services

Virtualization
eCommerce + Infrastructure

- **Self-service and “zero touch.”**
  - Scalable automatic rental of resource intensive goods

- **Transactional and asynchronous**
  - Interaction with the site is transactional
  - Delivery is asynchronous

- **Site integrity and site availability are critical**
  - Individual transactions can fail but the site cannot

- **Customer requests must be isolated**
  - Service venue must manage competing needs

- **Scale out for request volume, scale up for request weight**
Open-source Cloud Infrastructure

• **Idea:** Develop an open-source, freely available cloud platform for commodity hardware and software environments
  – Stimulate interest and build community knowledge
  – Quickly identify useful innovations
  – Act to dampen the “hype”

• **First-principles cloud implementation**
  – Not a refactorization of previously developed technology

• **Build from mature open source technologies**
  – J2EE, MySQL, Web Services are high quality and scalable as open source
What’s in a name?

- **Elastic Utility Computing Architecture Linking Your Programs To Useful Systems**
- Web services based implementation of elastic/utility/cloud computing infrastructure
  - Linux image hosting ala Amazon
- **How do we know if it is a cloud?**
  - Try and emulate an existing cloud: Amazon AWS
- Functions as a software overlay
  - Existing installation should not be violated (too much)
- Focus on installation and maintenance
  - “System Administrators are people too.”
Goals for Eucalyptus

• Foster greater understanding and uptake of cloud computing
  – Provide a vehicle for extending what is known about the utility model of computing

• Experimentation vehicle prior to buying commercial services
  – Provide development, debugging, and “tech preview” platform for Public Clouds

• Homogenize local IT environment with Public Clouds
  – AWS functionality locally makes moving using Amazon AWS easier, cheaper, and more sustainable

• Provide a basic software development platform for the open source community
  – E.g. the “Linux Experience”

• Not designed as a replacement technology for AWS or any other Public Cloud service
Requirements

- **Implement cloud abstractions and semantics**
  - Must be a cloud (inaugurably)

- **Simple**
  - Must be transparent and easy to understand

- **Scalable**
  - Interesting effects are observed at scale (e.g. not an SDK)

- **Extensible**
  - Must promote experimentation

- **Non-invasive**
  - Must not violate local control policies

- **System Portable**
  - Must not mandate a system software stack change

- **Configurable**
  - Must be able to run in the maximal number of settings

- **Easy**
  - To distribute, install, secure, and maintain
The Elements of Cloud Style

• The terms SaaS, PaaS, and IaaS are often viewed as creating a pain in the...

• SaaS (Software as a Service)
  – Applications exporting network-facing user interfaces
  – User transfers data to the cloud

• PaaS (Platform as a Service)
  – Program or scripting runtime exports network-facing interfaces
  – Internal platform services available
  – User transfers program code and data to the cloud

• IaaS (Infrastructure as a Service)
  – Resource provisioning services export network-facing interfaces
  – Internal platform services available
  – User transfers code, data, and environment to the cloud
Why IaaS?

- **Applications are often multi-technology**
  - System “images” for different technologies can be combined
  - Multiple language technologies at different revision levels

- **Legacy support**
  - System images that mimic bare metal deployments can be used
  - Legacies are archived with the environment necessary to run them

- **Transparency**
  - Debugging and performance tuning can go down to the hypervisor

- **QoS containers**
  - QoS is implemented in the infrastructure today => familiar

- **Anti-lock in**
  - If clouds fail, a return path to bare metal is available
Why not IaaS?

- **Self-service pushes system administration tasks to the end-user**
  - Users must understand dynamic resource provisioning
- **QoS hard to optimize at a fine-grained level**
  - A machine is a pretty big QoS bundle
- **Heterogeneity is powerful but hard to manage**
  - Multi-technology development and maintenance is a tough software engineering problem
- **Tenancy density and cloud platform optimization**
  - Less optimization potential at the VM level
Three Research Questions for IaaS

• *How can a cloud resolve the tension between elasticity and specialization?*
• *What is the best development model for hybrid clouds?*
• *How should cloud software be organized within an application?*
Elasticity and Specialization

- **Elasticity** measures the ability of the cloud to map a single user request to different resources.
  - AWS VM can be implemented on a wide variety of infrastructure configurations
  - Simple device model is necessary for OS elasticity

- **Most data centers use specialization to encode “process”**
  - Technology lifecycle
  - User priority
  - Workload priority
  - QoS

- **The more elastic, the less specialized, but the less specialized, the less customized**
Hybrid Clouds

• **Public Cloud**
  – Flat ID Management system and “limitless” scale
  – “roll forward” development
    • Craft a new VM when a run time exception occurs
    • Garbage collect asynchronously

• **Private Cloud**
  – Complex access controls and limited resources/quotas
  – Resource management throughout the stack is critical

• *How can one application live comfortably in both worlds?*
• **Software stacks are losing their “polarity” in clouds**
  – File system on top of NoSQL on top of Put/Get on top of File system on top of…
  – “The Stack is Lost.”

• **New Model: The Service Ensemble**
  – Applications are composed of service graphs not layered stacks

• **What software engineering principles make sense?**
  – Communication is asynchronous
  – Failures are common
  – Whole “machines” can be composed dynamically
Three Questions we have Answered

• *How is Cloud Computing Different from other Approaches?*
• *Why use a private cloud?*
• *Can the “cloudification” of applications be automated?*
It is and It isn’t

- **Cloud:** Elastic eCommerce-style service venue for resource access and automatic configuration
- **Not Cloud:**
  - Data Center Virtualization
    - synchronous
    - Not user scalable
  - Grid
    - Policy federated
    - Inelastic
    - One user, many resources
  - Peer2Peer
    - Lack of administratable abstractions
Why Private Clouds?

- **Technology Lifecycle Independence**
  - Lost of OS, Communication, Hardware, Data, Virtualization in the data center
  - One platform to remain stable as these technologies age and roll forward

- **Separation of support concerns**
  - “Below” the cloud platform managed by administrators
  - “Above” the cloud platform managed by users
  - Infrastructure support externalized toward the users

- **On-boarding Ecosystem**
  - Isolation properties imply the “Linux Distro of the Future.”
“Cloudification” of Applications

• **Step 1: Configuration must be discovered**
  – Metadata service
  – Templating

• **Step 2: SLA is in the abstraction and not in the configuration**
  – Examples:
    • Network interface and not IP address carries QoS
    • Block device and not the specific volume carries the DB QoS

• **It is not, at present, possible to map arbitrary Data Center semantics onto an elastic cloud model**
  – Requires some human intervention
The Case for Open Source

- Linux is the operating system platform of choice for machines because...
  - Hardware portable
    - Separates software lifecycle from hardware lifecycle
    - Prevents lock-in
  - Vast ecosystem of software
    - Linux distros provide QA (free or paid)
  - Transparent
    - Possible to own the source code for everything
  - Fast to remediate
    - Open source web community is often faster than paid support
  - Cost effective
    - Possible to mix free and paid offerings fluidly
OSS and The Next Data Center

- If…
  - the most mature eCommerce technologies are open source
- And…
  - Enterprise IT prefers open source platforms for deployment at scale
- And…
  - Private Clouds are the next platform for IT
- Then…
  - The On-premise Private Cloud will be built from Open Source
Happening Already?
Thanks!

• Thanks to our original research sponsors…

[Logos of UCSB, VGrADS, NSF]

• …and to our new commercial friends

[Logos of Benchmark Capital, NEA, BV Capital]

www.eucalyptus.com
805-845-8000
rich@eucalyptus.com