



BITS Pilani
Pilani Campus

BITS Pilani presentation

Mridul Moitra
Cloud Computing



BITS Pilani
Pilani Campus



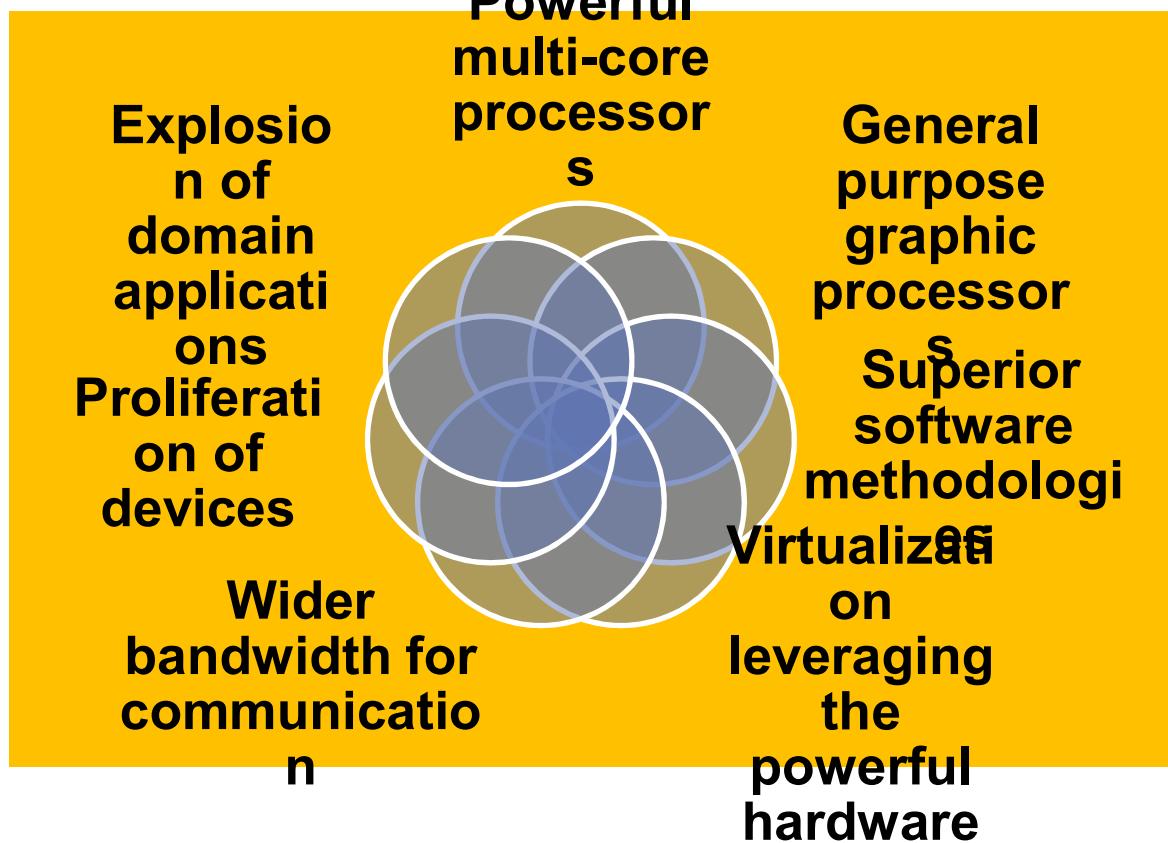
<CSI ZG527 / SS ZG527 / SE ZG527 **Cloud Computing** **Lecture No. 1**

Introduction to Cloud Computing, services and deployment models



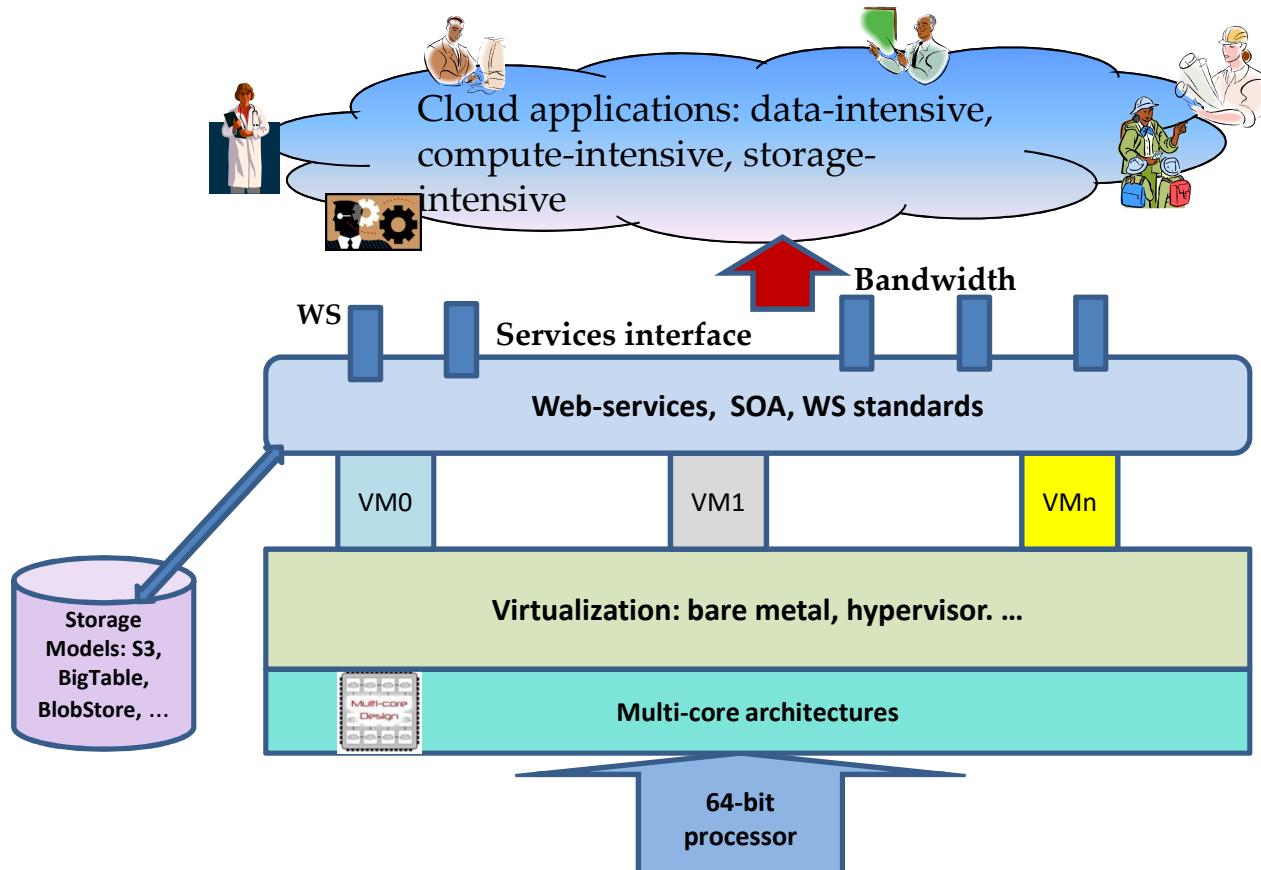
- **Agenda**
 1. **Introduction to Cloud Computing – Origins and Evolution**
 2. **Characteristics of cloud platform**
 3. **Types of Clouds and Services**
 4. **Cloud Delivery Model**
-

Motivation



- 1. Web Scale Problems**
- 2. Web 2.0 and Social Networking**
- 3. Information Explosion**
- 4. Mobile Web**

Technology Advances



Evolution of cloud computing

?

- **The evolution of cloud computing can be bifurcated into three basic phases:**
- **1. The Idea Phase-** This phase inceptioned in the early 1960s with the emergence of utility and grid computing and lasted till pre-internet bubble era. Joseph Carl Robnett Licklider was the founder of cloud computing.
- **2. The Pre-cloud Phase-** The pre-cloud phase originated in 1999 and extended to 2006. In this phase the internet as the mechanism to provide Application as Service.
- **3. The Cloud Phase-** The much talked about real cloud phase started in the year 2007 when the classification of IaaS, PaaS, and SaaS got formalized. The history of cloud computing has witnessed some very interesting breakthroughs launched by some of the leading computer/web organizations of the world.



What is Cloud Computing?

Cloud Computing is a general term used to describe a new class of network based computing that takes place over the Internet,

- basically a step on from Utility Computing
- a collection/group of integrated and networked hardware, software and Internet infrastructure (called a platform).
- Using the Internet for communication and transport provides hardware, software and networking services to clients

These platforms hide the complexity and details of the underlying infrastructure from users and applications by providing very simple graphical interface or API (Applications Programming Interface).

What is Cloud Computing cont....



the platform provides

- on-demand services, that are always on, anywhere, anytime and any place.
- Pay for use and as needed, elastic
- scale up and down in capacity and functionalities
- The hardware and software services are available to
- The hardware and software services are available to
- general public, enterprises, corporations and businesses markets

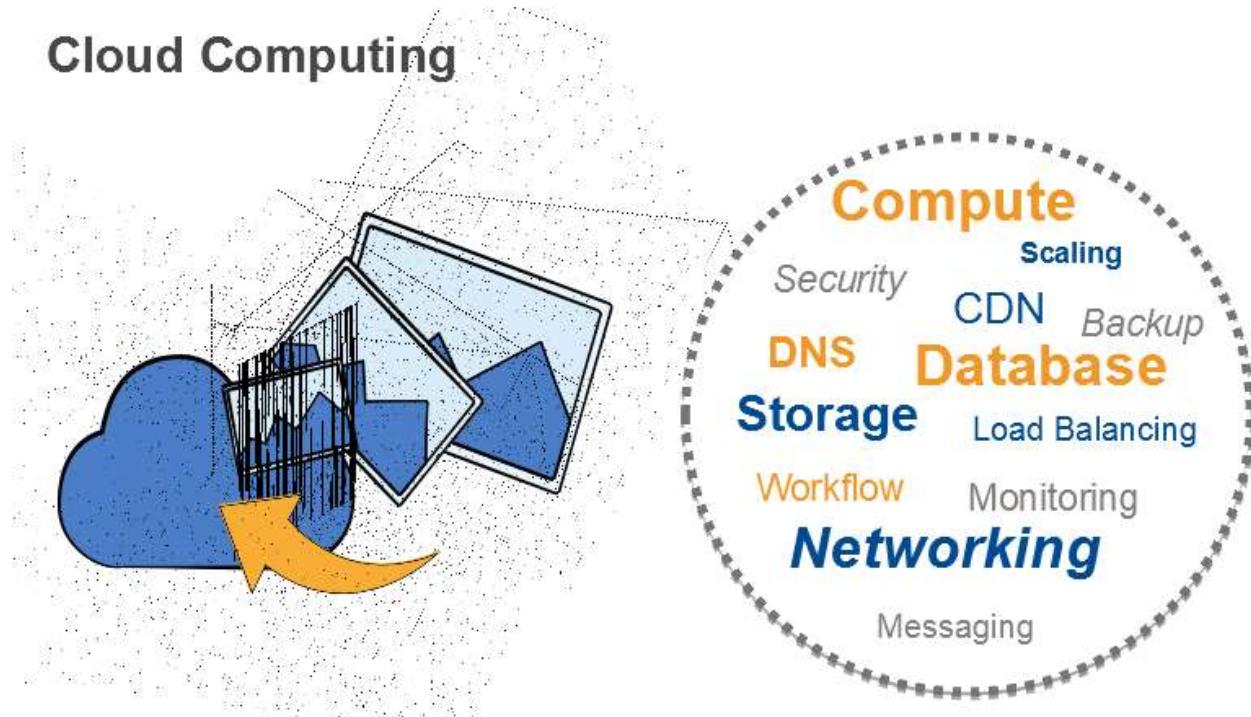


Cloud Computing: Definition

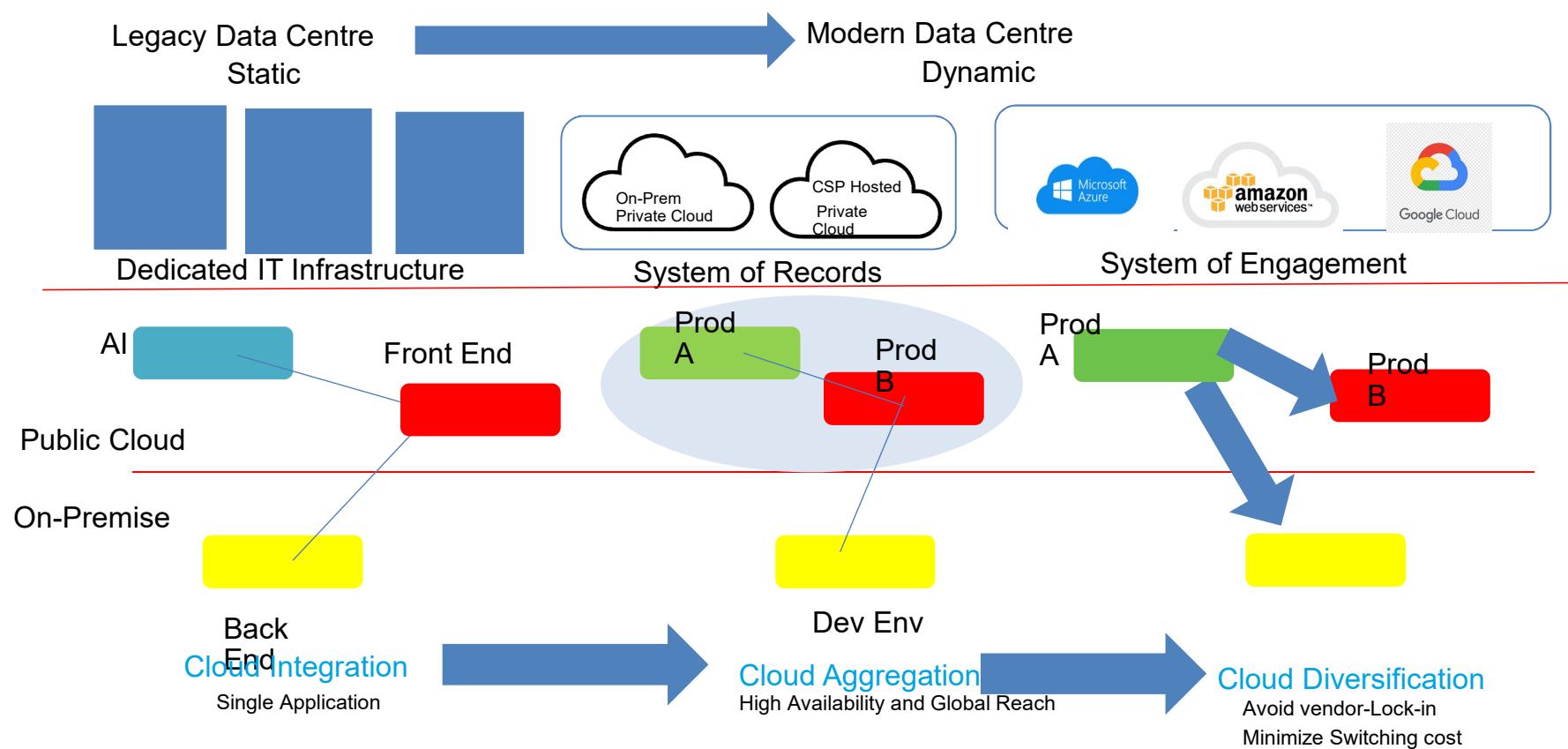
The US National Institute of Standards (NIST) defines cloud computing as follows:

Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.

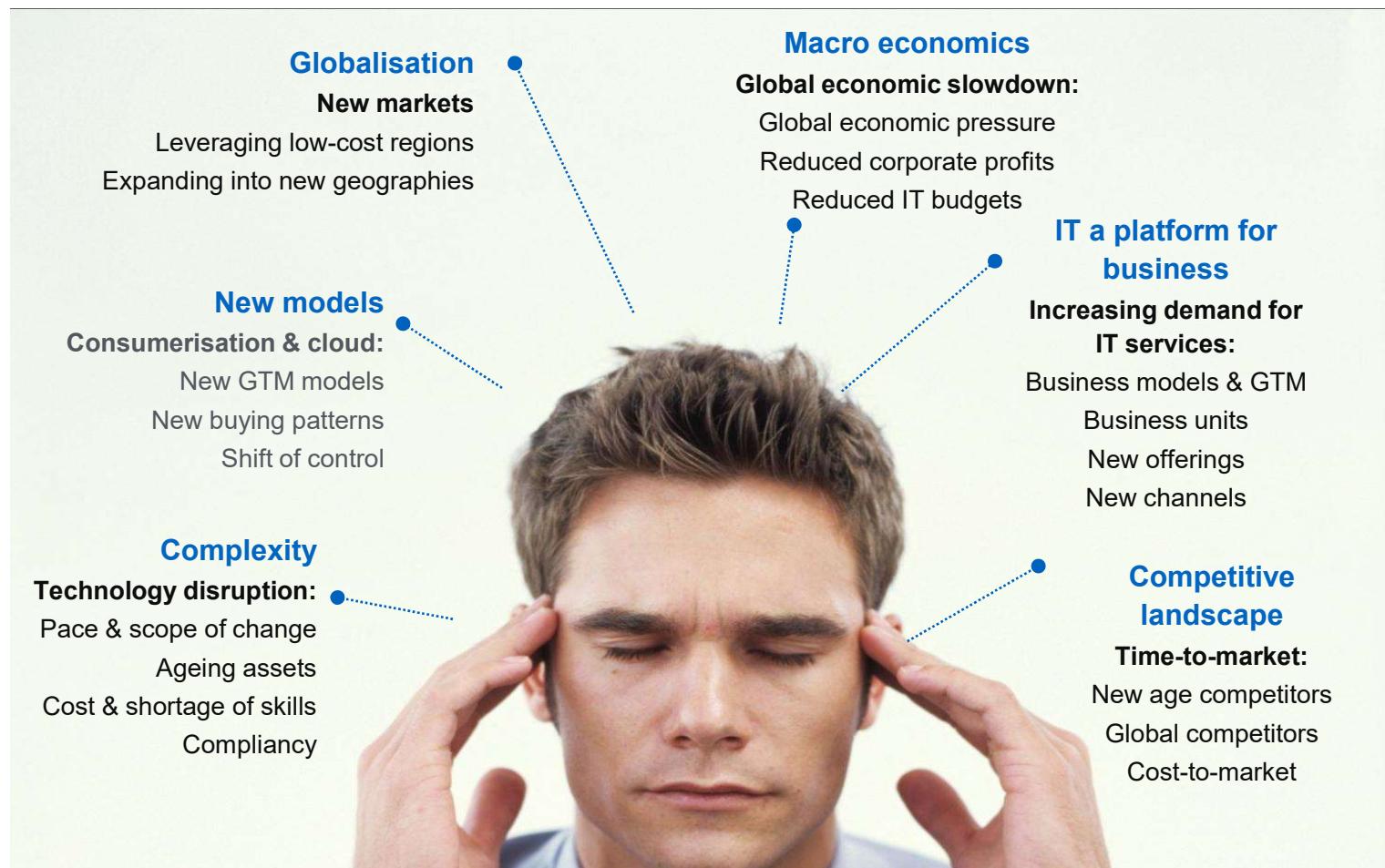
Cloud Computing



Cloud Transformation Journey



Challenges Of the CIO





Drivers for the new Platform

Generational Shift of Computing Platform

Technology	Economic	Business
	Centralized compute & storage, thin clients	Optimized for efficiency due to high cost
	PCs and servers for distributed compute, storage, etc.	Optimized for agility due to low cost
	Large DCs, commodity HW, scale-out, devices	Perpetual license for OS and application software
	Order of magnitude better efficiency and agility	Pay as you go, and only for what you use

<http://blogs.technet.com/b/yungchou/archive/2011/03/03/chou-s-theories-of-cloud-computing-the-5-3-2-principle.aspx>

Cloud Computing Business Drivers



Cost optimisation



- No capex, less assets
- Pay-as-you-use
- On-demand capacity
- Elasticity
- Economies of scale
- Time-to-value

Risk optimisation



- Business continuity
- Technology independence
- Operational complexity
- Specialised skills

Strategic agility



- Time-to-market
- Innovation
- New business models
- Resource leverage
- Adaptability
- Flexibility

...why would one not consider these benefits?

3-4-5 rule of Cloud Computing



NIST specifies 3-4-5 rule of Cloud Computing

- 3** cloud service models or service types for any cloud platform
- 4** deployment models
- 5** essential characteristics of cloud computing infrastructure

Cloud Summary



- Shared pool of configurable computing resources
- On-demand network access
- Provisioned by the Service Provider



Cloud Summary...

Cloud computing is an umbrella term used to refer to Internet based development and services

A number of characteristics define cloud data, applications services and infrastructure:

Remotely hosted: Services or data are hosted on remote infrastructure.

Ubiquitous: Services or data are available from anywhere.

Commodity model: The result is a utility computing model similar to traditional that of traditional utilities, like gas and electricity - you pay for what you would want!

Characteristics of Cloud Computing



5 Essential Characteristics of Cloud Computing

Ref: The NIST Definition of Cloud Computing

<http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf>



Source: <http://aka.ms/532>

- On demand self-service
- Broad network access
- Resource pooling
- Rapid elasticity
- Measured service

Characteristics of Cloud Computing



Essential Characteristics

- On-demand self-service
- Broad network access
- Resource pooling
- Rapid elasticity
- Measured service

Service Models

- Software as a Service
- Platform as a Service
- Infrastructure as a Service

Deployment Models

- Private
- Public
- Hybrid
- Community



Traditional Infrastructure

Amazon Web Services



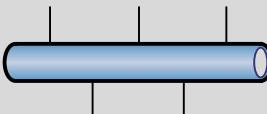
Security



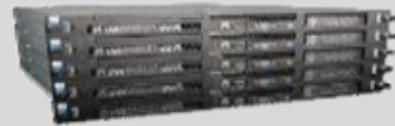
Security Groups

NACLs

Access Mgmt



Networking



Servers



Amazon EC2
Instances



RDBMS

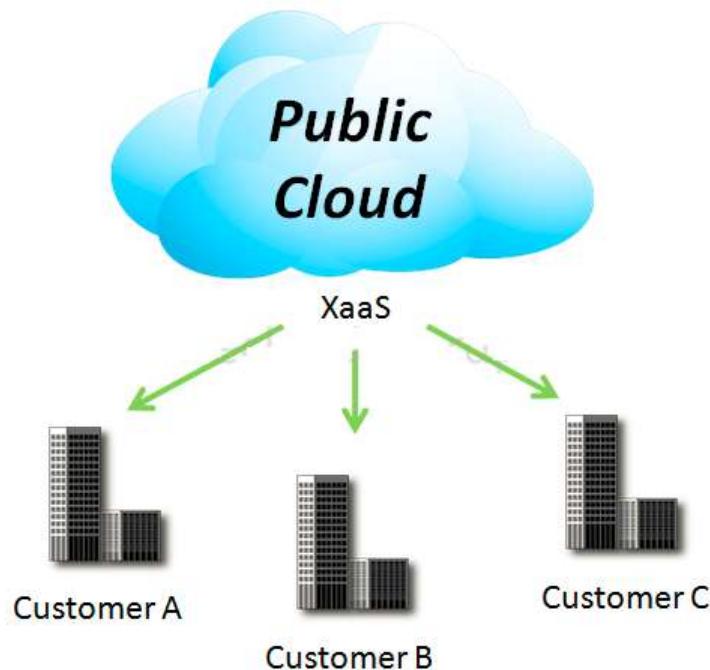
Storage & Database



4 Deployment Models



1. Public Cloud

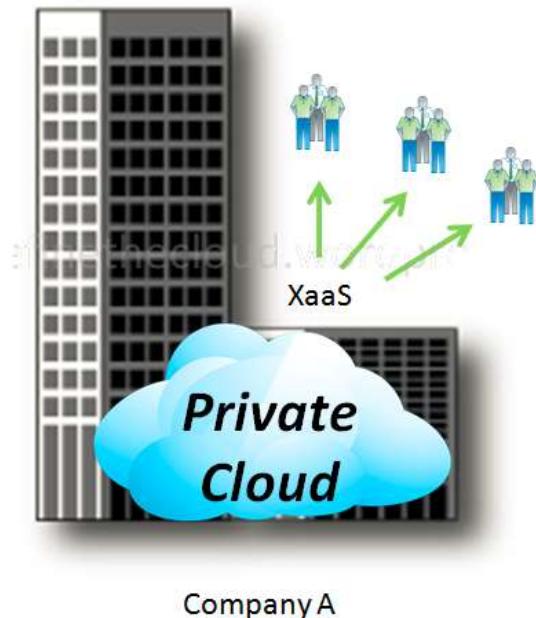


Mega-scale cloud infrastructure is made available to the general public or a large industry group and is owned by an organization selling cloud services.

4 Deployment Models



2. Private Cloud

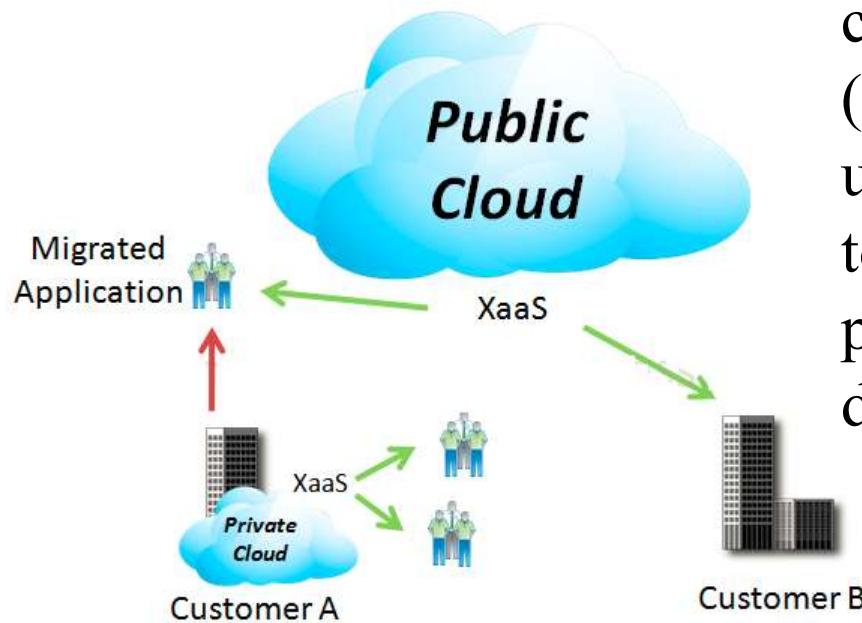


The cloud infrastructure is operated solely for an organization. It may be managed by the organization or a third party and may exist on premise or off premise.

4 Deployment Models



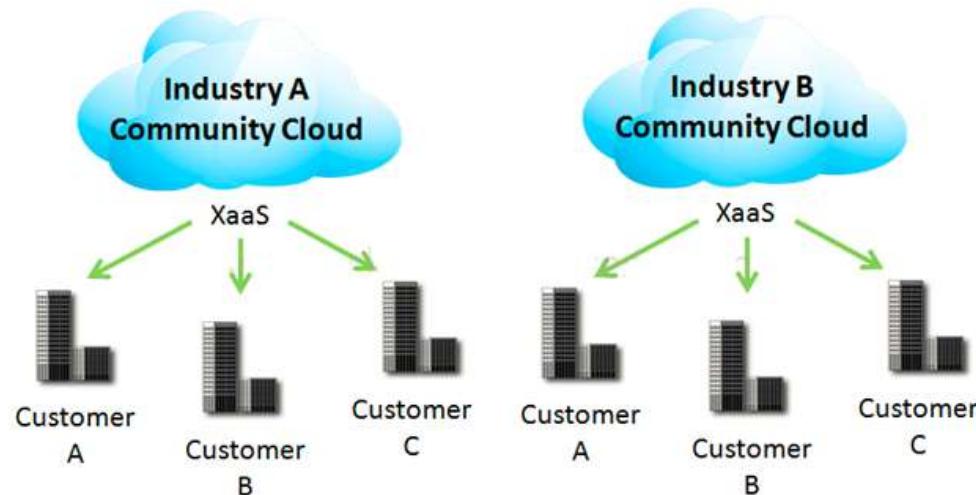
3. Hybrid Cloud



The cloud infrastructure is a composition of two or more clouds (private or public) that remain unique entities but are bound together by standardized or proprietary technology that enables data and application portability

4 Deployment Models

4. Community Cloud



Community Clouds are when an ‘infrastructure is shared by several organizations and supports a specific community that has shared concerns (e.g., mission, security requirements, policy, and compliance considerations). It may be managed by the organizations or a third party and may exist on premise or off premise’ according to NIST. A community cloud is a cloud service shared between multiple organizations with a common tie/goal/objective. E.g. OpenCirrus

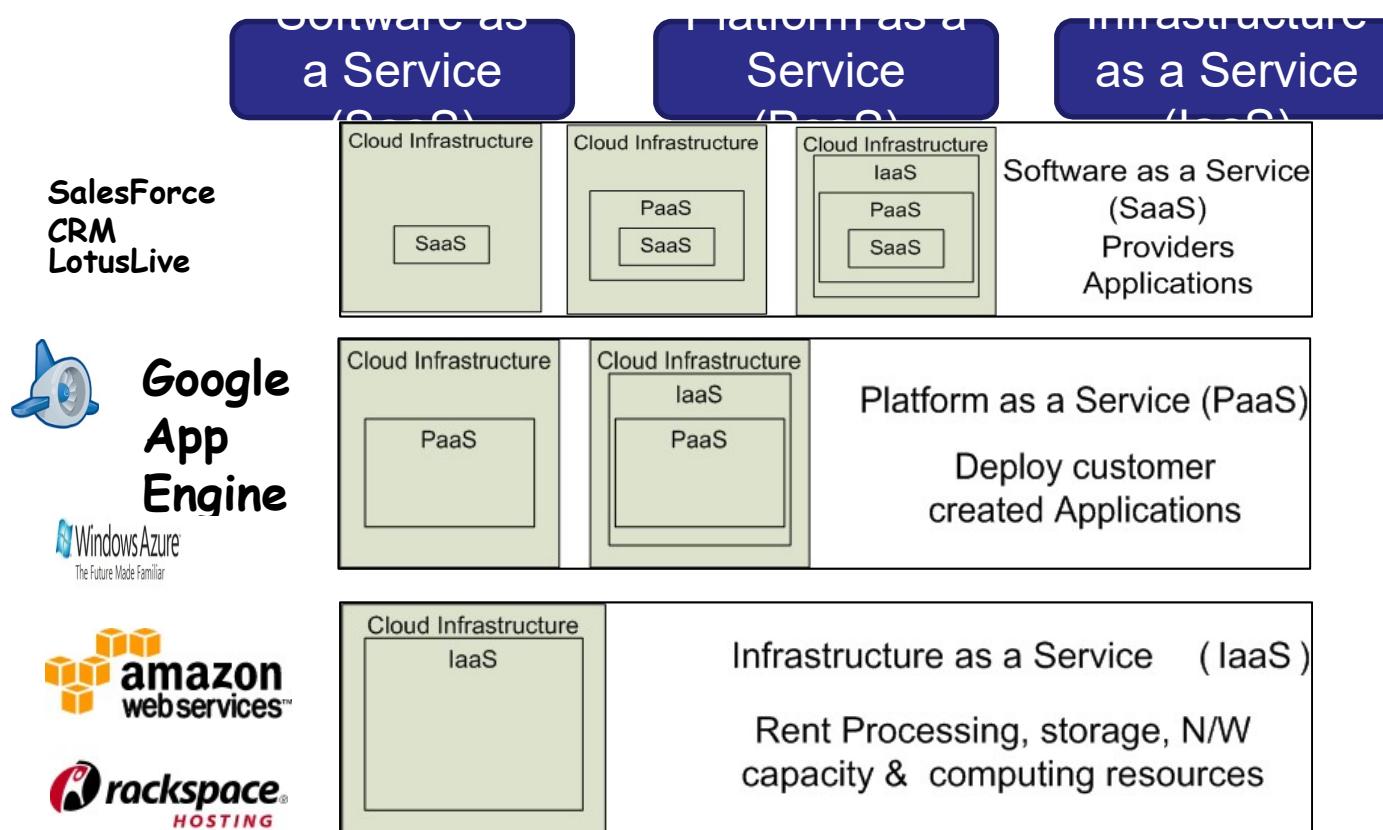
Introduction to Cloud Computing, services and deployment models



- **Agenda**
 1. **Introduction to Cloud Computing – Origins and Motivation**
 2. **3-4-5 rule of Cloud Computing**
 3. **Types of Clouds and Services**
 4. **Cloud Infrastructure and Deployment**
-



3 Cloud Service Models





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.

E.g. salesforce.com , Google Apps



Platform as a Service

Platform as a service encapsulates a layer of software and provides it as a service that can be used to build higher-level services.

2 Perspectives for PaaS :-

- 1. Producer:-** 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.
- 2. Consumer:-** 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.



Infrastructure as a Service

Infrastructure as a service delivers basic storage and computing 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.



Service Models Summary

Cloud Software as a Service (SaaS)

The **capability provided to the consumer is to use the provider's applications** running on a cloud infrastructure and accessible from various client devices through a thin client interface such as a Web browser (e.g., web-based email). The consumer does not manage or control the underlying cloud infrastructure, network, servers, operating systems, storage, or even individual application capabilities, with the possible exception of limited user-specific application configuration settings.

Cloud Platform as a Service (PaaS)

The **capability provided to the consumer is to deploy onto the cloud infrastructure consumer-created applications using programming languages and tools supported by the provider** (e.g., Java, Python, .Net). The consumer does not manage or control the underlying cloud infrastructure, network, servers, operating systems, or storage, but the consumer has control over the deployed applications and possibly application hosting environment configurations.

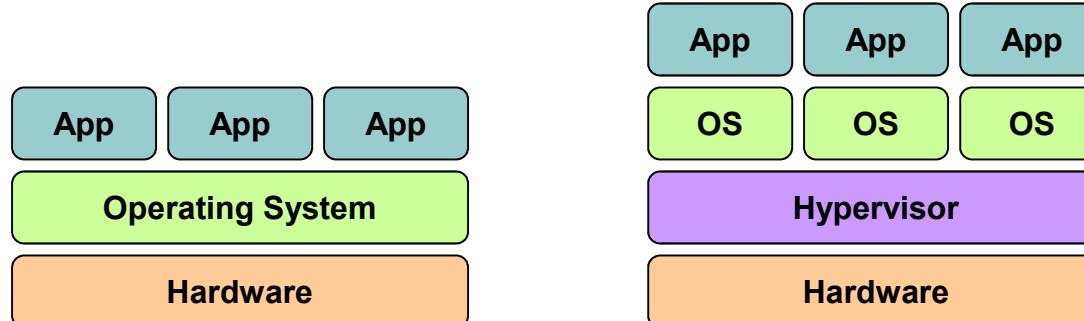
Cloud Infrastructure as a Service (IaaS)

The **capability provided to the consumer is to rent processing, storage, networks, and other fundamental computing resources** where the consumer is able to deploy and run arbitrary software, which can include operating systems and applications. The consumer does not manage or control the underlying cloud infrastructure but has control over operating systems, storage, deployed applications, and possibly select networking components (e.g., firewalls, load balancers).

Cloud Infrastructures



Key Technology is Virtualization



Virtualization plays an important role as an enabling technology for datacentre implementation by abstracting compute, network, and storage service platforms from the underlying physical hardware

Cloud Providers Characteristics



- Provide on-demand provisioning of computational resources
- Use virtualization technologies to lease these resources
- Provide public and simple remote interfaces to manage those resources
- Use a pay-as-you-go cost model, typically charging by the hour
- Operate data centers large enough to provide a seemingly unlimited amount of resources to their clients



Management of Virtualized Resources

Distributed Management of Virtual Machines

Reservation-Based Provisioning of Virtualized Resources

Provisioning to Meet SLA Commitments



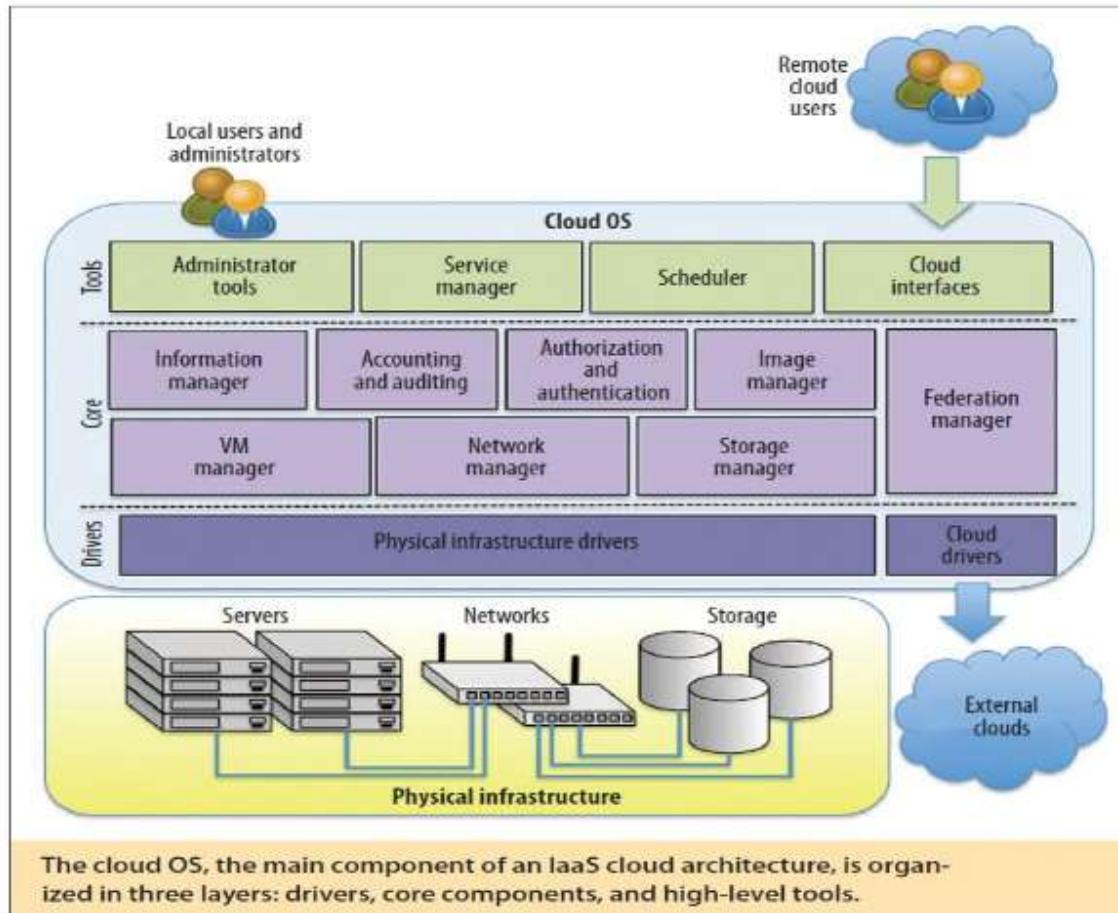
Cloud Infrastructure Anatomy

The key component of an IaaS cloud architecture is the cloud OS, which manages the physical and virtual infrastructures and controls the provisioning of virtual resources according to the needs of the user services

A cloud OS's role is to efficiently manage datacenter resources to deliver a flexible, secure, and isolated multitenant execution environment for user services that abstracts the underlying physical infrastructure and offers different interfaces and APIs for interacting with the cloud

While local users and administrators can interact with the cloud using local interfaces and administrative tools that offer rich functionality for managing, controlling, and monitoring the virtual and physical infrastructure, remote cloud users employ public cloud interfaces that usually provide more limited functionality

The Cloud OS



The cloud operating system is responsible for:

1. managing the physical and virtual infrastructure,
2. orchestrating and commanding service provisioning and deployment
3. providing federation capabilities for accessing and deploying virtual resources in remote cloud infrastructures

Pros and Cons of Cloud Computing



- Why is it becoming a Big Deal:
 - Using high-scale/low-cost providers,
 - Any time/place access via web browser,
 - Rapid scalability; incremental cost and load sharing,
 - Can forget need to focus on local IT.
- Concerns:
 - Performance, reliability, and SLAs,
 - Control of data, and service parameters,
 - Application features and choices,
 - Interaction between Cloud providers,
 - No standard API – mix of SOAP and REST!
 - Privacy, security, compliance, trust...

Public Cloud



Implementation of cloud services on resources that are shared between many customers, managed off-premises



Typically, cloud provider owns and controls the resources/assets, definition of services, costs and risks

Variations exist – such as hosters and integrated cloud platforms



Cloud solutions

Office 365 (SaaS)
Dynamics CRM Online (SaaS)
EC2, RDS
Windows Azure platform (PaaS)

Private Cloud



Implementation of cloud services on resources that are dedicated to your organization, whether they exist on-premises or off-premises



Typically, your organization owns and controls the resources/assets, definition of services, costs and risks

Variations exist – such as hosted and virtual private clouds



Cloud solutions

Windows Server 2008 R2 Hyper-V,
System Center (IaaS)

Windows Azure Appliance (PaaS)

Database Services



Private Cloud

- Private clouds are cloud infrastructures that are deployed for a single organization.
- These can be managed internally or externally, but all systems and infrastructure are for the purposes of the organization.
- When considering a private cloud, the biggest decision that a business needs to make is the scope of the needed investment to create the private cloud, as implementation can be very expensive.



Cloud vs. Public Cloud

- More than a location and ownership distinction

- ▶ Private Cloud
 - ▶ Control
 - ▶ Conventional storage
 - ▶ Custom policies
 - ▶ Heterogeneous infrastructure
 - ▶ Regulatory compliance & data sovereignty
- ▶ Public Cloud
 - ▶ Scale
 - ▶ Cloud storage
 - ▶ Common policies
 - ▶ Homogeneous infrastructure
 - ▶ Work in progress



Mixed/blended model of private and public clouds

- Variations and multiple interpretations exist

On-premises and off-premises bridging

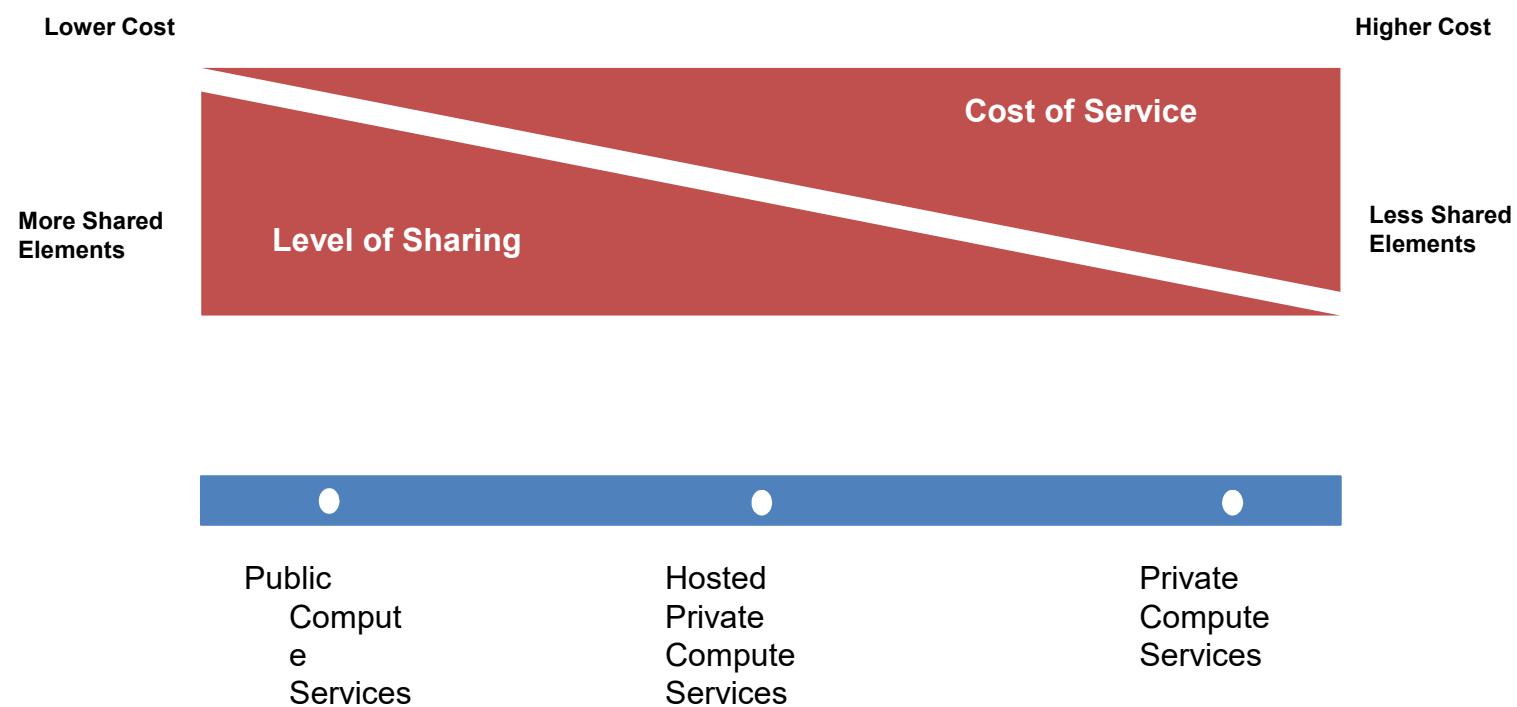
- Most common scenario today
- Especially for large enterprises

More than a deployment / delivery model

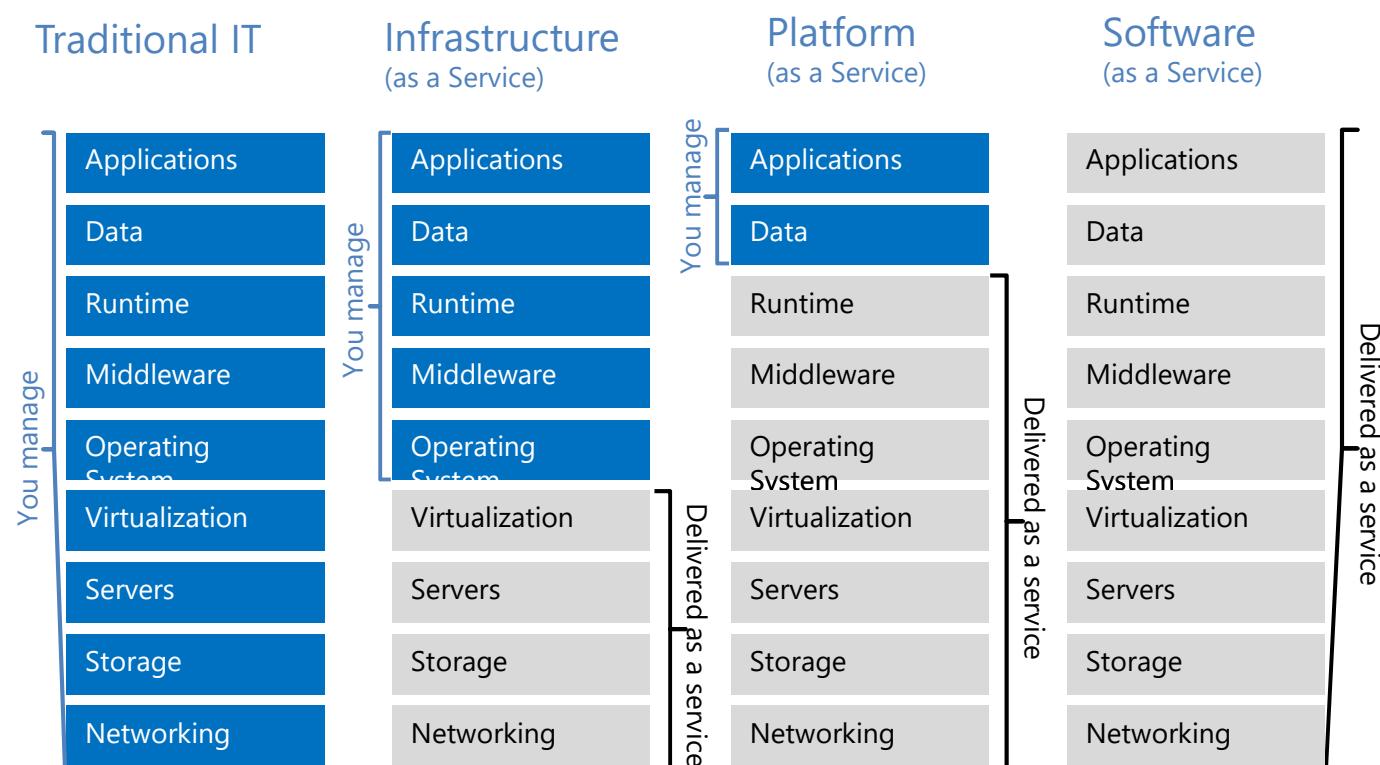
- Application design, architectural model



Product Families and Cost Principles



Cloud Service Models





Cloud Provider List

Cloud providers

The slide displays a grid of logos for various cloud providers. The companies listed include:

- Microsoft
- iCloud
- salesforce
- IBM Cloud
- Google
- Joyent
- RedMonk
- bluelock
- Amazon Web Services
- HP Cloud Services
- CITRIX
- greenqloud
- rackspace
- at&t
- SAP
- Cloud Solutions

Partial list of cloud companies



Is public cloud the best option?

Businesses that are best suited for using the public cloud are ones that need to bring a product/service to market quickly without the internal infrastructure and support to build out their own private cloud system. It does provide smaller companies without IT departments the opportunity to match the deployment speed of larger companies.

Pros and Cons of Public Cloud



Increased network efficiency and more resources

Reduced complexity and lead times (because the architecture is fixed)

Ready-to-go applications used within the public cloud can conform to the demands of business

Disadvantages Of Public Cloud:

Fewer options for customization

Substantially less secure

Fixed architecture cannot (at times) grow with the needs of the business

Is the private cloud the best option?



- Businesses that are best suited for the private cloud are ones that must comply with
- regulatory guidelines or have highly volatile applications needed within the cloud.
- Additionally, these businesses will be required to install their own servers and storage
- hardware that allows for modifications in workloads. Though this can be a significant
- investment, it is required if the business deals with regulated data or must comply with
- industry rules.

Pros and Cons of private Cloud



Advantages Of Private Cloud:

Extensive security options and capabilities, substantially more than the public cloud availability to the internal network and increased access/communication for internal users can grow with a business and be expanded or changed as needed

Disadvantages Of Private Cloud:

Significant level of engagement from both management and IT departments are required

Significant investment is required, both to build the private cloud and maintain/grow it
Does not deliver the short-term solutions – given the required time needed to build it out – that the public cloud does

Pros and Cons of Hybrid Cloud



Advantages Of Hybrid Cloud:

Best of both the public and private cloud in terms of needed resources

Added accessibility for internal and external users

Security parameters are higher than that of the public cloud

Disadvantages Of Hybrid Cloud:

Inherent inefficiency of monitoring several different security platforms

Customization of rules and policies to govern security and the elements of infrastructure that do not link between the public and private cloud

Added security risk



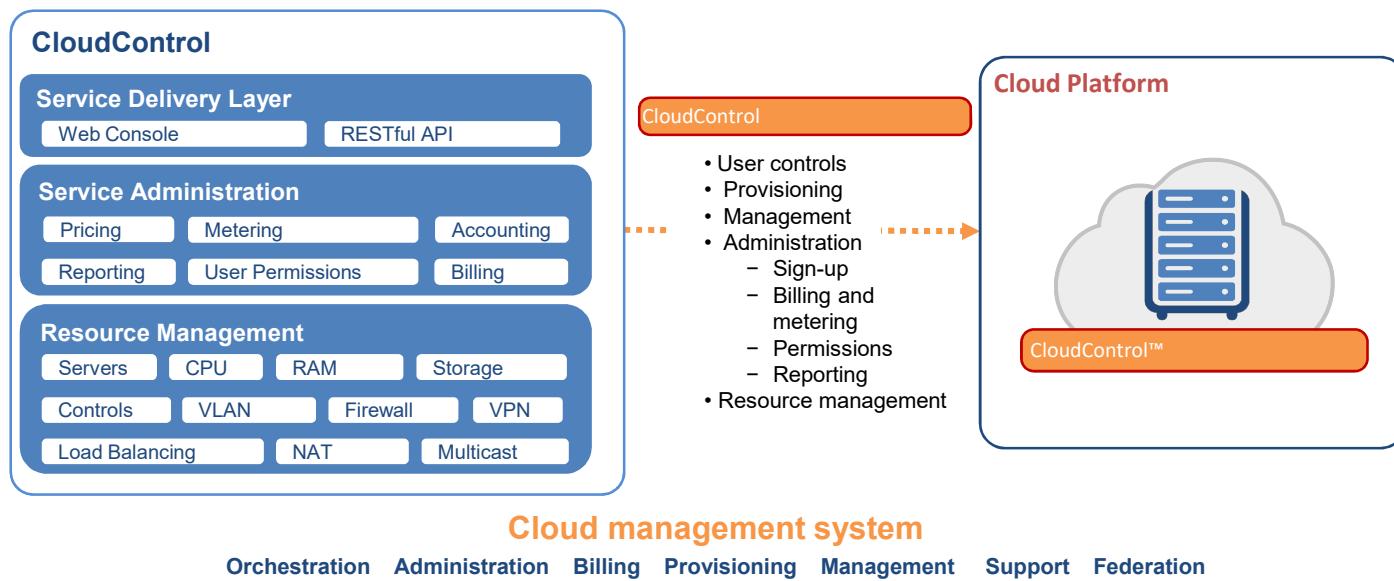
Is hybrid cloud the best option?

Businesses that are best suited to the hybrid cloud are those with sites that have unreliable fluctuations in traffic. With e-commerce, these fluctuations vary greatly between times of day and seasons in the year. Because the hybrid cloud has the same security of the private cloud, businesses can utilize both the added resources and the security to facilitate the needs of their customers. In essence, it is the best of both worlds.



Cloud management system

Addresses the complexity of cloud orchestration, provisioning and billing



Consumption Model



Model	<u>Open Community</u>	<u>Controlled Open Mode</u>	<u>Contractual Open</u>	<u>Public/Private Hybrid</u>	<u>Private Closed</u>
Examples	Facebook Twitter LinkedIn MyFitnessPal Google Groups	IBM SmartCloud Enterprise Amazon Web Services RackSpace OpSour	Salesforce.com Workday MailChimp QuickBooks Online	IBM SmartCloud HP Cloud Service Microsoft Azure	Internal but can be implemented by a third-party vendor
Characteristics	No SLA	Simple SLA	SLA with no indemnification	SLA guaranteeing uptime	Explicit SLA
	No Contract	Transactional pricing	Contract	Contract	Capital expense with ongoing maintenance
	Simple Password Protection	More security	High security provided	Highest level of security	Secure platform
	No governance model	No explicit governance	Governance in place	Explicit governance	Explicit governance



Use Cases

- Elastic/burst capacity (e.g., apps with variable load, HPC / parallel processing, etc.)
- Temporal applications (e.g., marketing apps, test & QA environments, etc.)
- Cloud-based DMZ / Perimeter Zone
- High Performance Compute
- Backup and storage
- Disaster recovery

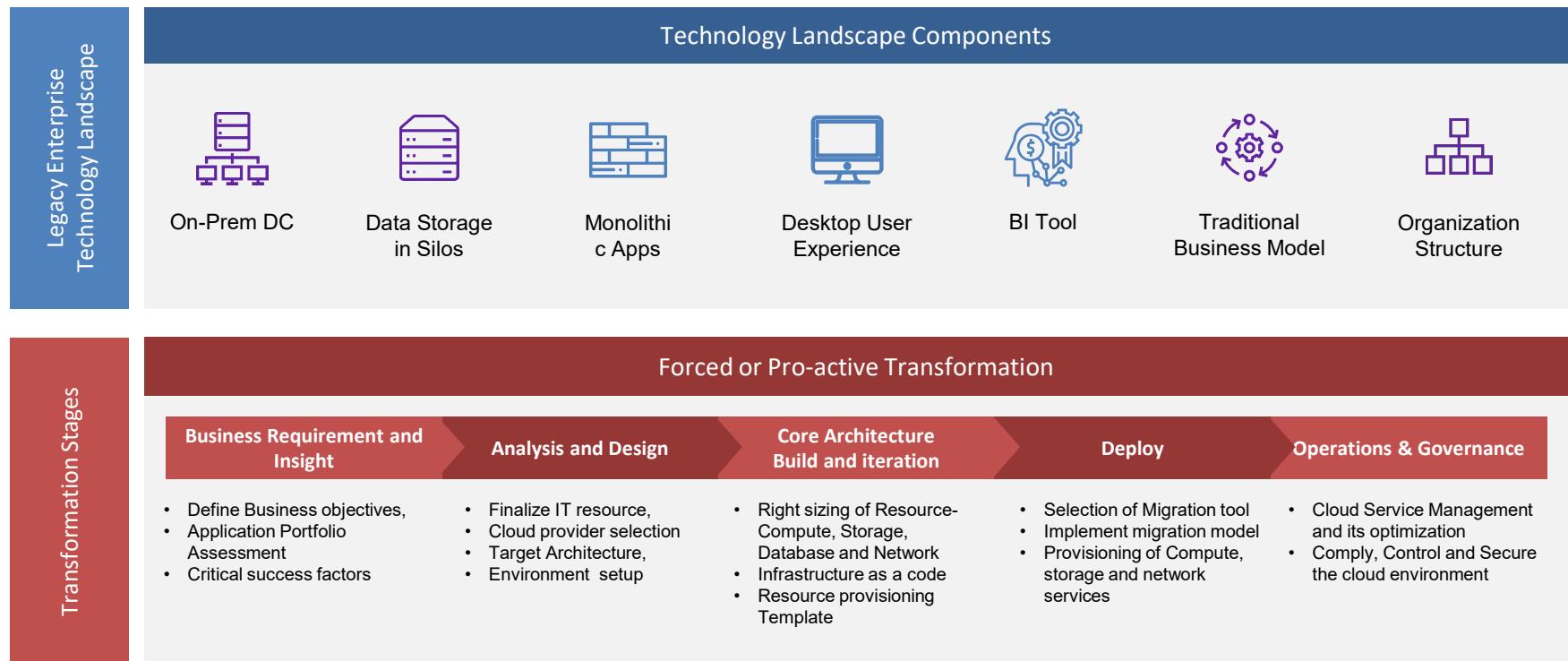
These work, but they are still deployment models



Topics of Interest as GF

Course ID	Course Description
DSE*ZG522	Big Data Systems What is big data - are existing systems sufficient?; Data Warehouse v/s Data Lakes; Hadoop – Components; Storage - Relational DBs/ NoSQL dbs / HDFS / HBase / Object Data stores - S3; Serialization; Interfaces - Hive/ Pig; Stream Processing; Spark; Mahout.
SS ZG527	Cloud Computing Concurrency and distributed computing, message passing over the network, connectivity and failure models, local vs remote connectivity, distributed resource modeling, distributed data models; replication & consistency; virtualization; CPU virtualization, memory and storage virtualization, virtualized networks, computing over WAN and Internet; computing on the cloud, computing models, service models and service contracts, programming on the cloud; Cloud infrastructure, LAN vs Wan issue, resource scaling and resource provisions, performance models, scalability, performance measurement and enhancement techniques; cloud applications and infrastructure services.
IS ZC446	Data Storage Technologies & Networks Storage Media and Technologies – Magnetic, Optical and Semiconductor media, techniques for read/write operations, issues and limitations. Usage and Access – Positioning in the memory hierarchy, Hardware and Software Design for access, Performance issues. Large Storages – Hard Disks, Networked Attached Storage, Scalability issues, Networking issues. Storage Architecture. - Storage Partitioning, Storage System Design, Caching, Legacy Systems. Storage Area Networks – Hardware and Software Components, Storage Clusters/Grids. Storage QoS – Performance, Reliability, and Security issues.
SS ZG515/DSE*ZG515	Data Warehousing Introduction, evolution of data warehousing; decision support systems; goals, benefit, and challenges of data warehousing; architecture; data warehouse information flows; software and hardware requirements; approaches to data warehouse design; creating and maintaining a data warehouse; Online Analytical Processing (OLAP) and multi-dimensional data, multi-dimensional modeling; view materialization; data marts; data warehouse metadata; data mining.
CSI** ZG522	Design and Operation of Data Centers Data Center Design: Principles (Scalability, Reliability, and Elasticity), Components - Computing Infrastructure (Processing, Storage, and Networking) and Physical Infrastructure (Power, Cooling, and Physical Security); Servers – Server Hardening, Server Optimization, Server Deployment and Consolidation, Converged and Hyper-Converged Infrastructure. Application monitoring and maintenance. Networking for data centers – device hardening, bandwidth aggregation, traffic management, redundancy, network isolation, deployment of internal security and peripheral security; Contingency Planning & Disaster Recovery: Backup, recovery, and redundancy/replication technologies and approaches. Data Center Architecture: Private, Public, and Hybrid models; Distributed Data Centers; Introduction to Software Defined DataCenters. Costing and Pricing—Costing and Cost Optimization, Pricing and Economics of Data Center Operation.
	Ethics for Data Science Introduction to data ethics, perils of big data, foundations of data privacy, challenges of privacy in the digital age, data policies, consent and fair usage.
	Infrastructure Management Introduction to System Management IT Infrastructure, Introduction to System Management IT Infrastructure, Staffing Legislation , Ethics, Outsourcing for ITSM, Customer Service, Availability, 6. Performance and Tuning, Production Acceptance, Change Management, Problem Management, Storage Management, Network Management, Configuration Management, Capacity Planning, Strategic Security, Business Continuity, Facilities Management, Developing Robust Processes, Using Technology to Automate and Evaluate Robust Processes, Integrating Systems Management Processes, Special Considerations for Client-Server and Web-Enabled Environments
DSE*ZG523	Introduction to Data Science Context and use of Data Science. Highdimensional data, graphs, vectors in high dimensional space and large matrices; Algorithms for massive data problems, sampling techniques. Techniques for extracting information/patterns from data.
CSI** ZG515	Introduction to DevOps Continual Service - continuous integration and continuous delivery; Scaling: automating infrastructure and infrastructure-as-code; DevOps and Cloud: platform-as-a-service and DevOps, use of virtual machines and containers for deployment, Micro-services; application lifecycle management: deployment pipeline and application deployment,continuous deployment pipeline; stack management - life cycle of stack and events, resource and event monitoring, auto healing; Security: security of deployment pipeline, policy-as- code.
CSI** ZG511	IT Infrastructure Projects & Processes The course introduces infrastructure software project process as a structured methodology for professional software Maintenance and infrastructure. Introduction to Software process, Introduction to ITIL, CMMI – SVC – Capacity Maturity Model integrated for Services.

Cloud Adoption Journey – Enterprise Landscape



Key Cloud Adoption Drivers

 Global Scale

Worldwide reach, Hyperscale & Elasticity with **economic benefits**

 Digital User Experience

Omni channel experience on **Mobile** platforms & enabling user **Mobility**

 Reliability & Resiliency

Reliable trading - High Availability, Dynamic Failover, COVID lessons learnt

 Compliance & Security

Regularity Compliance, **Data Protection** & **confidentiality**

 IT Simplification

Accelerated **Time to Market** - DevOps, DevSecOps, Managed PaaS

 Driving Innovation

SaaS leverage - **Microservices, Analytics, AI, Blockchain and IoT** enabling business workflows



Cloud Adoption Journey Road Map



Cloud Adoption Delivery Framework

Customers do business in a heterogeneous World so DXC Luxoft is committed to delivering the best customer experience across software, services, and support in their cloud adoption journey



Cloud Adoption Goals

- Empower enterprise mobility
- Transform the datacenter
- Enable application innovation
- Unlock insights on any data
- Support application lifecycle management



Cloud Adoption Approach

- Define how to select and prioritize applications for migrations.
- Understand proven best practices to migrate applications to cloud Platform
- Get Exposure to variety of application migration patterns – ranging from cloud Server Migration/Data Migration, to App Code* for CI/CD.
- Validate Landing Zone, Operations run book and Security playbook by actually testing applications in Cloud platform



Cloud Adoption Phases

- Planning and Discovery
- Design and Build
- Migrate
- Operation
- Governance



Cloud Adoption Tools

- Discovery tools
- Migration tools
- Monitoring tools
- DevOps Tools
- Project Management Tools



Cloud Adoption Key Success

- Cost Optimization
- Automation and Orchestration
- Improve operation efficiency and performance
- Reduce Risk
- Increase agility

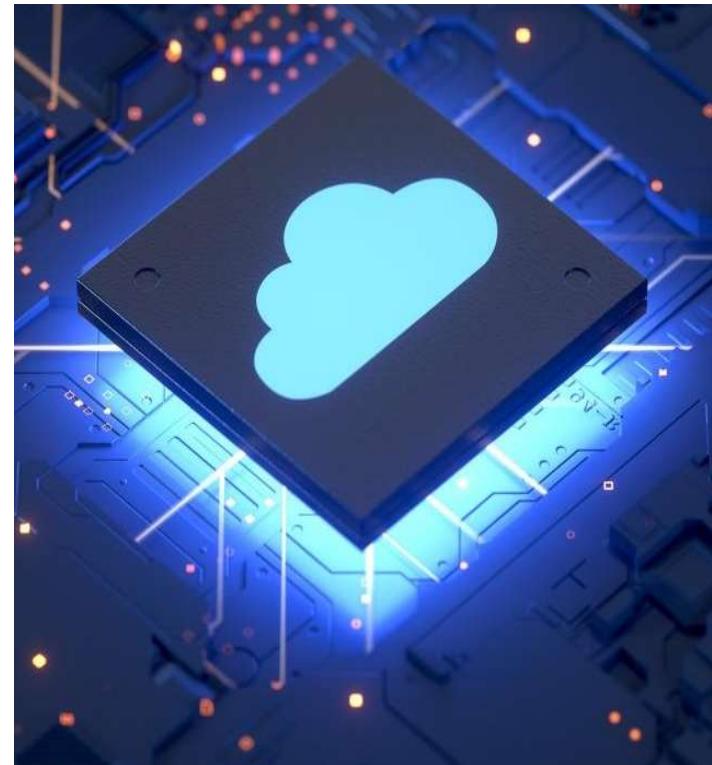
Cloud Adoption

Framework (CAF)

The Cloud Adoption Framework brings together cloud adoption best practices. It provides a set of tools, guidance, and narratives that would help shape technology, infrastructure, and people strategies for driving your desired business outcomes during your cloud adoption effort.

The cloud adoption framework objectives includes:

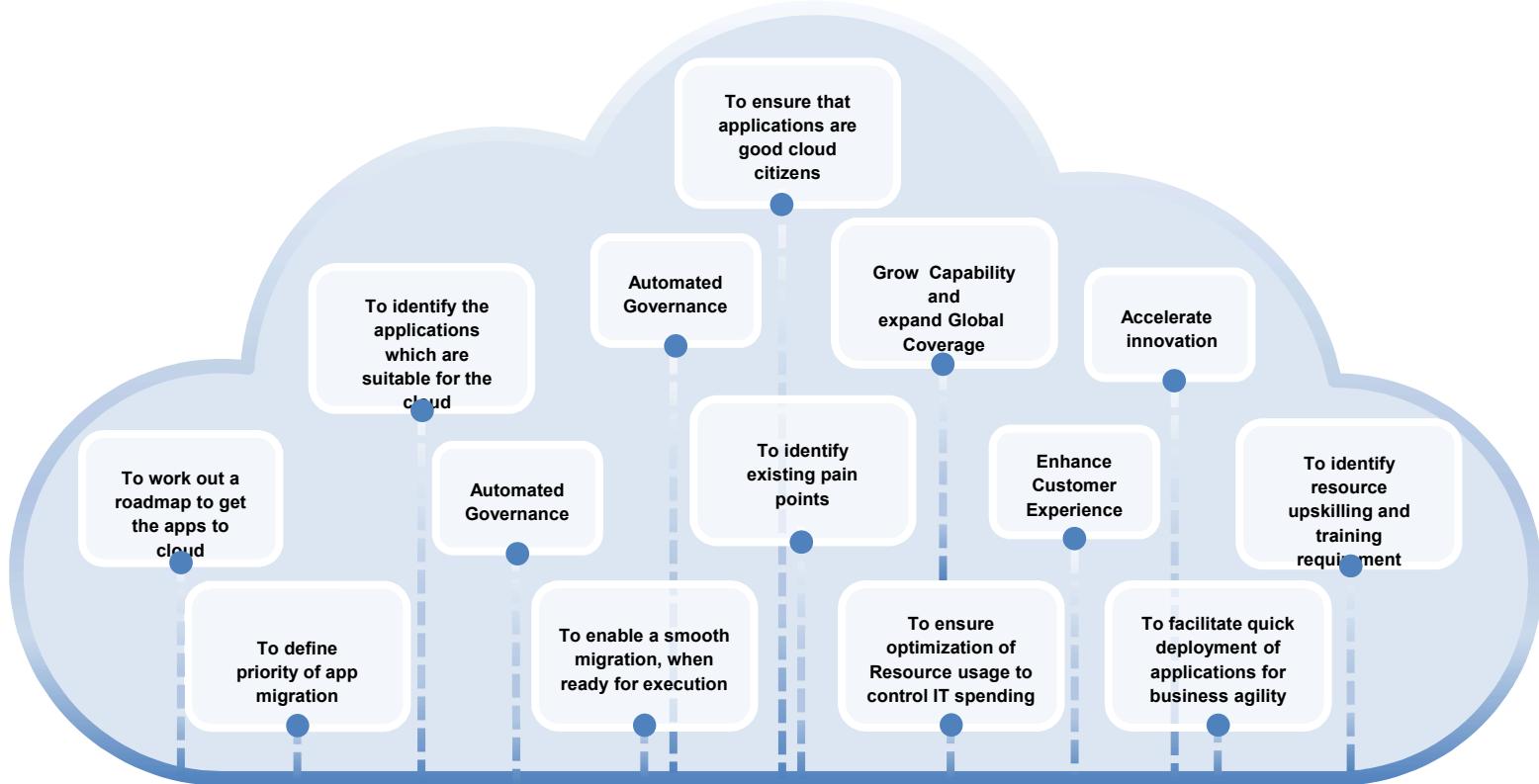
- The Cloud Adoption Framework provides tools and guidance for implementing cloud technologies to incorporate business, people and process changes,
- "Cloud Adoption framework" is used to describe collections of development tools to middleware to database services that ease the creation, deployment and management of cloud applications
- Aligns cloud adoption with business objectives across the cloud adoption stages.
- CAF Standardize technology adoption to reduce technology debt and streamlines cloud services management.
- CAF ensures security of infrastructure, applications, and data, while ensuring data sovereignty.
- CAF drives audit readiness for infrastructure applications.
- CAF allows periodical review of the reference architecture, approved list of services, security controls and cost optimization techniques.



Slide 59

- VL1** The framework should have objectives like:
how to adopt cloud
how to speed up adoption
how to make applications compliant
I think we should incorporate and have a think about this. I think you have it right on slide 5 so we should decide which to use. I think I actually prefer this slide now, with some updated objectives.
The objectives we have currently are more around the benefits of a migration?
Vadgama, Vishal (DXC Luxoft), 7/15/2021
- ML1** made changes as per your suggestions
Moitra, Mridul (DXC Luxoft), 7/16/2021
- ML2** [@Vadgama, Vishal (DXC Luxoft)]
made changes as per your suggestions
Moitra, Mridul (DXC Luxoft), 7/16/2021

Why do we need a Cloud Adoption Framework





Evolution of Web

Explosive growth in applications:

biomedical informatics, space exploration, business analytics,
web 2.0 social networking: YouTube, Facebook

Extreme scale content generation: e-science and e-business data deluge

Extraordinary rate of digital content consumption: digital gluttony:

Apple iPhone, iPad, Amazon Kindle, Android, Windows Phone

Exponential growth in compute capabilities:

multi-core, storage, bandwidth, virtual machines (virtualization)

Very short cycle of obsolescence in technologies:

Windows 8, Ubuntu, Mac; Java versions; C → C#; Python

Newer architectures: web services, persistence models, distributed file systems/repositories (Google, Hadoop), multi-core, wireless and mobile

Diverse knowledge and skill levels of the workforce



In [software engineering](#), **SOA (service-oriented architecture)** is an architectural style that focus on discrete services instead of a monolithic design.^[1] By consequence, it is as well applied in the field of [software design](#) where services are provided to the other components by [application components](#), through a [communication protocol](#) over a network. A service is a discrete unit of functionality that can be accessed remotely and acted upon and updated independently, such as retrieving a credit card statement online. SOA is also intended to be independent of vendors, products and technologies.

A **web service** is any piece of software that makes itself available over the internet and uses a standardized XML messaging system



BITS Pilani
Pilani Campus

BITS Pilani presentation

Mridul Moitra
Cloud Computing



BITS Pilani
Pilani Campus



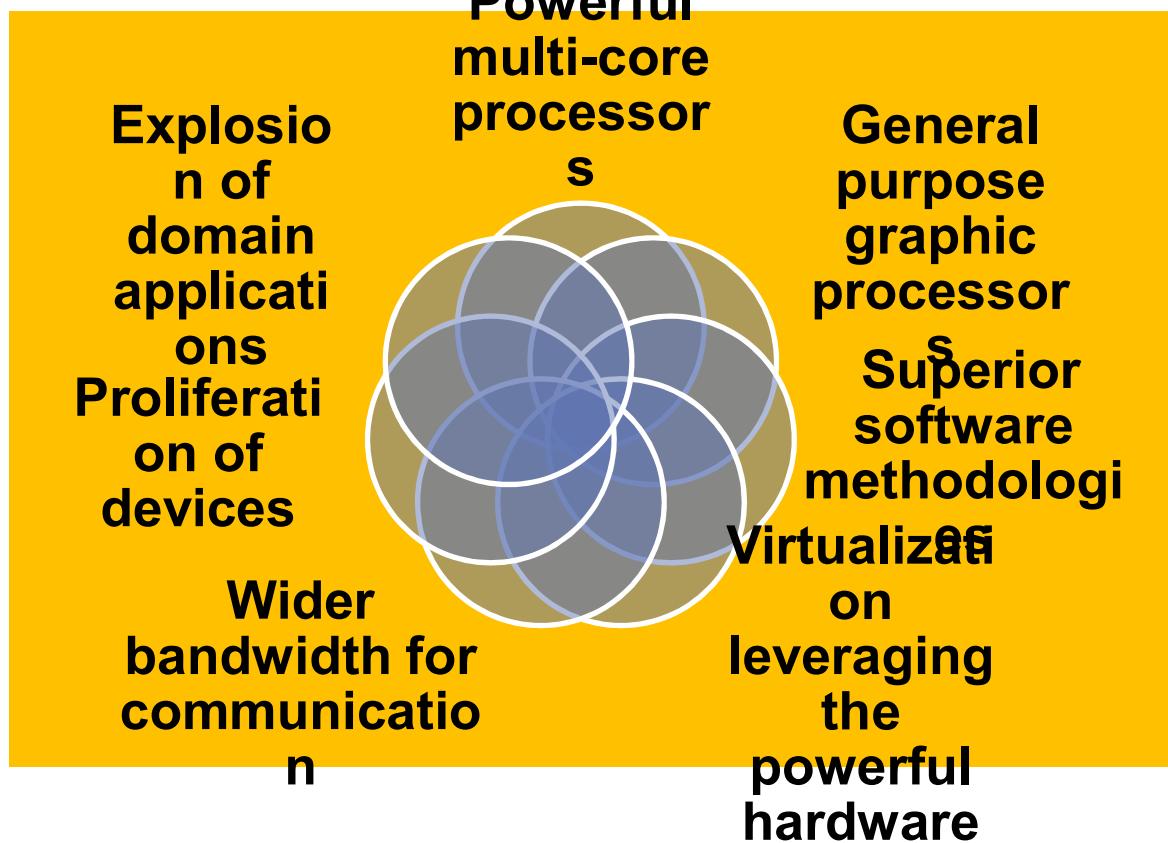
<CSI ZG527 / SS ZG527 / SE ZG527 **Cloud Computing** **Lecture No. 1**

Introduction to Cloud Computing, services and deployment models



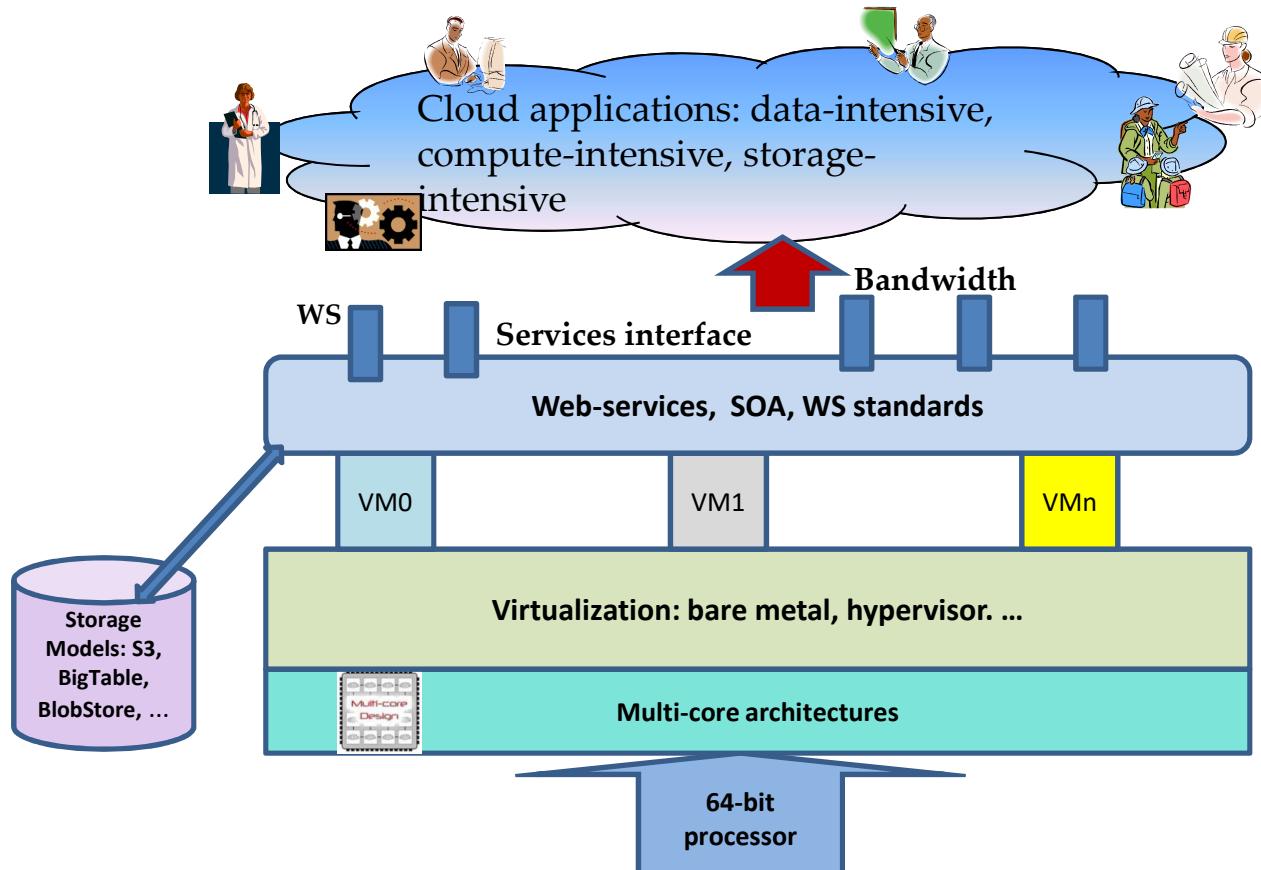
- **Agenda**
 1. **Introduction to Cloud Computing – Origins and Evolution**
 2. **Characteristics of cloud platform**
 3. **Types of Clouds and Services**
 4. **Cloud Delivery Model**
-

Motivation



- 1. Web Scale Problems**
- 2. Web 2.0 and Social Networking**
- 3. Information Explosion**
- 4. Mobile Web**

Technology Advances



Evolution of cloud computing

?

- **The evolution of cloud computing can be bifurcated into three basic phases:**
- **1. The Idea Phase-** This phase inceptioned in the early 1960s with the emergence of utility and grid computing and lasted till pre-internet bubble era. Joseph Carl Robnett Licklider was the founder of cloud computing.
- **2. The Pre-cloud Phase-** The pre-cloud phase originated in 1999 and extended to 2006. In this phase the internet as the mechanism to provide Application as Service.
- **3. The Cloud Phase-** The much talked about real cloud phase started in the year 2007 when the classification of IaaS, PaaS, and SaaS got formalized. The history of cloud computing has witnessed some very interesting breakthroughs launched by some of the leading computer/web organizations of the world.



What is Cloud Computing?

Cloud Computing is a general term used to describe a new class of network based computing that takes place over the Internet,

- basically a step on from Utility Computing
- a collection/group of integrated and networked hardware, software and Internet infrastructure (called a platform).
- Using the Internet for communication and transport provides hardware, software and networking services to clients

These platforms hide the complexity and details of the underlying infrastructure from users and applications by providing very simple graphical interface or API (Applications Programming Interface).

What is Cloud Computing cont....



the platform provides

- on-demand services, that are always on, anywhere, anytime and any place.
- Pay for use and as needed, elastic
- scale up and down in capacity and functionalities
- The hardware and software services are available to
- The hardware and software services are available to
- general public, enterprises, corporations and businesses markets

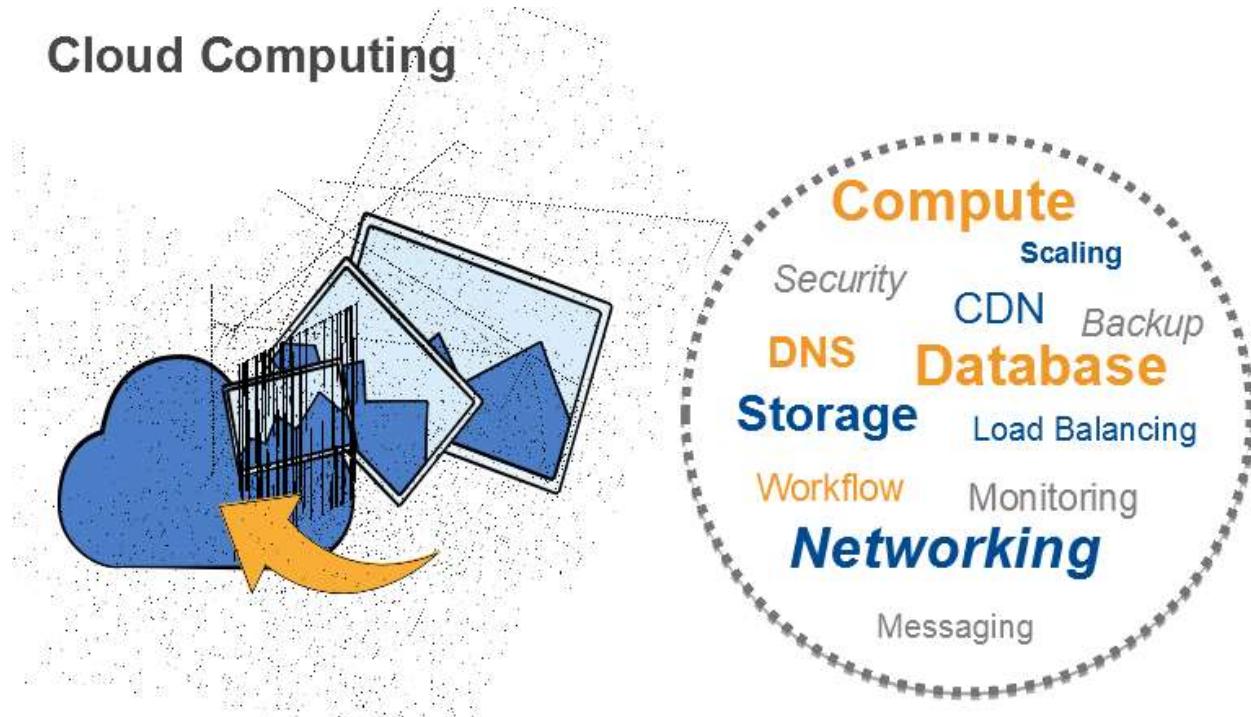


Cloud Computing: Definition

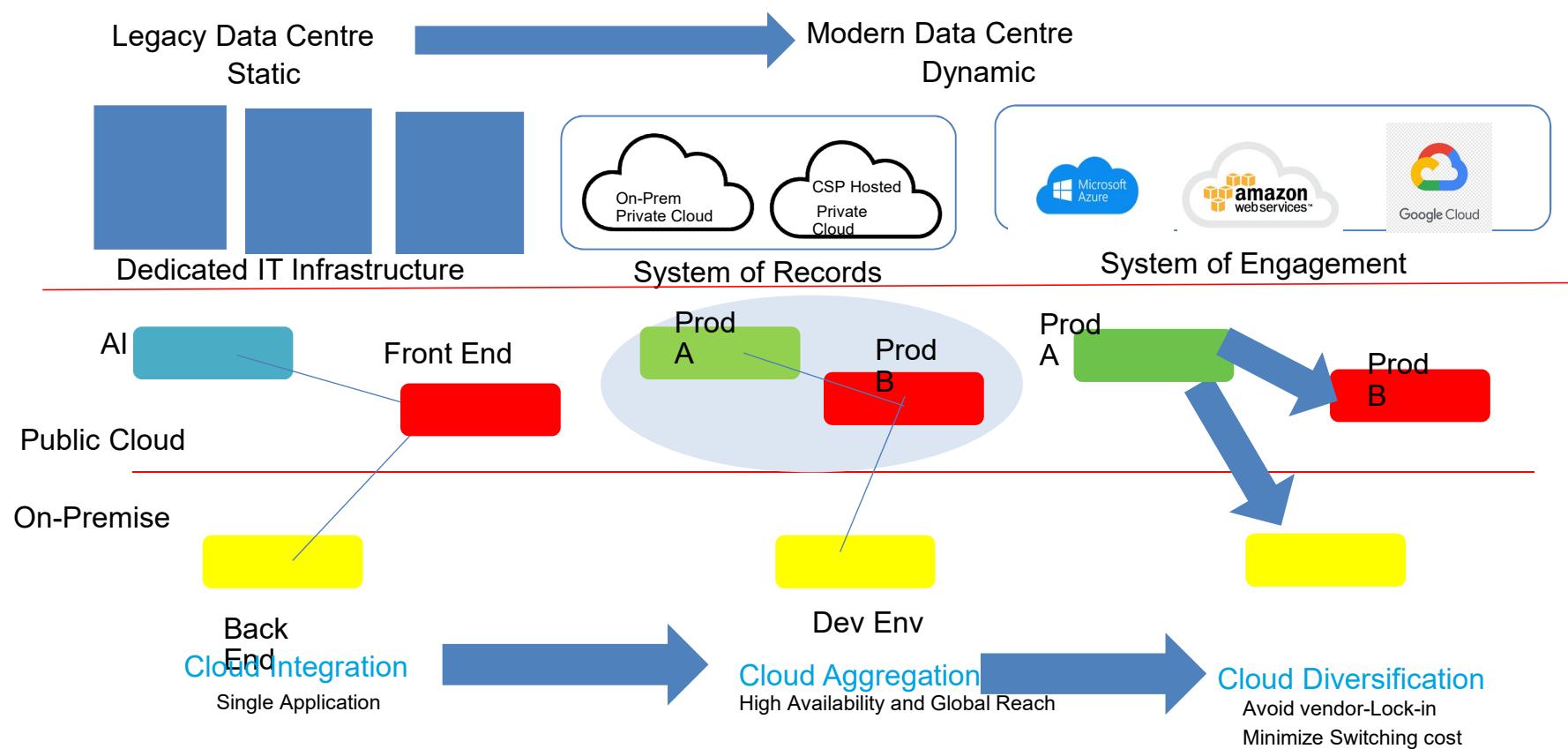
The US National Institute of Standards (NIST) defines cloud computing as follows:

Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.

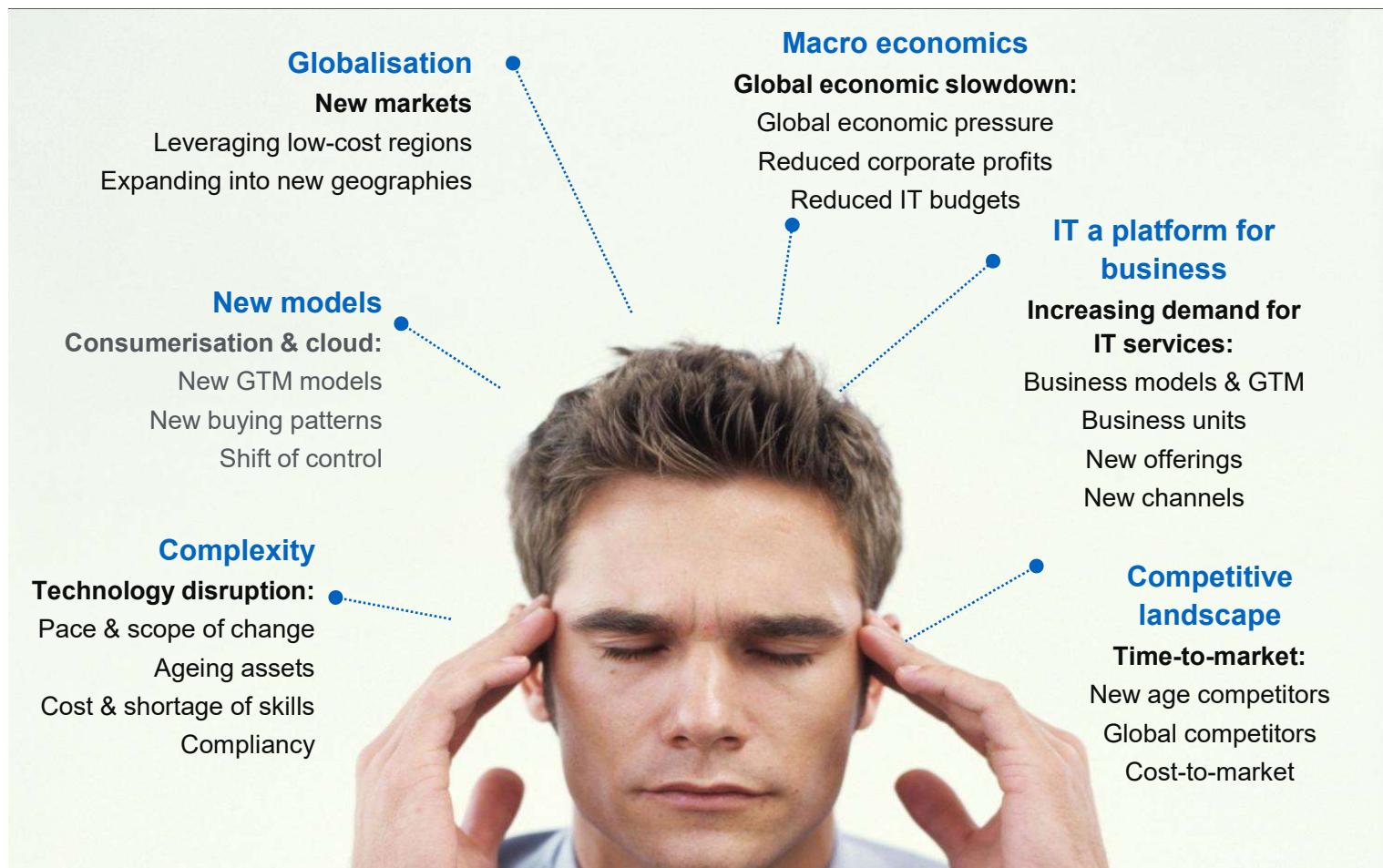
Cloud Computing



Cloud Transformation Journey



Challenges Of the CIO





Drivers for the new Platform

Generational Shift of Computing Platform

Technology	Economic	Business
	Centralized compute & storage, thin clients	Optimized for efficiency due to high cost
	PCs and servers for distributed compute, storage, etc.	Optimized for agility due to low cost
	Large DCs, commodity HW, scale-out, devices	Perpetual license for OS and application software
	Order of magnitude better efficiency and agility	Pay as you go, and only for what you use

<http://blogs.technet.com/b/yungchou/archive/2011/03/03/chou-s-theories-of-cloud-computing-the-5-3-2-principle.aspx>

Cloud Computing Business Drivers



Cost optimisation



- No capex, less assets
- Pay-as-you-use
- On-demand capacity
- Elasticity
- Economies of scale
- Time-to-value

Risk optimisation



- Business continuity
- Technology independence
- Operational complexity
- Specialised skills

Strategic agility



- Time-to-market
- Innovation
- New business models
- Resource leverage
- Adaptability
- Flexibility

...why would one not consider these benefits?

3-4-5 rule of Cloud Computing



NIST specifies 3-4-5 rule of Cloud Computing

- 3** cloud service models or service types for any cloud platform
- 4** deployment models
- 5** essential characteristics of cloud computing infrastructure

Cloud Summary



- Shared pool of configurable computing resources
- On-demand network access
- Provisioned by the Service Provider



Cloud Summary...

Cloud computing is an umbrella term used to refer to Internet based development and services

A number of characteristics define cloud data, applications services and infrastructure:

Remotely hosted: Services or data are hosted on remote infrastructure.

Ubiquitous: Services or data are available from anywhere.

Commodity model: The result is a utility computing model similar to traditional that of traditional utilities, like gas and electricity - you pay for what you would want!

Characteristics of Cloud Computing



5 Essential Characteristics of Cloud Computing

Ref: The NIST Definition of Cloud Computing

<http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf>



Source: <http://aka.ms/532>

- On demand self-service
- Broad network access
- Resource pooling
- Rapid elasticity
- Measured service

Characteristics of Cloud Computing



Essential Characteristics

- On-demand self-service
- Broad network access
- Resource pooling
- Rapid elasticity
- Measured service

Service Models

- Software as a Service
- Platform as a Service
- Infrastructure as a Service

Deployment Models

- Private
- Public
- Hybrid
- Community

Traditional Infrastructure

Amazon Web Services



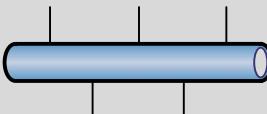
Security



Security Groups

NACLs

Access Mgmt



Networking



Servers



Amazon EC2
Instances



RDBMS

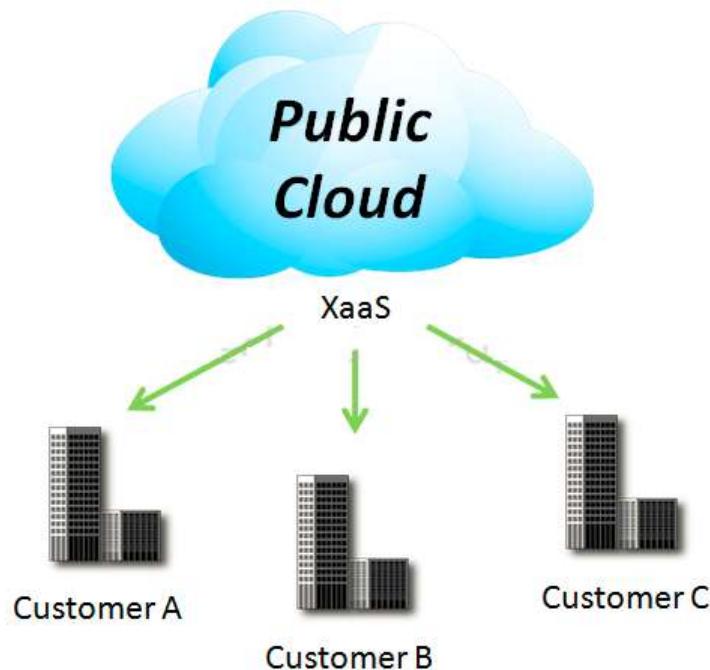
Storage &
Database



4 Deployment Models



1. Public Cloud

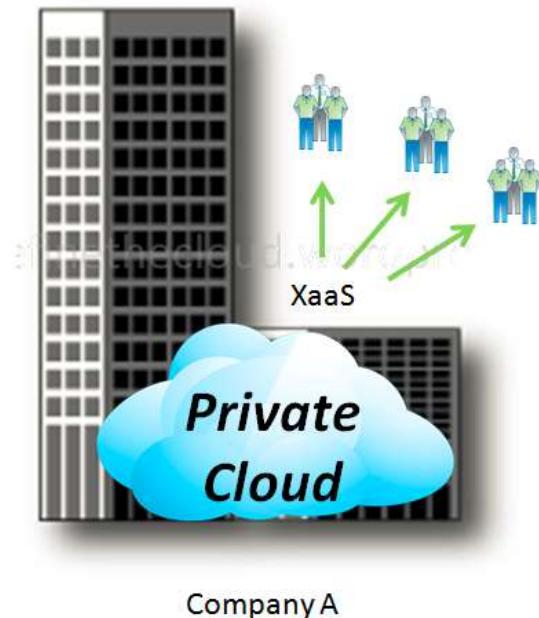


Mega-scale cloud infrastructure is made available to the general public or a large industry group and is owned by an organization selling cloud services.

4 Deployment Models



2. Private Cloud

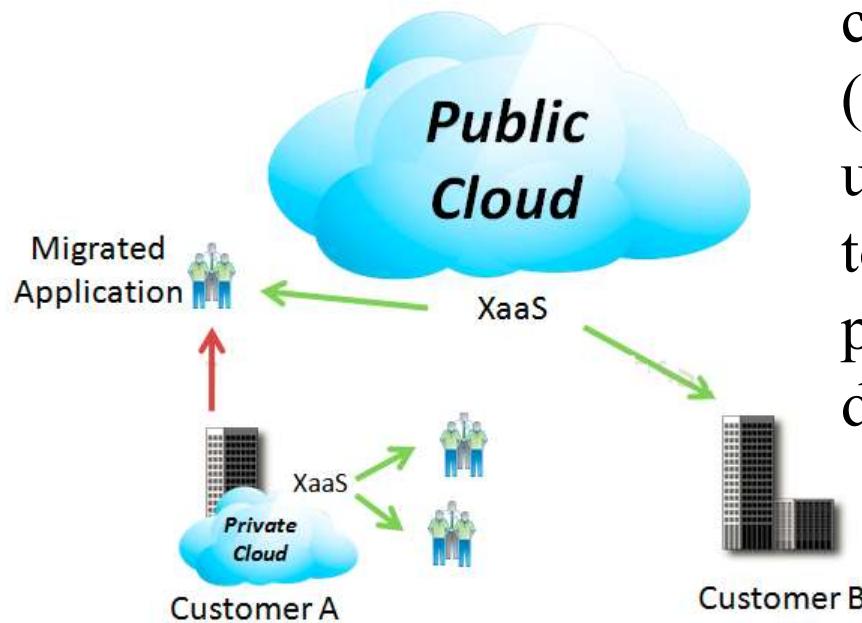


The cloud infrastructure is operated solely for an organization. It may be managed by the organization or a third party and may exist on premise or off premise.

4 Deployment Models



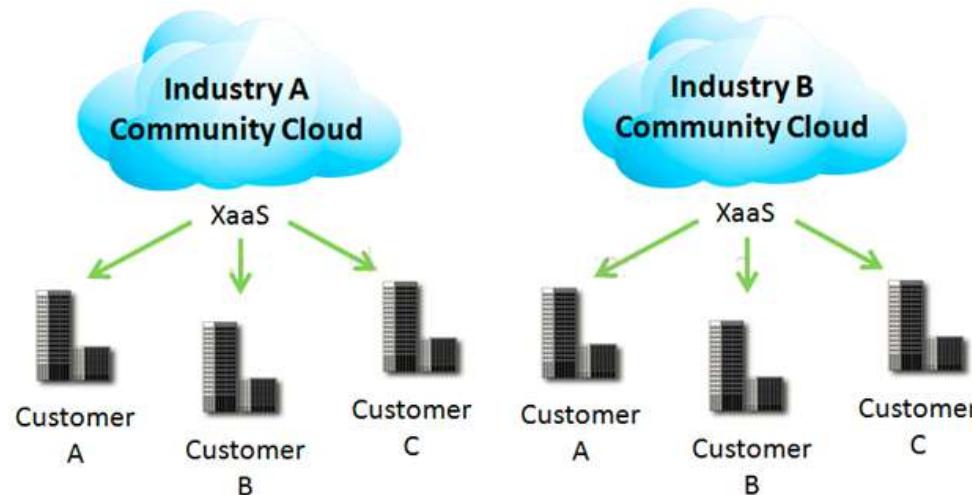
3. Hybrid Cloud



The cloud infrastructure is a composition of two or more clouds (private or public) that remain unique entities but are bound together by standardized or proprietary technology that enables data and application portability

4 Deployment Models

4. Community Cloud



Community Clouds are when an ‘infrastructure is shared by several organizations and supports a specific community that has shared concerns (e.g., mission, security requirements, policy, and compliance considerations). It may be managed by the organizations or a third party and may exist on premise or off premise’ according to NIST. A community cloud is a cloud service shared between multiple organizations with a common tie/goal/objective. E.g. OpenCirrus

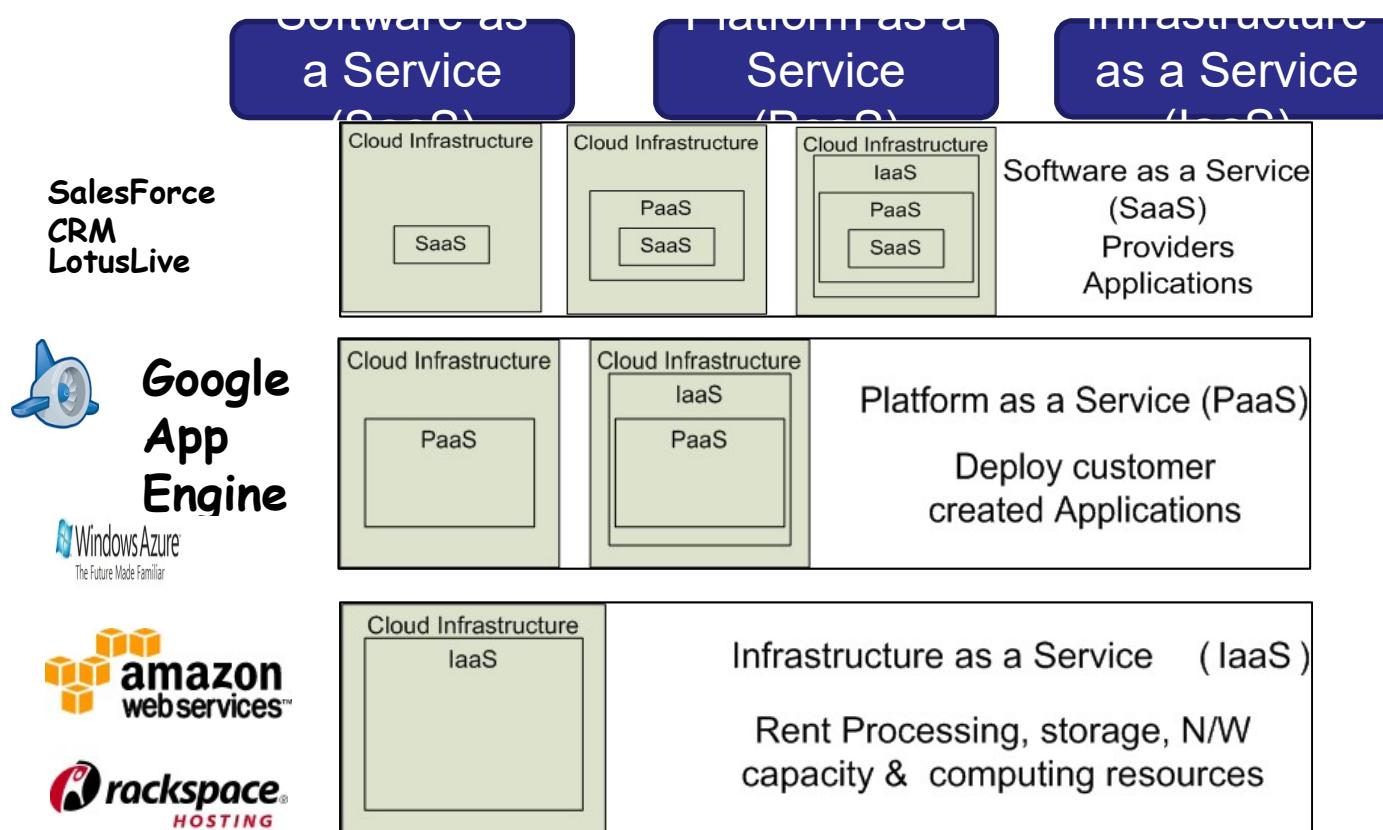
Introduction to Cloud Computing, services and deployment models



- **Agenda**
 1. **Introduction to Cloud Computing – Origins and Motivation**
 2. **3-4-5 rule of Cloud Computing**
 3. **Types of Clouds and Services**
 4. **Cloud Infrastructure and Deployment**
-



3 Cloud Service Models





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.

E.g. salesforce.com , Google Apps



Platform as a Service

Platform as a service encapsulates a layer of software and provides it as a service that can be used to build higher-level services.

2 Perspectives for PaaS :-

- 1. Producer:-** 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.
- 2. Consumer:-** 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.



Infrastructure as a Service

Infrastructure as a service delivers basic storage and computing 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.



Service Models Summary

Cloud Software as a Service (SaaS)

The **capability provided to the consumer is to use the provider's applications** running on a cloud infrastructure and accessible from various client devices through a thin client interface such as a Web browser (e.g., web-based email). The consumer does not manage or control the underlying cloud infrastructure, network, servers, operating systems, storage, or even individual application capabilities, with the possible exception of limited user-specific application configuration settings.

Cloud Platform as a Service (PaaS)

The **capability provided to the consumer is to deploy onto the cloud infrastructure consumer-created applications using programming languages and tools supported by the provider** (e.g., Java, Python, .Net). The consumer does not manage or control the underlying cloud infrastructure, network, servers, operating systems, or storage, but the consumer has control over the deployed applications and possibly application hosting environment configurations.

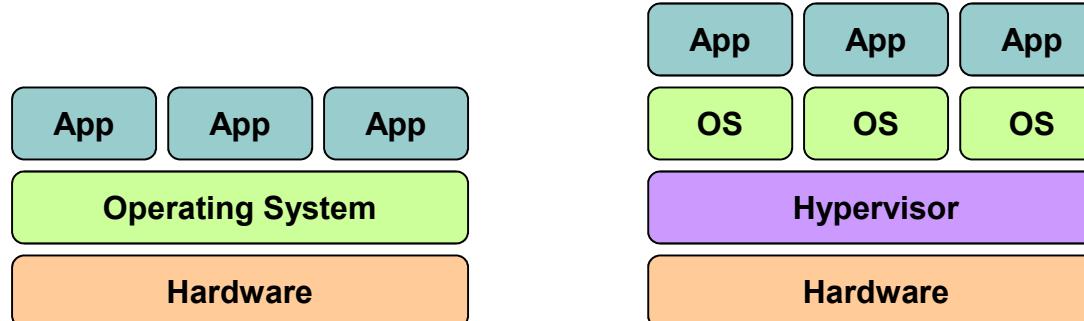
Cloud Infrastructure as a Service (IaaS)

The **capability provided to the consumer is to rent processing, storage, networks, and other fundamental computing resources** where the consumer is able to deploy and run arbitrary software, which can include operating systems and applications. The consumer does not manage or control the underlying cloud infrastructure but has control over operating systems, storage, deployed applications, and possibly select networking components (e.g., firewalls, load balancers).

Cloud Infrastructures



Key Technology is Virtualization



Virtualization plays an important role as an enabling technology for datacentre implementation by abstracting compute, network, and storage service platforms from the underlying physical hardware

Cloud Providers Characteristics



- Provide on-demand provisioning of computational resources
- Use virtualization technologies to lease these resources
- Provide public and simple remote interfaces to manage those resources
- Use a pay-as-you-go cost model, typically charging by the hour
- Operate data centers large enough to provide a seemingly unlimited amount of resources to their clients



Management of Virtualized Resources

Distributed Management of Virtual Machines

Reservation-Based Provisioning of Virtualized Resources

Provisioning to Meet SLA Commitments



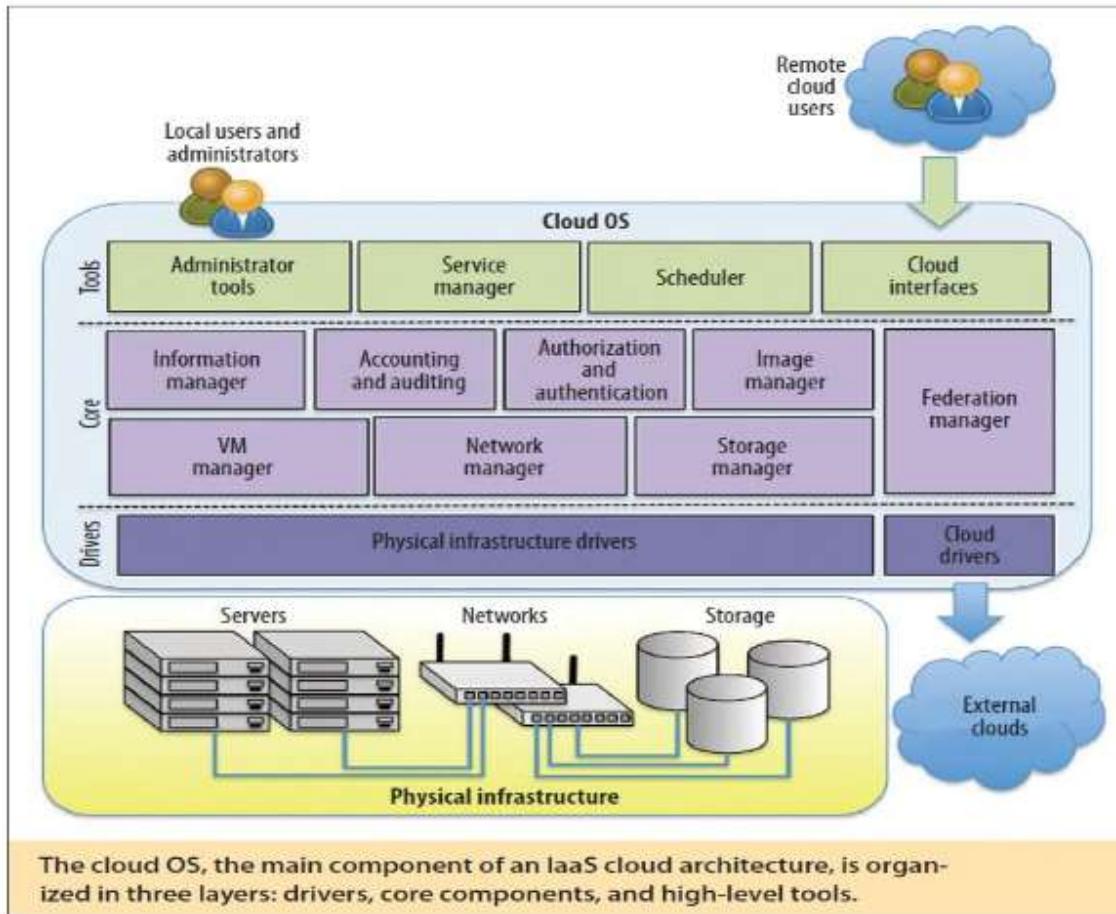
Cloud Infrastructure Anatomy

The key component of an IaaS cloud architecture is the cloud OS, which manages the physical and virtual infrastructures and controls the provisioning of virtual resources according to the needs of the user services

A cloud OS's role is to efficiently manage datacenter resources to deliver a flexible, secure, and isolated multitenant execution environment for user services that abstracts the underlying physical infrastructure and offers different interfaces and APIs for interacting with the cloud

While local users and administrators can interact with the cloud using local interfaces and administrative tools that offer rich functionality for managing, controlling, and monitoring the virtual and physical infrastructure, remote cloud users employ public cloud interfaces that usually provide more limited functionality

The Cloud OS



The cloud operating system is responsible for:

1. managing the physical and virtual infrastructure,
2. orchestrating and commanding service provisioning and deployment
3. providing federation capabilities for accessing and deploying virtual resources in remote cloud infrastructures

Pros and Cons of Cloud Computing



- Why is it becoming a Big Deal:
 - Using high-scale/low-cost providers,
 - Any time/place access via web browser,
 - Rapid scalability; incremental cost and load sharing,
 - Can forget need to focus on local IT.
- Concerns:
 - Performance, reliability, and SLAs,
 - Control of data, and service parameters,
 - Application features and choices,
 - Interaction between Cloud providers,
 - No standard API – mix of SOAP and REST!
 - Privacy, security, compliance, trust...

Public Cloud



Implementation of cloud services on resources that are shared between many customers, managed off-premises



Typically, cloud provider owns and controls the resources/assets, definition of services, costs and risks

Variations exist – such as hosters and integrated cloud platforms



Cloud solutions

Office 365 (SaaS)
Dynamics CRM Online (SaaS)
EC2, RDS
Windows Azure platform (PaaS)

Private Cloud



Implementation of cloud services on resources that are dedicated to your organization, whether they exist on-premises or off-premises



Typically, your organization owns and controls the resources/assets, definition of services, costs and risks

Variations exist – such as hosted and virtual private clouds



Cloud solutions

Windows Server 2008 R2 Hyper-V,
System Center (IaaS)

Windows Azure Appliance (PaaS)

Database Services



Private Cloud

- Private clouds are cloud infrastructures that are deployed for a single organization.
- These can be managed internally or externally, but all systems and infrastructure are for the purposes of the organization.
- When considering a private cloud, the biggest decision that a business needs to make is the scope of the needed investment to create the private cloud, as implementation can be very expensive.



Cloud vs. Public Cloud

- More than a location and ownership distinction

- ▶ Private Cloud
 - ▶ Control
 - ▶ Conventional storage
 - ▶ Custom policies
 - ▶ Heterogeneous infrastructure
 - ▶ Regulatory compliance & data sovereignty
- ▶ Public Cloud
 - ▶ Scale
 - ▶ Cloud storage
 - ▶ Common policies
 - ▶ Homogeneous infrastructure
 - ▶ Work in progress



Mixed/blended model of private and public clouds

- Variations and multiple interpretations exist

On-premises and off-premises bridging

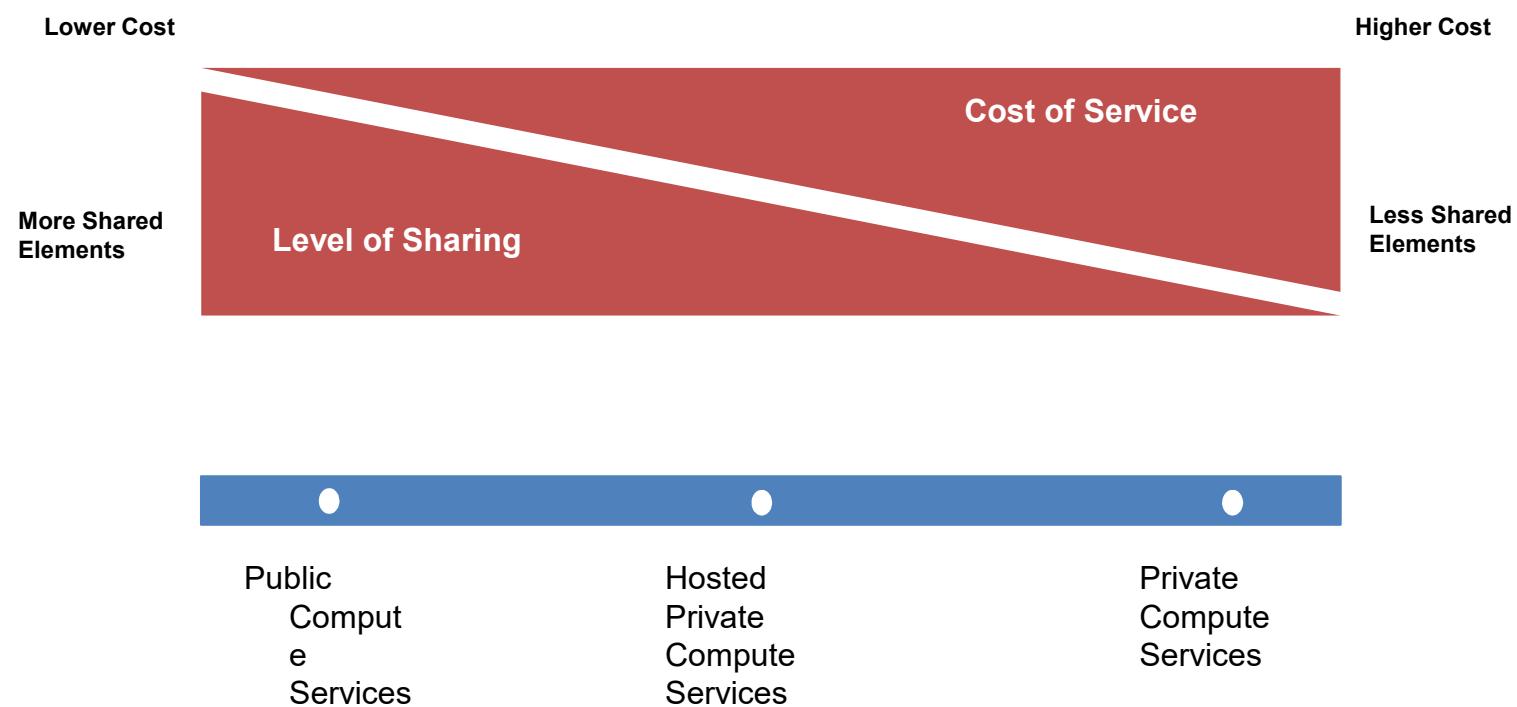
- Most common scenario today
- Especially for large enterprises

More than a deployment / delivery model

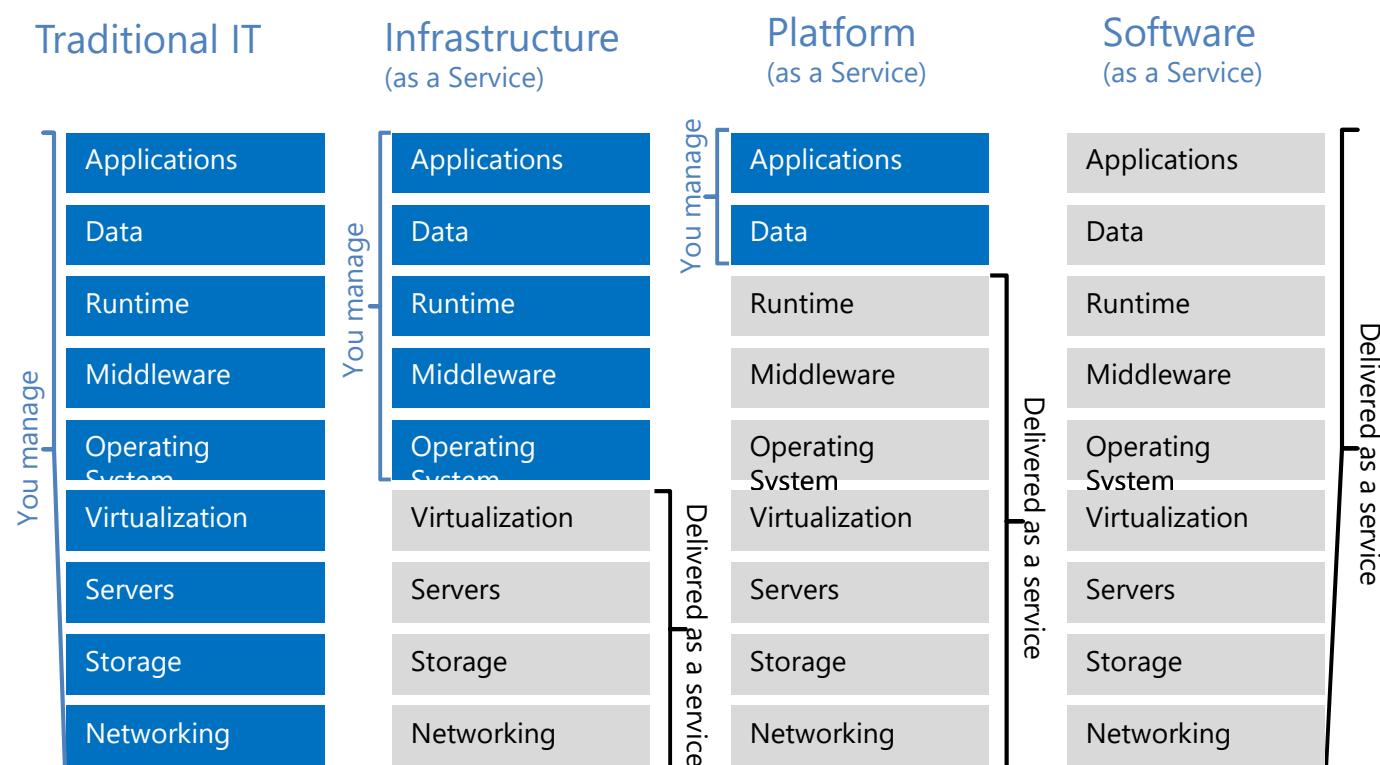
- Application design, architectural model



Product Families and Cost Principles



Cloud Service Models





Cloud Provider List

Cloud providers

The image displays a collection of logos for various cloud service providers, arranged in a grid. The companies listed include:

- Microsoft
- iCloud
- salesforce
- IBM Cloud
- Google
- Joyent
- RedMonk
- bluelock
- Amazon Web Services
- HP Cloud Services
- CITRIX
- greenqloud
- rackspace
- at&t
- SAP
- Cloud Solutions
- Microsoft

Partial list of cloud companies



Is public cloud the best option?

Businesses that are best suited for using the public cloud are ones that need to bring a product/service to market quickly without the internal infrastructure and support to build out their own private cloud system. It does provide smaller companies without IT departments the opportunity to match the deployment speed of larger companies.

Pros and Cons of Public Cloud



Increased network efficiency and more resources

Reduced complexity and lead times (because the architecture is fixed)

Ready-to-go applications used within the public cloud can conform to the demands of business

Disadvantages Of Public Cloud:

Fewer options for customization

Substantially less secure

Fixed architecture cannot (at times) grow with the needs of the business

Is the private cloud the best option?



- Businesses that are best suited for the private cloud are ones that must comply with
- regulatory guidelines or have highly volatile applications needed within the cloud.
- Additionally, these businesses will be required to install their own servers and storage
- hardware that allows for modifications in workloads. Though this can be a significant
- investment, it is required if the business deals with regulated data or must comply with
- industry rules.

Pros and Cons of private Cloud



Advantages Of Private Cloud:

Extensive security options and capabilities, substantially more than the public cloud availability to the internal network and increased access/communication for internal users can grow with a business and be expanded or changed as needed

Disadvantages Of Private Cloud:

Significant level of engagement from both management and IT departments are required

Significant investment is required, both to build the private cloud and maintain/grow it
Does not deliver the short-term solutions – given the required time needed to build it out – that the public cloud does

Pros and Cons of Hybrid Cloud



Advantages Of Hybrid Cloud:

Best of both the public and private cloud in terms of needed resources

Added accessibility for internal and external users

Security parameters are higher than that of the public cloud

Disadvantages Of Hybrid Cloud:

Inherent inefficiency of monitoring several different security platforms

Customization of rules and policies to govern security and the elements of infrastructure that do not link between the public and private cloud

Added security risk



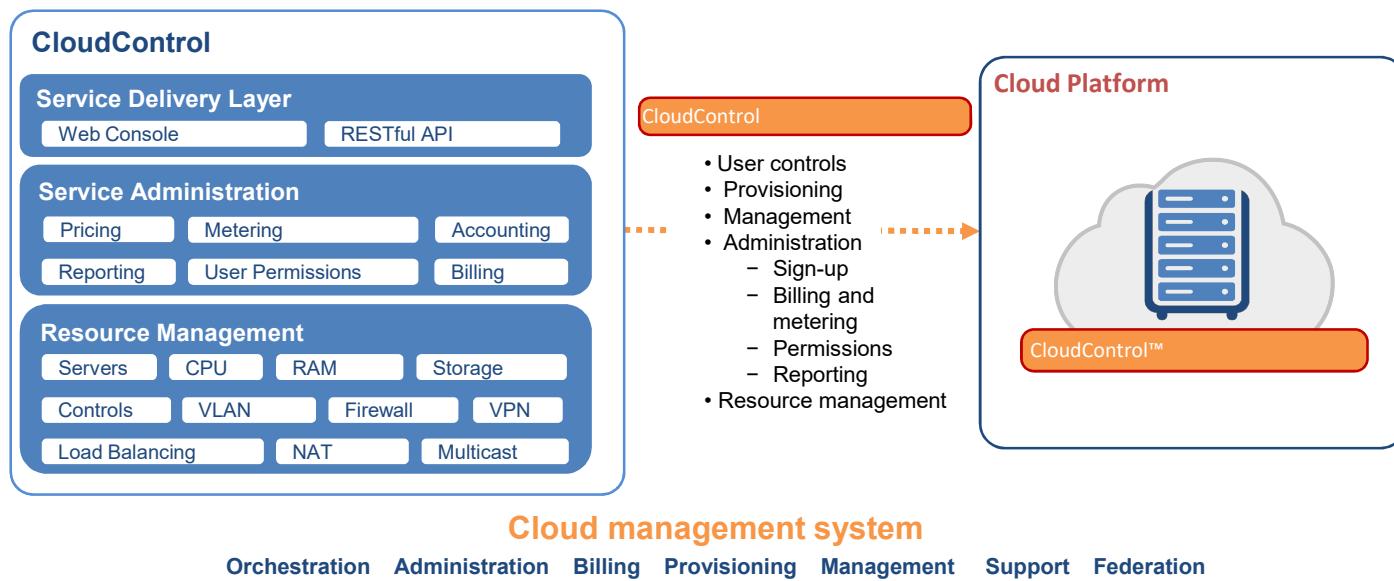
Is hybrid cloud the best option?

Businesses that are best suited to the hybrid cloud are those with sites that have unreliable fluctuations in traffic. With e-commerce, these fluctuations vary greatly between times of day and seasons in the year. Because the hybrid cloud has the same security of the private cloud, businesses can utilize both the added resources and the security to facilitate the needs of their customers. In essence, it is the best of both worlds.



Cloud management system

Addresses the complexity of cloud orchestration, provisioning and billing



Consumption Model



Model	<u>Open Community</u>	<u>Controlled Open Mode</u>	<u>Contractual Open</u>	<u>Public/Private Hybrid</u>	<u>Private Closed</u>
	Facebook Twitter LinkedIn MyFitnessPal Google Groups	IBM SmartCloud Enterprise Amazon Web Services RackSpace OpSour	Salesforce.com Workday MailChimp QuickBooks Online	IBM SmartCloud HP Cloud Service Microsoft Azure	Internal but can be implemented by a third-party vendor
Examples	No SLA	Simple SLA	SLA with no indemnification	SLA guaranteeing uptime	Explicit SLA
	No Contract	Transactional pricing	Contract	Contract	Capital expense with ongoing maintenance
Characteristics	Simple Password Protection	More security	High security provided	Highest level of security	Secure platform
	No governance model	No explicit governance	Governance in place	Explicit governance	Explicit governance



Use Cases

- Elastic/burst capacity (e.g., apps with variable load, HPC / parallel processing, etc.)
- Temporal applications (e.g., marketing apps, test & QA environments, etc.)
- Cloud-based DMZ / Perimeter Zone
- High Performance Compute
- Backup and storage
- Disaster recovery

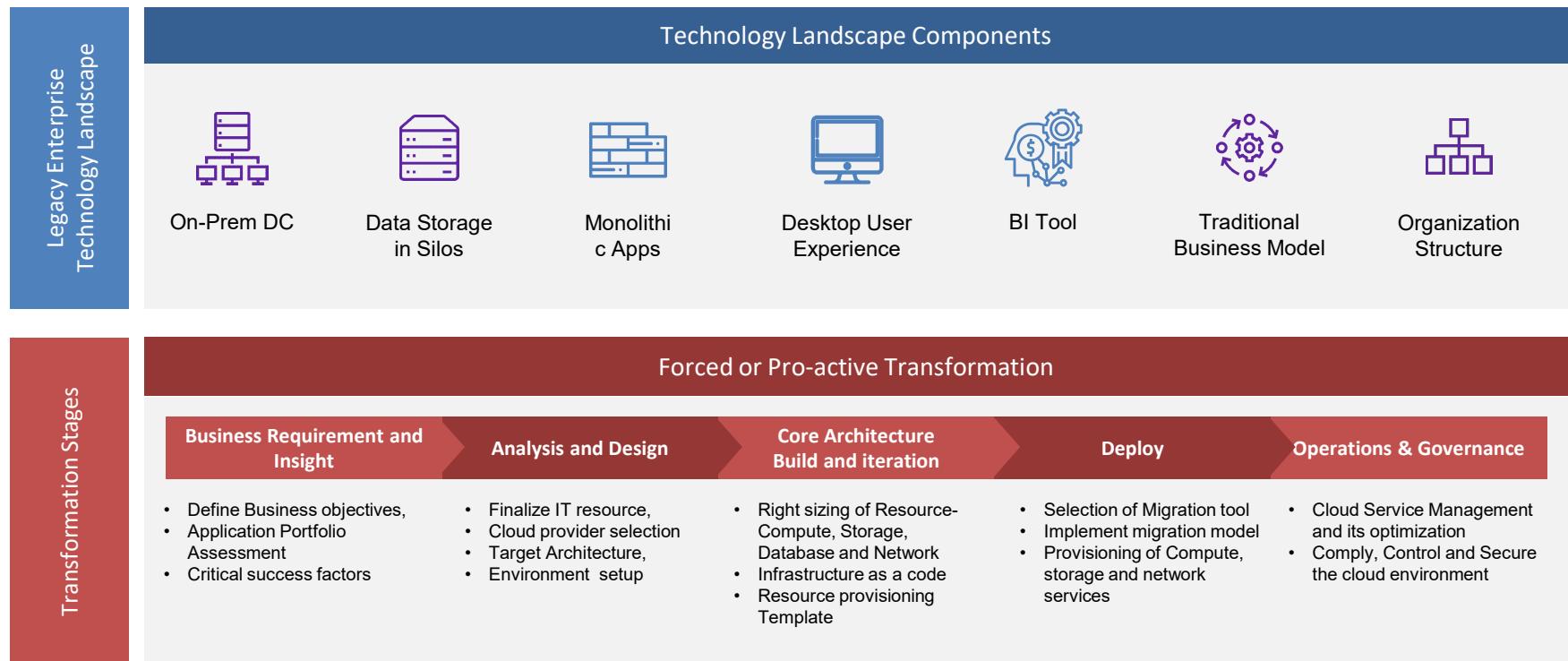
These work, but they are still deployment models



Topics of Interest as GF

Course ID	Course Description
DSE*ZG522	Big Data Systems What is big data - are existing systems sufficient?; Data Warehouse v/s Data Lakes; Hadoop – Components; Storage - Relational DBs/ NoSQL dbs / HDFS / HBase / Object Data stores - S3; Serialization; Interfaces - Hive/ Pig; Stream Processing; Spark; Mahout.
SS ZG527	Cloud Computing Concurrency and distributed computing, message passing over the network, connectivity and failure models, local vs remote connectivity, distributed resource modeling, distributed data models; replication & consistency; virtualization; CPU virtualization, memory and storage virtualization, virtualized networks, computing over WAN and Internet; computing on the cloud, computing models, service models and service contracts, programming on the cloud; Cloud infrastructure, LAN vs Wan issue, resource scaling and resource provisions, performance models, scalability, performance measurement and enhancement techniques; cloud applications and infrastructure services.
IS ZC446	Data Storage Technologies & Networks Storage Media and Technologies – Magnetic, Optical and Semiconductor media, techniques for read/write operations, issues and limitations. Usage and Access – Positioning in the memory hierarchy, Hardware and Software Design for access, Performance issues. Large Storages – Hard Disks, Networked Attached Storage, Scalability issues, Networking issues. Storage Architecture. - Storage Partitioning, Storage System Design, Caching, Legacy Systems. Storage Area Networks – Hardware and Software Components, Storage Clusters/Grids. Storage QoS – Performance, Reliability, and Security issues.
SS ZG515/DSE*ZG515	Data Warehousing Introduction, evolution of data warehousing; decision support systems; goals, benefit, and challenges of data warehousing; architecture; data warehouse information flows; software and hardware requirements; approaches to data warehouse design; creating and maintaining a data warehouse; Online Analytical Processing (OLAP) and multi-dimensional data, multi-dimensional modeling; view materialization; data marts; data warehouse metadata; data mining.
CSI** ZG522	Design and Operation of Data Centers Data Center Design: Principles (Scalability, Reliability, and Elasticity), Components - Computing Infrastructure (Processing, Storage, and Networking) and Physical Infrastructure (Power, Cooling, and Physical Security); Servers – Server Hardening, Server Optimization, Server Deployment and Consolidation, Converged and Hyper-Converged Infrastructure. Application monitoring and maintenance. Networking for data centers – device hardening, bandwidth aggregation, traffic management, redundancy, network isolation, deployment of internal security and peripheral security; Contingency Planning & Disaster Recovery: Backup, recovery, and redundancy/replication technologies and approaches. Data Center Architecture: Private, Public, and Hybrid models; Distributed Data Centers; Introduction to Software Defined DataCenters. Costing and Pricing—Costing and Cost Optimization, Pricing and Economics of Data Center Operation.
	Ethics for Data Science Introduction to data ethics, perils of big data, foundations of data privacy, challenges of privacy in the digital age, data policies, consent and fair usage.
	Infrastructure Management Introduction to System Management IT Infrastructure, Introduction to System Management IT Infrastructure, Staffing Legislation , Ethics, Outsourcing for ITSM, Customer Service, Availability, 6. Performance and Tuning, Production Acceptance, Change Management, Problem Management, Storage Management, Network Management, Configuration Management, Capacity Planning, Strategic Security, Business Continuity, Facilities Management, Developing Robust Processes, Using Technology to Automate and Evaluate Robust Processes, Integrating Systems Management Processes, Special Considerations for Client-Server and Web-Enabled Environments
DSE*ZG523	Introduction to Data Science Context and use of Data Science. Highdimensional data, graphs, vectors in high dimensional space and large matrices; Algorithms for massive data problems, sampling techniques. Techniques for extracting information/patterns from data.
CSI** ZG515	Introduction to DevOps Continual Service - continuous integration and continuous delivery; Scaling: automating infrastructure and infrastructure-as-code; DevOps and Cloud: platform-as-a-service and DevOps, use of virtual machines and containers for deployment, Micro-services; application lifecycle management: deployment pipeline and application deployment,continuous deployment pipeline; stack management - life cycle of stack and events, resource and event monitoring, auto healing; Security: security of deployment pipeline, policy-as- code.
CSI** ZG511	IT Infrastructure Projects & Processes The course introduces infrastructure software project process as a structured methodology for professional software Maintenance and infrastructure. Introduction to Software process, Introduction to ITIL, CMMI – SVC – Capacity Maturity Model integrated for Services.

Cloud Adoption Journey – Enterprise Landscape



Key Cloud Adoption Drivers

 Global Scale

Worldwide reach, Hyperscale & Elasticity with **economic benefits**

 Digital User Experience

Omni channel experience on **Mobile** platforms & enabling user **Mobility**

 Reliability & Resiliency

Reliable trading - High Availability, Dynamic Failover, COVID lessons learnt

 Compliance & Security

Regularity Compliance, **Data Protection** & **confidentiality**

 IT Simplification

Accelerated **Time to Market** - DevOps, DevSecOps, Managed PaaS

 Driving Innovation

SaaS leverage - **Microservices, Analytics, AI, Blockchain and IoT** enabling business workflows



Cloud Adoption Journey Road Map



Cloud Adoption Delivery Framework

Customers do business in a heterogeneous World so DXC Luxoft is committed to delivering the best customer experience across software, services, and support in their cloud adoption journey



Cloud Adoption Goals

- Empower enterprise mobility
- Transform the datacenter
- Enable application innovation
- Unlock insights on any data
- Support application lifecycle management



Cloud Adoption Approach

- Define how to select and prioritize applications for migrations.
- Understand proven best practices to migrate applications to cloud Platform
- Get Exposure to variety of application migration patterns – ranging from cloud Server Migration/Data Migration, to App Code* for CI/CD.
- Validate Landing Zone, Operations run book and Security playbook by actually testing applications in Cloud platform



Cloud Adoption Phases

- Planning and Discovery
- Design and Build
- Migrate
- Operation
- Governance



Cloud Adoption Tools

- Discovery tools
- Migration tools
- Monitoring tools
- DevOps Tools
- Project Management Tools



Cloud Adoption Key Success

- Cost Optimization
- Automation and Orchestration
- Improve operation efficiency and performance
- Reduce Risk
- Increase agility

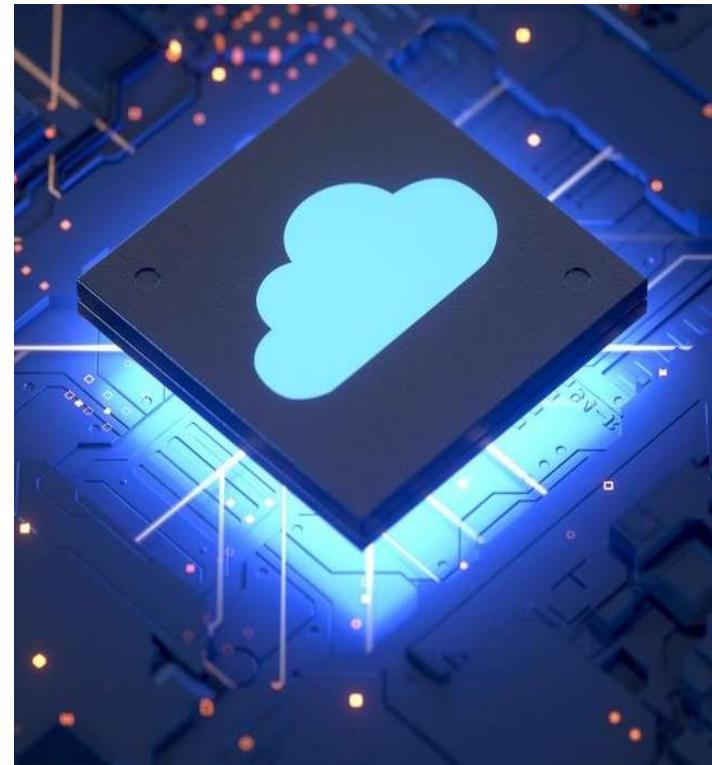
Cloud Adoption

Framework (CAF)

The Cloud Adoption Framework brings together cloud adoption best practices. It provides a set of tools, guidance, and narratives that would help shape technology, infrastructure, and people strategies for driving your desired business outcomes during your cloud adoption effort.

The cloud adoption framework objectives includes:

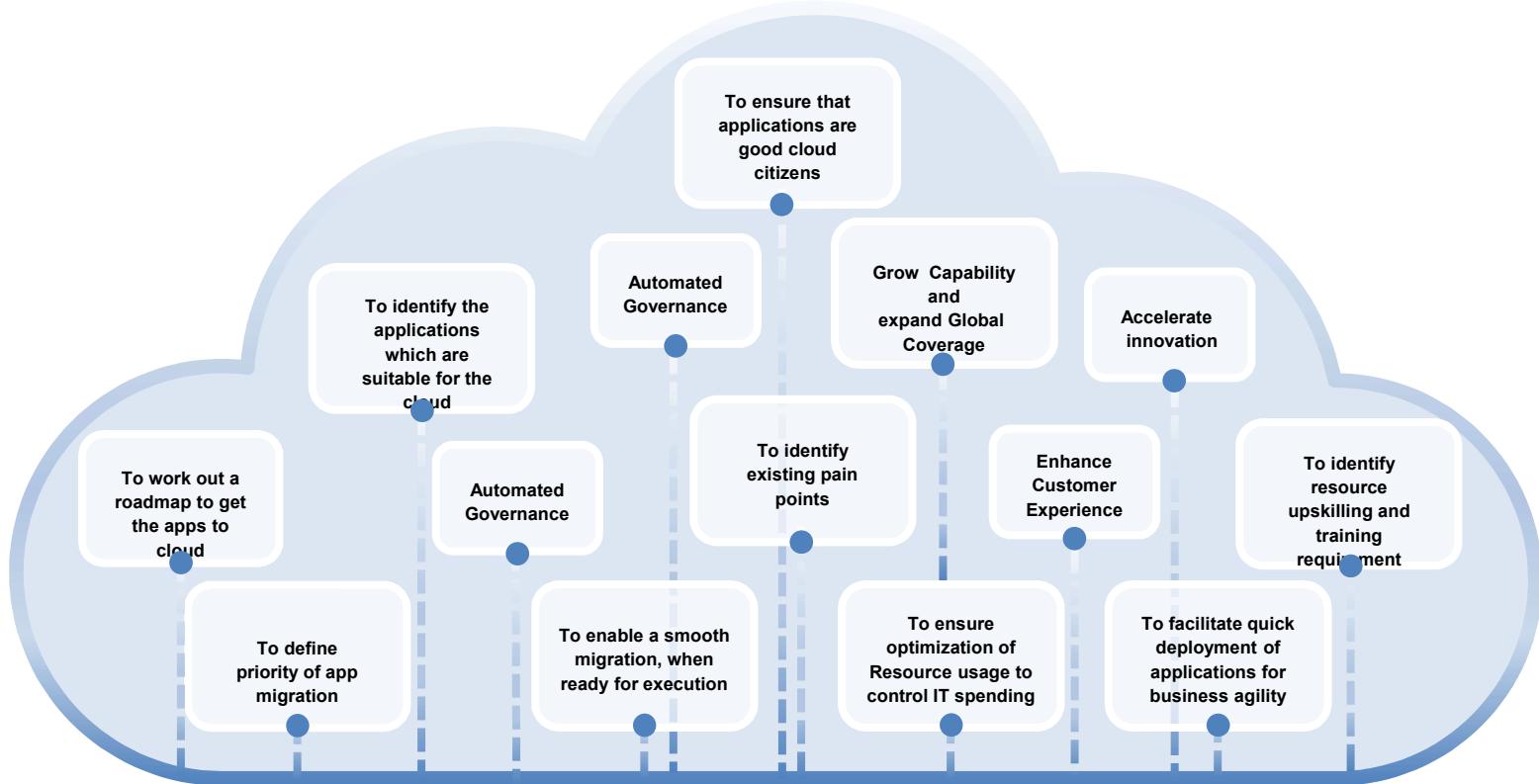
- The Cloud Adoption Framework provides tools and guidance for implementing cloud technologies to incorporate business, people and process changes,
- "Cloud Adoption framework" is used to describe collections of development tools to middleware to database services that ease the creation, deployment and management of cloud applications
- Aligns cloud adoption with business objectives across the cloud adoption stages.
- CAF Standardize technology adoption to reduce technology debt and streamlines cloud services management.
- CAF ensures security of infrastructure, applications, and data, while ensuring data sovereignty.
- CAF drives audit readiness for infrastructure applications.
- CAF allows periodical review of the reference architecture, approved list of services, security controls and cost optimization techniques.



Slide 59

- VL1** The framework should have objectives like:
 how to adopt cloud
 how to speed up adoption
 how to make applications compliant
I think we should incorporate and have a think about this. I think you have it right on slide 5 so we should decide which to use. I think I actually prefer this slide now, with some updated objectives.
The objectives we have currently are more around the benefits of a migration?
Vadgama, Vishal (DXC Luxoft), 7/15/2021
- ML1** made changes as per your suggestions
Moitra, Mridul (DXC Luxoft), 7/16/2021
- ML2** [@Vadgama, Vishal (DXC Luxoft)]
made changes as per your suggestions
Moitra, Mridul (DXC Luxoft), 7/16/2021

Why do we need a Cloud Adoption Framework





Evolution of Web

Explosive growth in applications:

biomedical informatics, space exploration, business analytics,
web 2.0 social networking: YouTube, Facebook

Extreme scale content generation: e-science and e-business data deluge

Extraordinary rate of digital content consumption: digital gluttony:

Apple iPhone, iPad, Amazon Kindle, Android, Windows Phone

Exponential growth in compute capabilities:

multi-core, storage, bandwidth, virtual machines (virtualization)

Very short cycle of obsolescence in technologies:

Windows 8, Ubuntu, Mac; Java versions; C → C#; Python

Newer architectures: web services, persistence models, distributed file systems/repositories (Google, Hadoop), multi-core, wireless and mobile

Diverse knowledge and skill levels of the workforce



In [software engineering](#), **SOA (service-oriented architecture)** is an architectural style that focus on discrete services instead of a monolithic design.^[1] By consequence, it is as well applied in the field of [software design](#) where services are provided to the other components by [application components](#), through a [communication protocol](#) over a network. A service is a discrete unit of functionality that can be accessed remotely and acted upon and updated independently, such as retrieving a credit card statement online. SOA is also intended to be independent of vendors, products and technologies.

A **web service** is any piece of software that makes itself available over the internet and uses a standardized XML messaging system



BITS Pilani
Pilani Campus

BITS Pilani presentation

Mridul Moitra
Cloud Computing



BITS Pilani
Pilani Campus



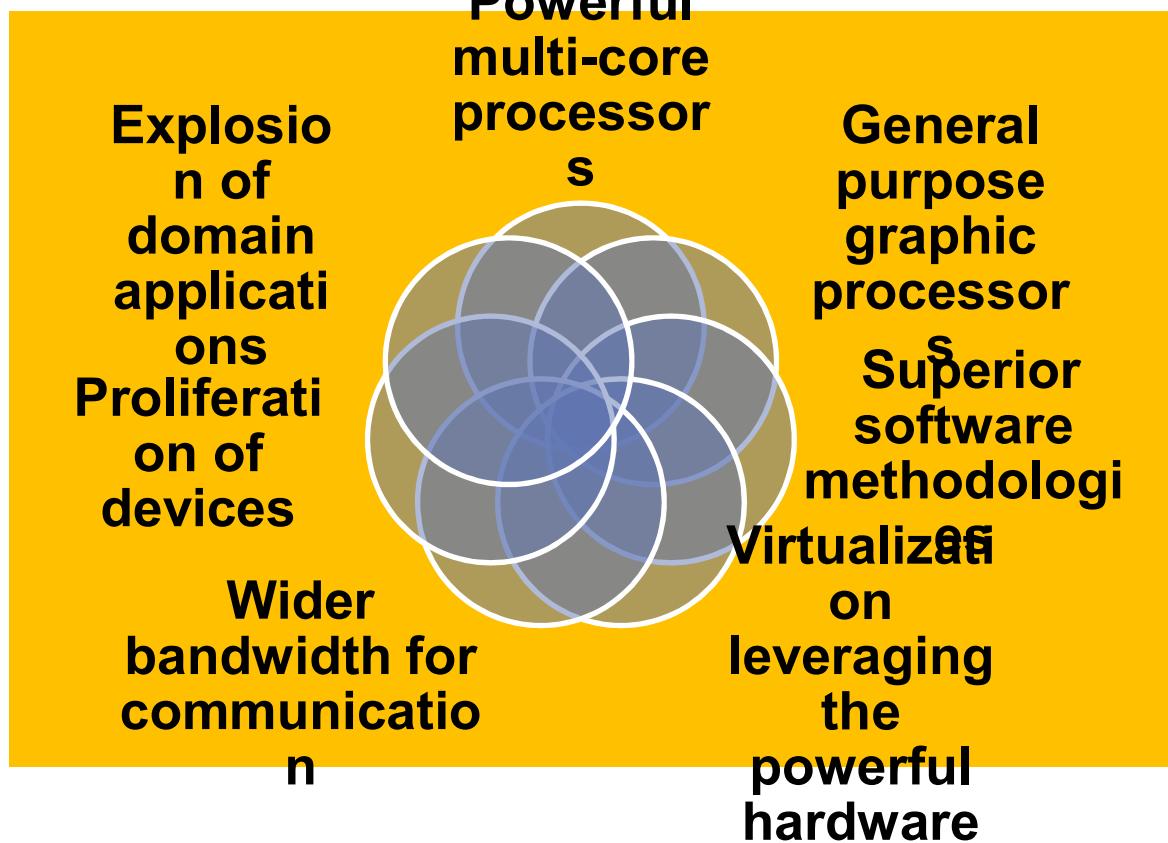
<CSI ZG527 / SS ZG527 / SE ZG527 **Cloud Computing** **Lecture No. 1**

Introduction to Cloud Computing, services and deployment models



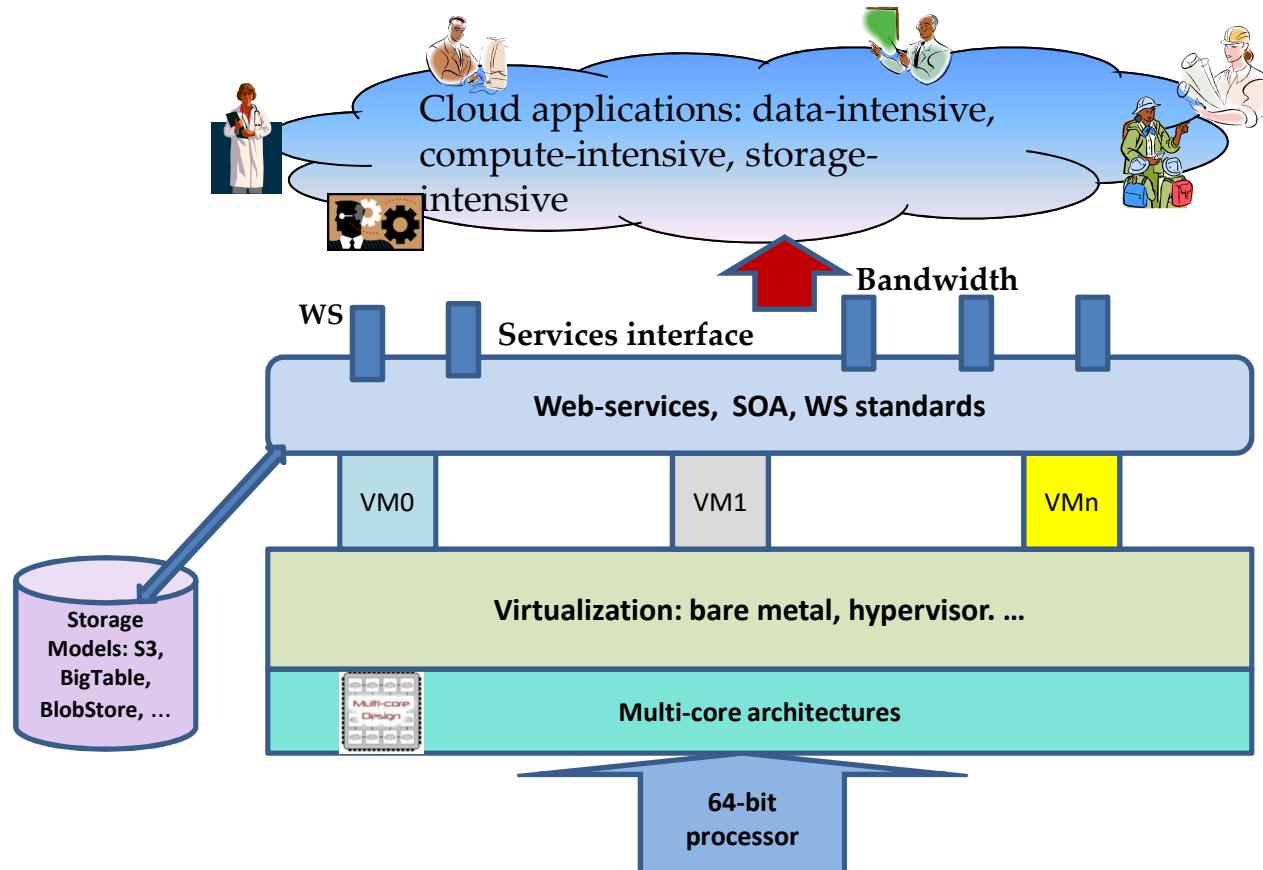
- **Agenda**
 1. **Introduction to Cloud Computing – Origins and Evolution**
 2. **Characteristics of cloud platform**
 3. **Types of Clouds and Services**
 4. **Cloud Delivery Model**
-

Motivation



- 1. Web Scale Problems**
- 2. Web 2.0 and Social Networking**
- 3. Information Explosion**
- 4. Mobile Web**

Technology Advances



Evolution of cloud computing

?

- **The evolution of cloud computing can be bifurcated into three basic phases:**
- **1. The Idea Phase-** This phase inceptioned in the early 1960s with the emergence of utility and grid computing and lasted till pre-internet bubble era. Joseph Carl Robnett Licklider was the founder of cloud computing.
- **2. The Pre-cloud Phase-** The pre-cloud phase originated in 1999 and extended to 2006. In this phase the internet as the mechanism to provide Application as Service.
- **3. The Cloud Phase-** The much talked about real cloud phase started in the year 2007 when the classification of IaaS, PaaS, and SaaS got formalized. The history of cloud computing has witnessed some very interesting breakthroughs launched by some of the leading computer/web organizations of the world.



What is Cloud Computing?

Cloud Computing is a general term used to describe a new class of network based computing that takes place over the Internet,

- basically a step on from Utility Computing
- a collection/group of integrated and networked hardware, software and Internet infrastructure (called a platform).
- Using the Internet for communication and transport provides hardware, software and networking services to clients

These platforms hide the complexity and details of the underlying infrastructure from users and applications by providing very simple graphical interface or API (Applications Programming Interface).

What is Cloud Computing cont....



the platform provides

- on-demand services, that are always on, anywhere, anytime and any place.
- Pay for use and as needed, elastic
- scale up and down in capacity and functionalities
- The hardware and software services are available to
- The hardware and software services are available to
- general public, enterprises, corporations and businesses markets

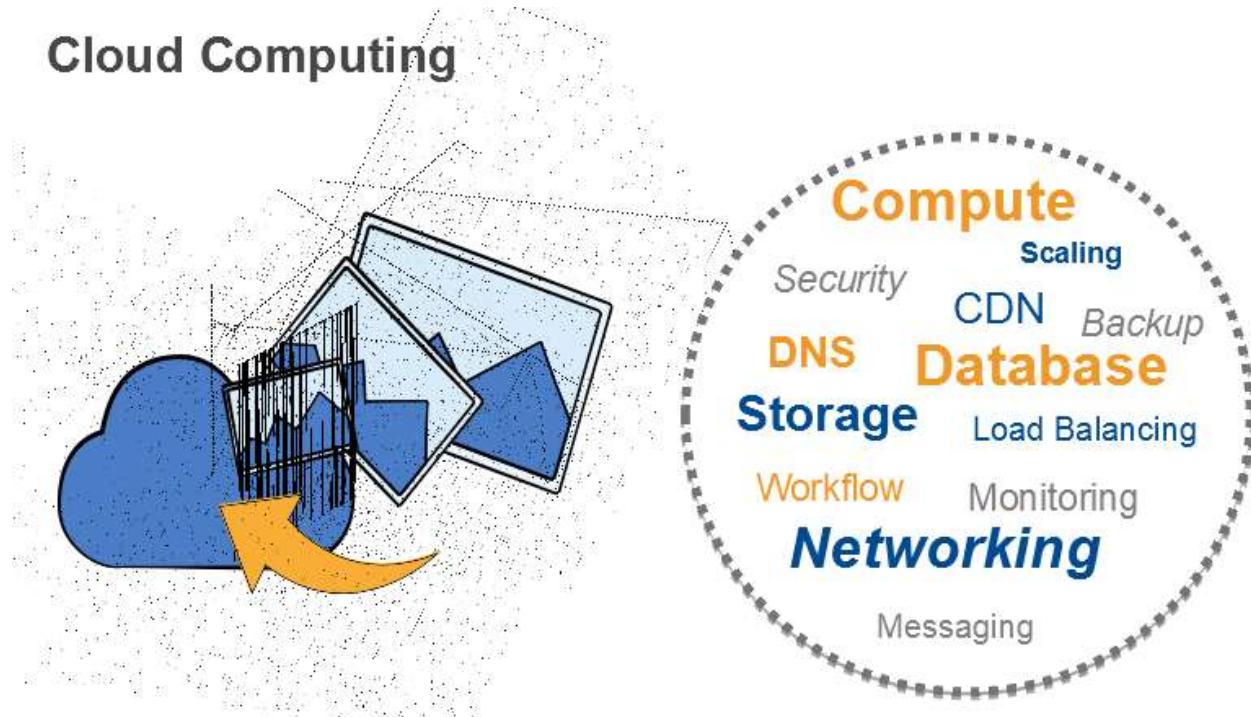


Cloud Computing: Definition

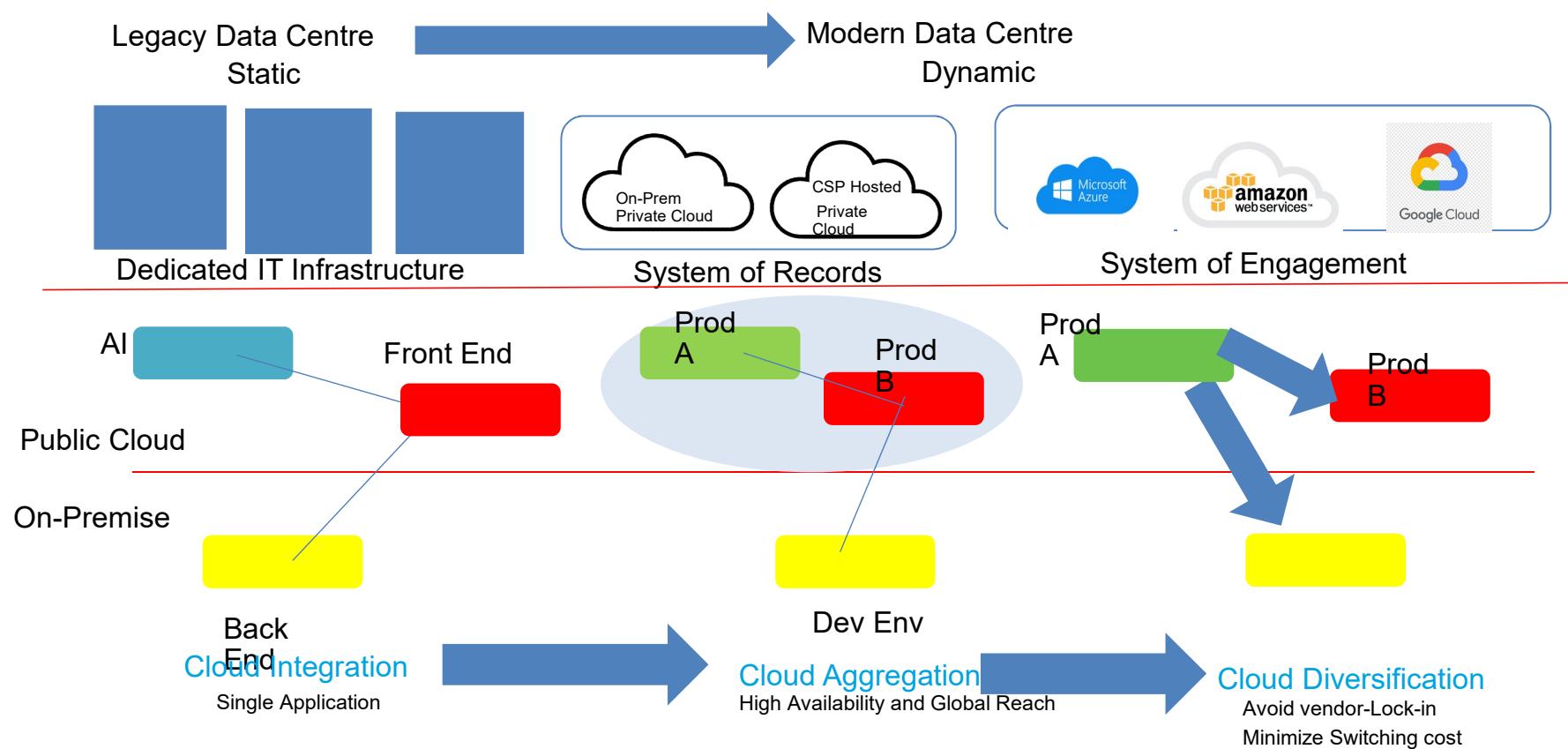
The US National Institute of Standards (NIST) defines cloud computing as follows:

Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.

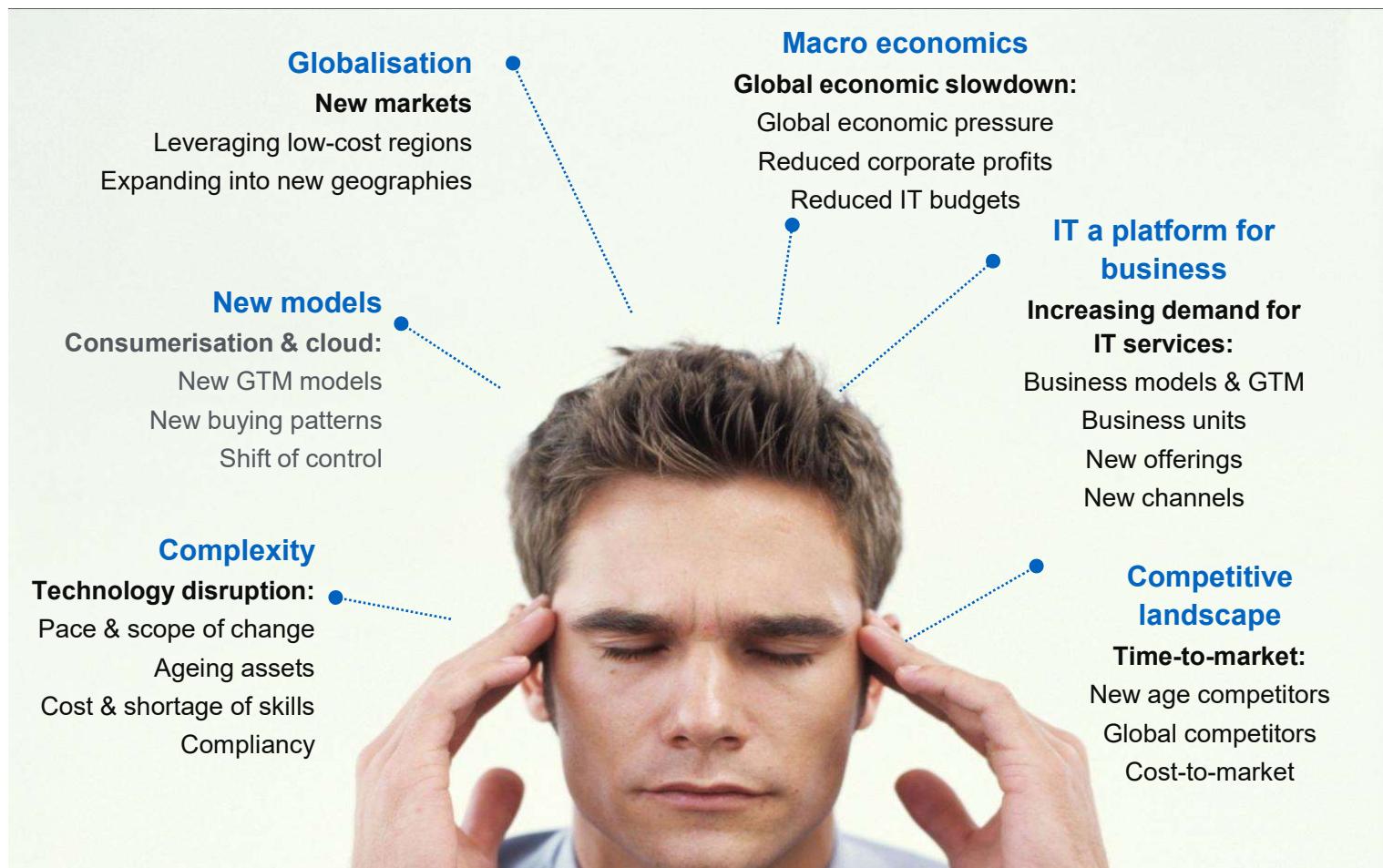
Cloud Computing



Cloud Transformation Journey



Challenges Of the CIO





Drivers for the new Platform

Generational Shift of Computing Platform

Technology	Economic	Business
	Centralized compute & storage, thin clients	Optimized for efficiency due to high cost
	PCs and servers for distributed compute, storage, etc.	Optimized for agility due to low cost
	Large DCs, commodity HW, scale-out, devices	Perpetual license for OS and application software
	Order of magnitude better efficiency and agility	Pay as you go, and only for what you use

<http://blogs.technet.com/b/yungchou/archive/2011/03/03/chou-s-theories-of-cloud-computing-the-5-3-2-principle.aspx>

Cloud Computing Business Drivers



Cost optimisation



- No capex, less assets
- Pay-as-you-use
- On-demand capacity
- Elasticity
- Economies of scale
- Time-to-value

Risk optimisation



- Business continuity
- Technology independence
- Operational complexity
- Specialised skills

Strategic agility



- Time-to-market
- Innovation
- New business models
- Resource leverage
- Adaptability
- Flexibility

...why would one not consider these benefits?

3-4-5 rule of Cloud Computing



NIST specifies 3-4-5 rule of Cloud Computing

- 3** cloud service models or service types for any cloud platform
- 4** deployment models
- 5** essential characteristics of cloud computing infrastructure

Cloud Summary



- Shared pool of configurable computing resources
- On-demand network access
- Provisioned by the Service Provider



Cloud Summary...

Cloud computing is an umbrella term used to refer to Internet based development and services

A number of characteristics define cloud data, applications services and infrastructure:

Remotely hosted: Services or data are hosted on remote infrastructure.

Ubiquitous: Services or data are available from anywhere.

Commodity model: The result is a utility computing model similar to traditional that of traditional utilities, like gas and electricity - you pay for what you would want!

Characteristics of Cloud Computing



5 Essential Characteristics of Cloud Computing

Ref: The NIST Definition of Cloud Computing

<http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf>



Source: <http://aka.ms/532>

- On demand self-service
- Broad network access
- Resource pooling
- Rapid elasticity
- Measured service

Characteristics of Cloud Computing



Essential Characteristics

- On-demand self-service
- Broad network access
- Resource pooling
- Rapid elasticity
- Measured service

Service Models

- Software as a Service
- Platform as a Service
- Infrastructure as a Service

Deployment Models

- Private
- Public
- Hybrid
- Community

Traditional Infrastructure

Amazon Web Services



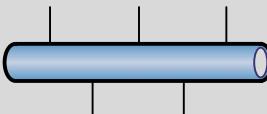
Security



Security Groups

NACLs

Access Mgmt



Networking



Servers



Amazon EC2
Instances



RDBMS

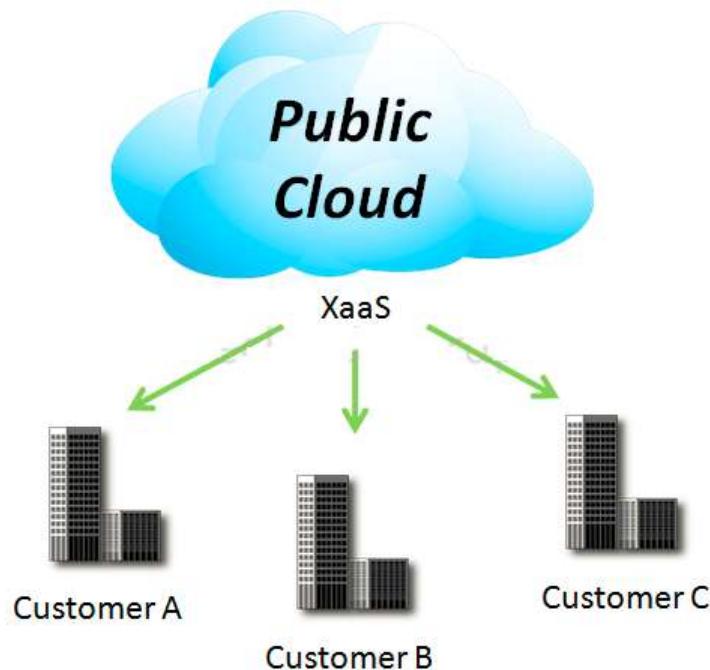
Storage &
Database



4 Deployment Models



1. Public Cloud

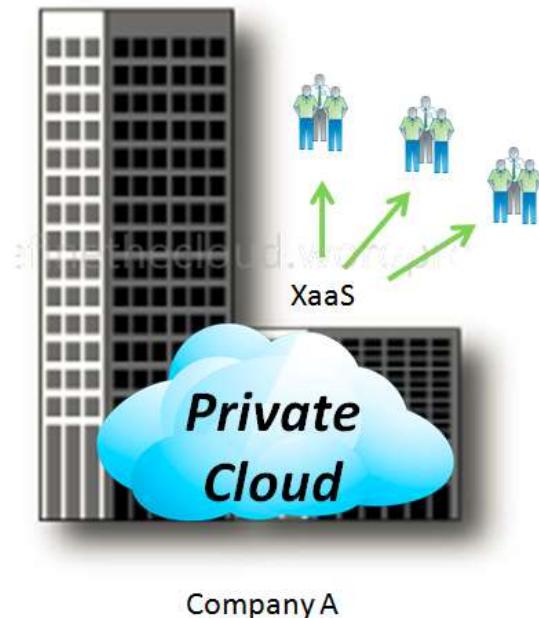


Mega-scale cloud infrastructure is made available to the general public or a large industry group and is owned by an organization selling cloud services.

4 Deployment Models



2. Private Cloud

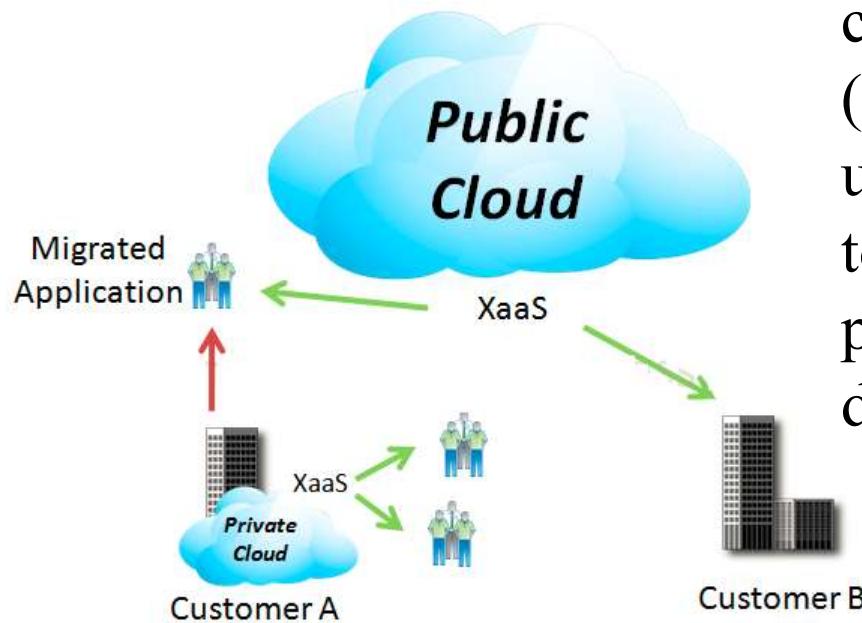


The cloud infrastructure is operated solely for an organization. It may be managed by the organization or a third party and may exist on premise or off premise.

4 Deployment Models



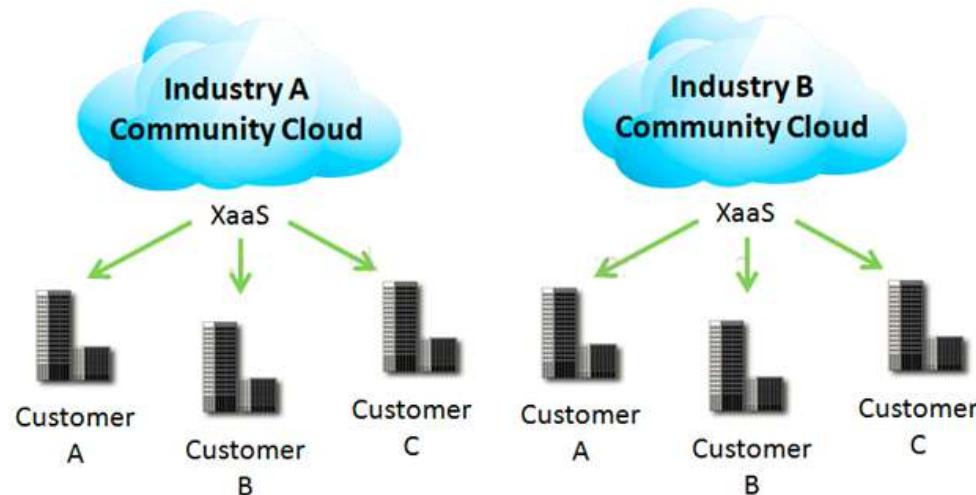
3. Hybrid Cloud



The cloud infrastructure is a composition of two or more clouds (private or public) that remain unique entities but are bound together by standardized or proprietary technology that enables data and application portability

4 Deployment Models

4. Community Cloud



Community Clouds are when an ‘infrastructure is shared by several organizations and supports a specific community that has shared concerns (e.g., mission, security requirements, policy, and compliance considerations). It may be managed by the organizations or a third party and may exist on premise or off premise’ according to NIST. A community cloud is a cloud service shared between multiple organizations with a common tie/goal/objective. E.g. OpenCirrus

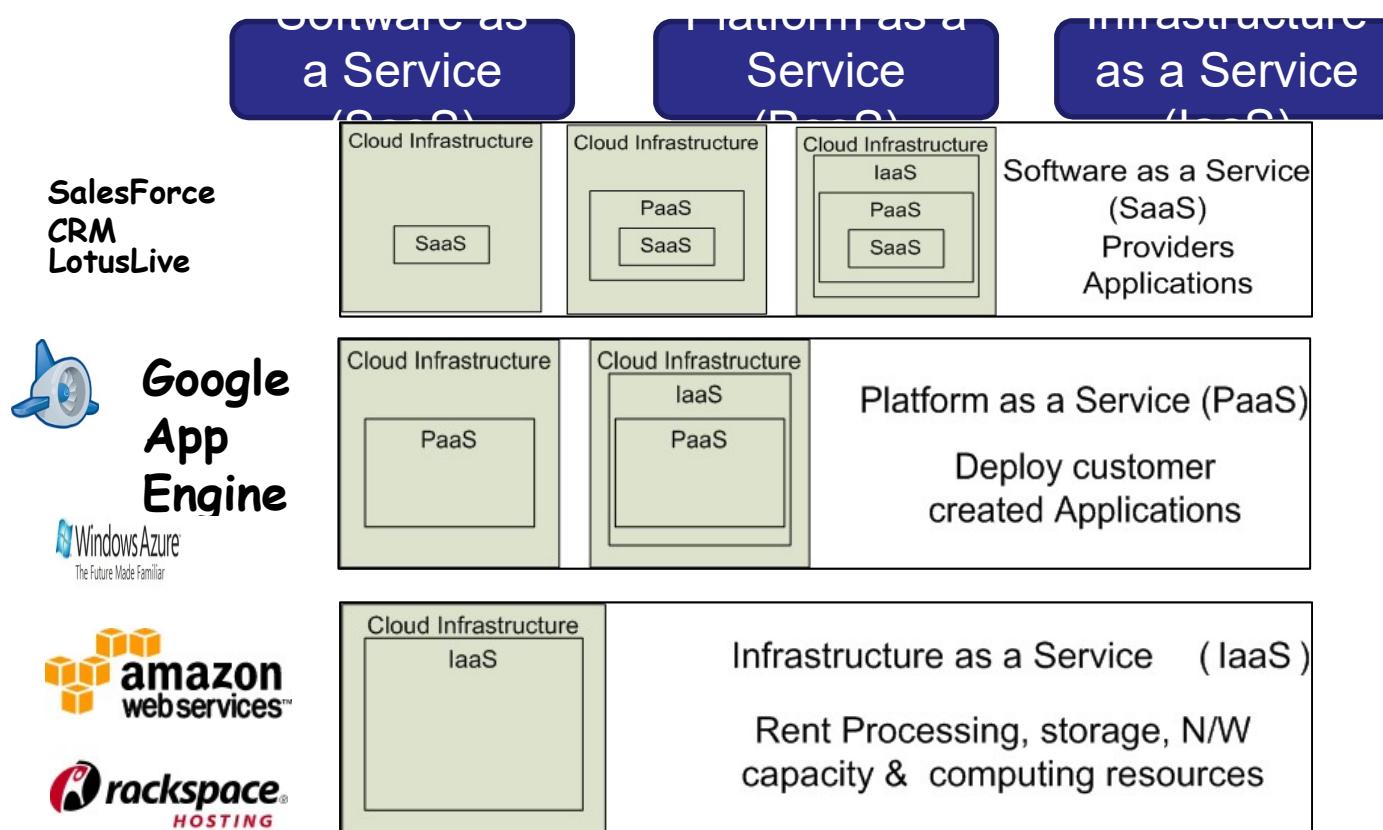
Introduction to Cloud Computing, services and deployment models



- **Agenda**
 1. **Introduction to Cloud Computing – Origins and Motivation**
 2. **3-4-5 rule of Cloud Computing**
 3. **Types of Clouds and Services**
 4. **Cloud Infrastructure and Deployment**
-



3 Cloud Service Models





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.

E.g. salesforce.com , Google Apps



Platform as a Service

Platform as a service encapsulates a layer of software and provides it as a service that can be used to build higher-level services.

2 Perspectives for PaaS :-

- 1. Producer:-** 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.
- 2. Consumer:-** 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.



Infrastructure as a Service

Infrastructure as a service delivers basic storage and computing 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.



Service Models Summary

Cloud Software as a Service (SaaS)

The **capability provided to the consumer is to use the provider's applications** running on a cloud infrastructure and accessible from various client devices through a thin client interface such as a Web browser (e.g., web-based email). The consumer does not manage or control the underlying cloud infrastructure, network, servers, operating systems, storage, or even individual application capabilities, with the possible exception of limited user-specific application configuration settings.

Cloud Platform as a Service (PaaS)

The **capability provided to the consumer is to deploy onto the cloud infrastructure consumer-created applications using programming languages and tools supported by the provider** (e.g., Java, Python, .Net). The consumer does not manage or control the underlying cloud infrastructure, network, servers, operating systems, or storage, but the consumer has control over the deployed applications and possibly application hosting environment configurations.

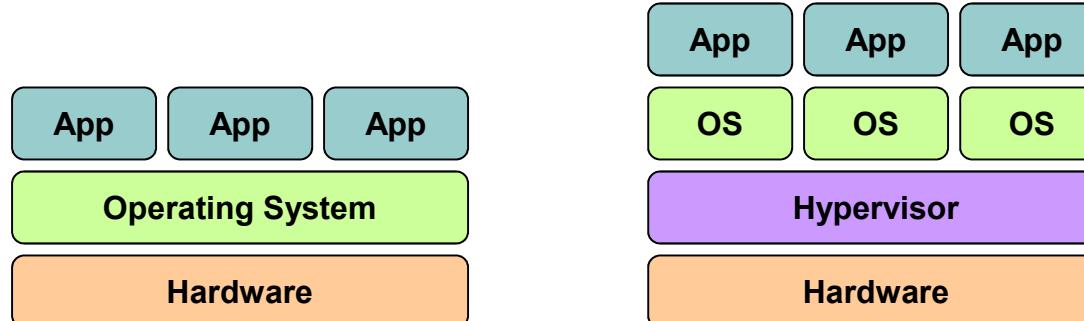
Cloud Infrastructure as a Service (IaaS)

The **capability provided to the consumer is to rent processing, storage, networks, and other fundamental computing resources** where the consumer is able to deploy and run arbitrary software, which can include operating systems and applications. The consumer does not manage or control the underlying cloud infrastructure but has control over operating systems, storage, deployed applications, and possibly select networking components (e.g., firewalls, load balancers).

Cloud Infrastructures



Key Technology is Virtualization



Virtualization plays an important role as an enabling technology for datacentre implementation by abstracting compute, network, and storage service platforms from the underlying physical hardware

Cloud Providers Characteristics



- Provide on-demand provisioning of computational resources
- Use virtualization technologies to lease these resources
- Provide public and simple remote interfaces to manage those resources
- Use a pay-as-you-go cost model, typically charging by the hour
- Operate data centers large enough to provide a seemingly unlimited amount of resources to their clients



Management of Virtualized Resources

Distributed Management of Virtual Machines

Reservation-Based Provisioning of Virtualized Resources

Provisioning to Meet SLA Commitments



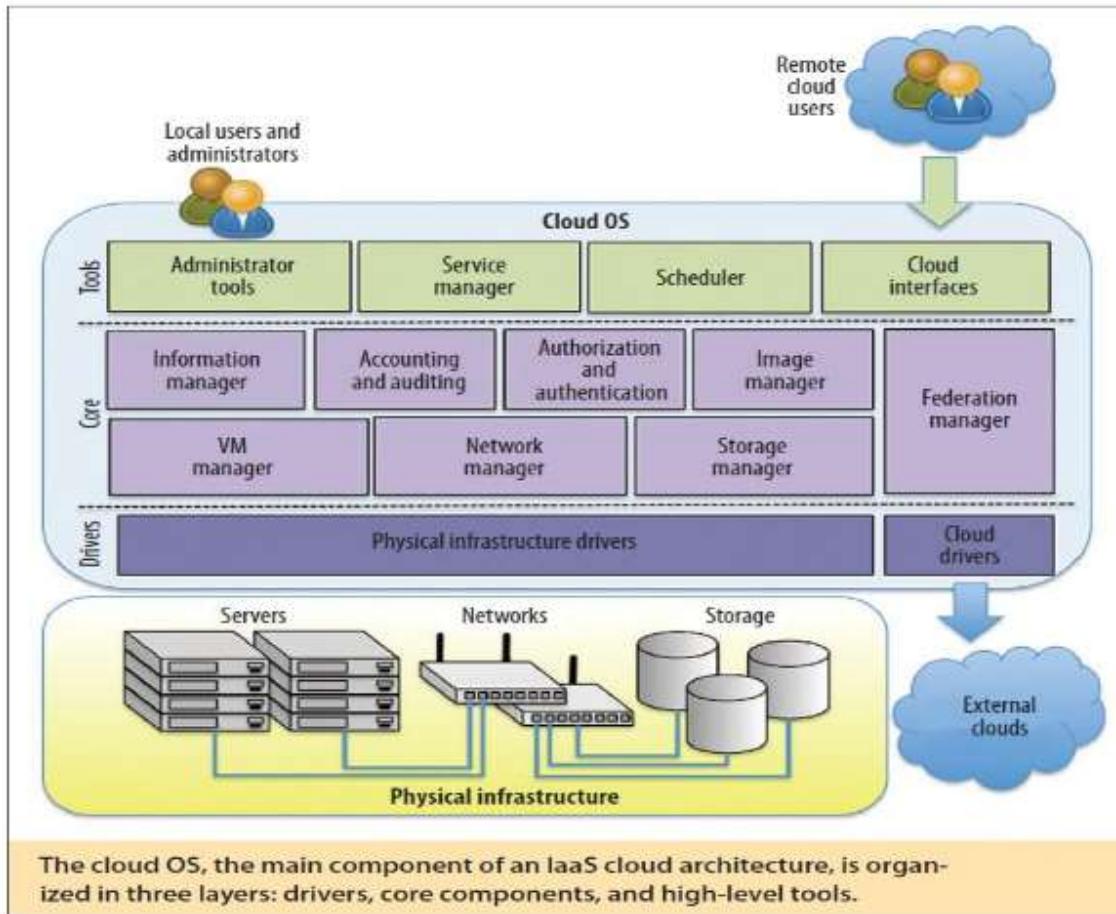
Cloud Infrastructure Anatomy

The key component of an IaaS cloud architecture is the cloud OS, which manages the physical and virtual infrastructures and controls the provisioning of virtual resources according to the needs of the user services

A cloud OS's role is to efficiently manage datacenter resources to deliver a flexible, secure, and isolated multitenant execution environment for user services that abstracts the underlying physical infrastructure and offers different interfaces and APIs for interacting with the cloud

While local users and administrators can interact with the cloud using local interfaces and administrative tools that offer rich functionality for managing, controlling, and monitoring the virtual and physical infrastructure, remote cloud users employ public cloud interfaces that usually provide more limited functionality

The Cloud OS



The cloud operating system is responsible for:

1. managing the physical and virtual infrastructure,
2. orchestrating and commanding service provisioning and deployment
3. providing federation capabilities for accessing and deploying virtual resources in remote cloud infrastructures

Pros and Cons of Cloud Computing



- Why is it becoming a Big Deal:
 - Using high-scale/low-cost providers,
 - Any time/place access via web browser,
 - Rapid scalability; incremental cost and load sharing,
 - Can forget need to focus on local IT.
- Concerns:
 - Performance, reliability, and SLAs,
 - Control of data, and service parameters,
 - Application features and choices,
 - Interaction between Cloud providers,
 - No standard API – mix of SOAP and REST!
 - Privacy, security, compliance, trust...

Public Cloud



Implementation of cloud services on resources that are shared between many customers, managed off-premises



Typically, cloud provider owns and controls the resources/assets, definition of services, costs and risks

Variations exist – such as hosters and integrated cloud platforms



Cloud solutions

Office 365 (SaaS)
Dynamics CRM Online (SaaS)
EC2, RDS
Windows Azure platform (PaaS)

Private Cloud



Implementation of cloud services on resources that are dedicated to your organization, whether they exist on-premises or off-premises



Typically, your organization owns and controls the resources/assets, definition of services, costs and risks

Variations exist – such as hosted and virtual private clouds



Cloud solutions

Windows Server 2008 R2 Hyper-V,
System Center (IaaS)

Windows Azure Appliance (PaaS)

Database Services



Private Cloud

- Private clouds are cloud infrastructures that are deployed for a single organization.
- These can be managed internally or externally, but all systems and infrastructure are for the purposes of the organization.
- When considering a private cloud, the biggest decision that a business needs to make is the scope of the needed investment to create the private cloud, as implementation can be very expensive.



Cloud vs. Public Cloud

- More than a location and ownership distinction

- ▶ Private Cloud
 - ▶ Control
 - ▶ Conventional storage
 - ▶ Custom policies
 - ▶ Heterogeneous infrastructure
 - ▶ Regulatory compliance & data sovereignty
- ▶ Public Cloud
 - ▶ Scale
 - ▶ Cloud storage
 - ▶ Common policies
 - ▶ Homogeneous infrastructure
 - ▶ Work in progress



Mixed/blended model of private and public clouds

- Variations and multiple interpretations exist

On-premises and off-premises bridging

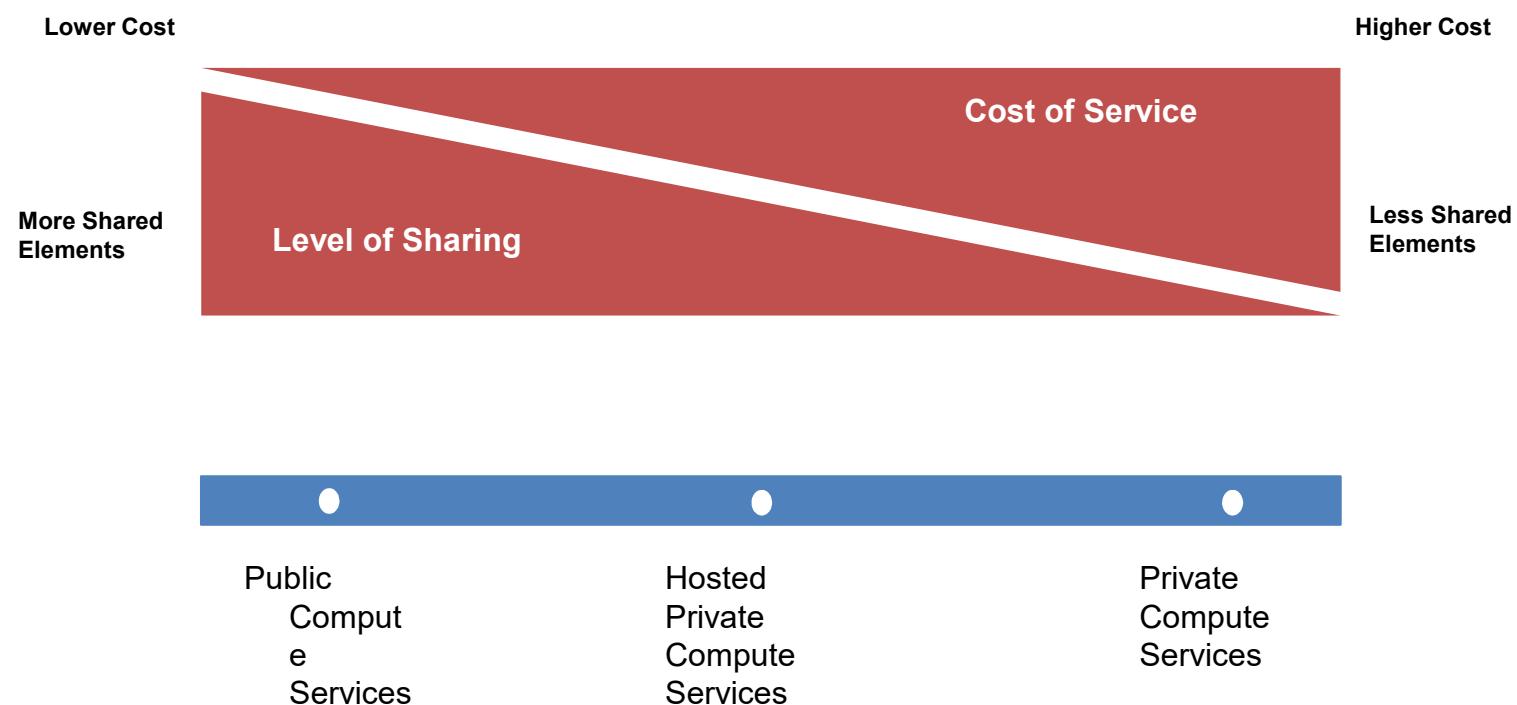
- Most common scenario today
- Especially for large enterprises

More than a deployment / delivery model

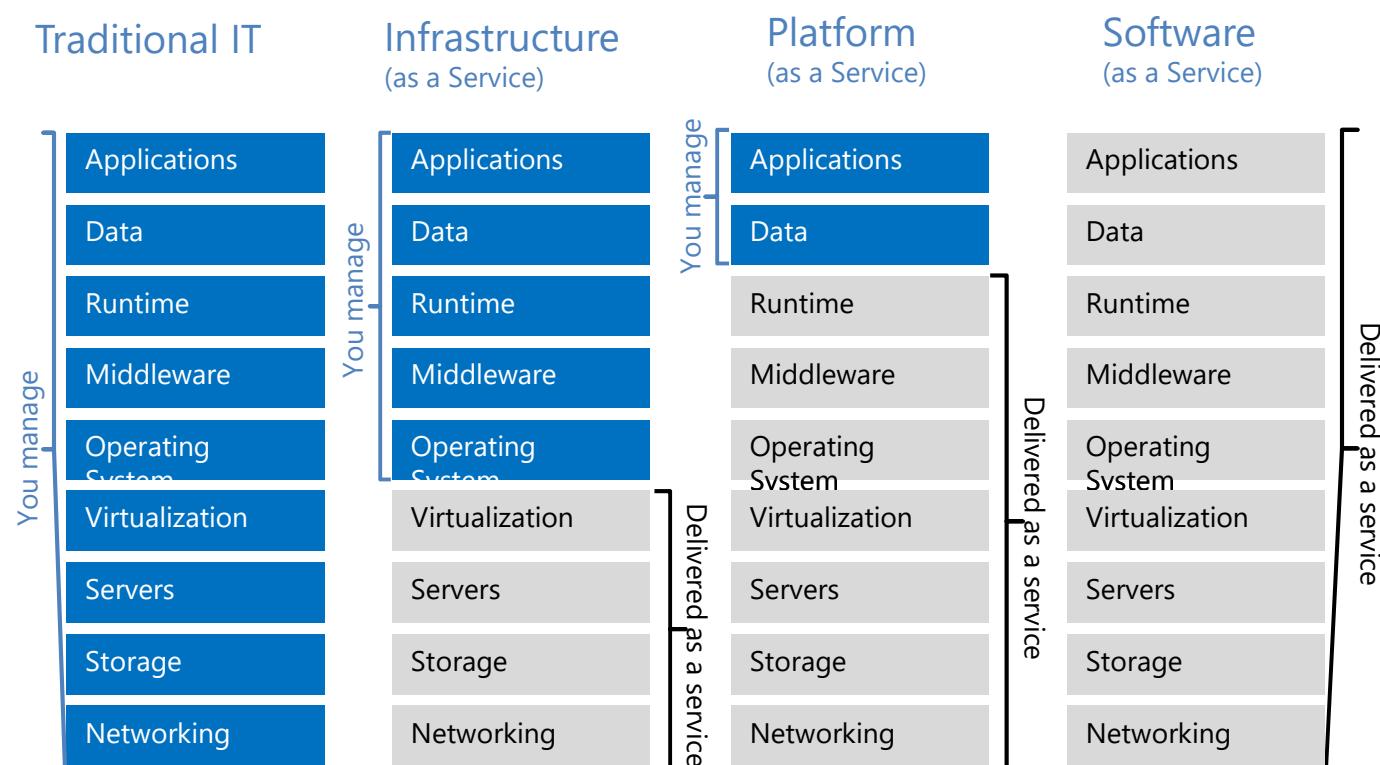
- Application design, architectural model



Product Families and Cost Principles



Cloud Service Models





Cloud Provider List

Cloud providers

The slide displays a grid of logos for various cloud providers. The companies listed include:

- Microsoft
- iCloud
- salesforce
- IBM Cloud
- Google
- Joyent
- RedMonk
- bluelock
- Amazon Web Services
- HP Cloud Services
- CITRIX
- greenqloud
- rackspace
- at&t
- SAP
- Cloud Solutions

Partial list of cloud companies



Is public cloud the best option?

Businesses that are best suited for using the public cloud are ones that need to bring a product/service to market quickly without the internal infrastructure and support to build out their own private cloud system. It does provide smaller companies without IT departments the opportunity to match the deployment speed of larger companies.

Pros and Cons of Public Cloud



Increased network efficiency and more resources

Reduced complexity and lead times (because the architecture is fixed)

Ready-to-go applications used within the public cloud can conform to the demands of business

Disadvantages Of Public Cloud:

Fewer options for customization

Substantially less secure

Fixed architecture cannot (at times) grow with the needs of the business

Is the private cloud the best option?



- Businesses that are best suited for the private cloud are ones that must comply with
- regulatory guidelines or have highly volatile applications needed within the cloud.
- Additionally, these businesses will be required to install their own servers and storage
- hardware that allows for modifications in workloads. Though this can be a significant
- investment, it is required if the business deals with regulated data or must comply with
- industry rules.

Pros and Cons of private Cloud



Advantages Of Private Cloud:

Extensive security options and capabilities, substantially more than the public cloud availability to the internal network and increased access/communication for internal users can grow with a business and be expanded or changed as needed

Disadvantages Of Private Cloud:

Significant level of engagement from both management and IT departments are required

Significant investment is required, both to build the private cloud and maintain/grow it
Does not deliver the short-term solutions – given the required time needed to build it out – that the public cloud does

Pros and Cons of Hybrid Cloud



Advantages Of Hybrid Cloud:

Best of both the public and private cloud in terms of needed resources

Added accessibility for internal and external users

Security parameters are higher than that of the public cloud

Disadvantages Of Hybrid Cloud:

Inherent inefficiency of monitoring several different security platforms

Customization of rules and policies to govern security and the elements of infrastructure that do not link between the public and private cloud

Added security risk



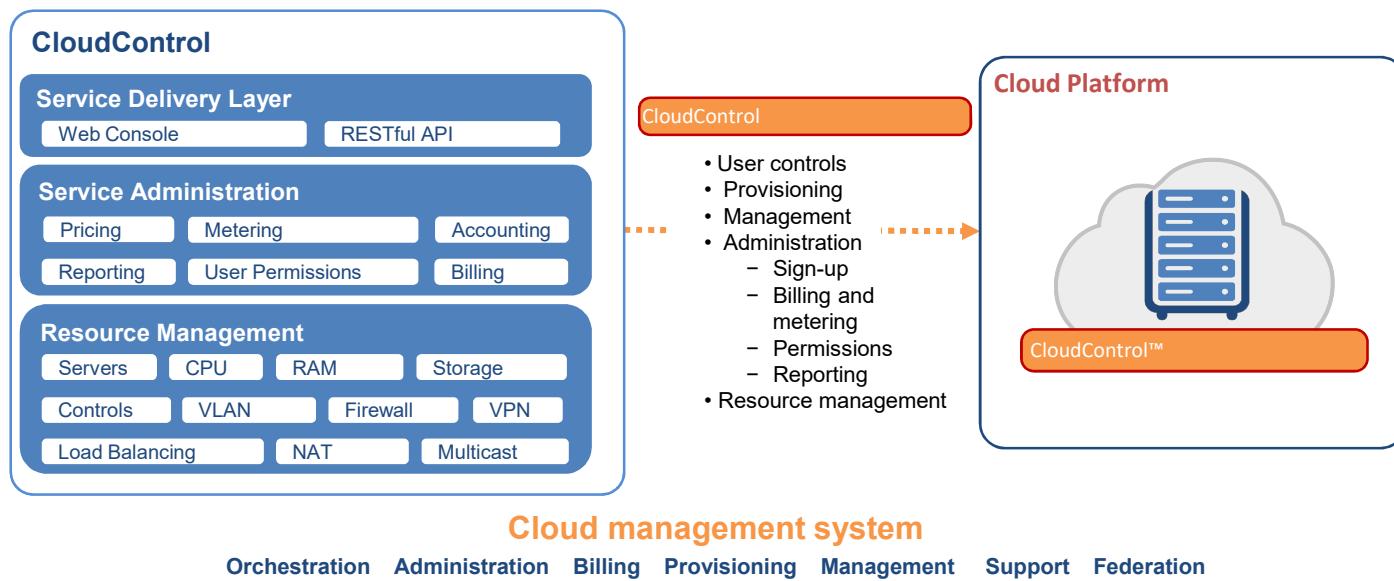
Is hybrid cloud the best option?

Businesses that are best suited to the hybrid cloud are those with sites that have unreliable fluctuations in traffic. With e-commerce, these fluctuations vary greatly between times of day and seasons in the year. Because the hybrid cloud has the same security of the private cloud, businesses can utilize both the added resources and the security to facilitate the needs of their customers. In essence, it is the best of both worlds.



Cloud management system

Addresses the complexity of cloud orchestration, provisioning and billing



Consumption Model



Model	<u>Open Community</u>	<u>Controlled Open Mode</u>	<u>Contractual Open</u>	<u>Public/Private Hybrid</u>	<u>Private Closed</u>
Examples	Facebook Twitter LinkedIn MyFitnessPal Google Groups	IBM SmartCloud Enterprise Amazon Web Services RackSpace OpSour	Salesforce.com Workday MailChimp QuickBooks Online	IBM SmartCloud HP Cloud Service Microsoft Azure	Internal but can be implemented by a third-party vendor
Characteristics	No SLA	Simple SLA	SLA with no indemnification	SLA guaranteeing uptime	Explicit SLA
	No Contract	Transactional pricing	Contract	Contract	Capital expense with ongoing maintenance
	Simple Password Protection	More security	High security provided	Highest level of security	Secure platform
	No governance model	No explicit governance	Governance in place	Explicit governance	Explicit governance



Use Cases

- Elastic/burst capacity (e.g., apps with variable load, HPC / parallel processing, etc.)
- Temporal applications (e.g., marketing apps, test & QA environments, etc.)
- Cloud-based DMZ / Perimeter Zone
- High Performance Compute
- Backup and storage
- Disaster recovery

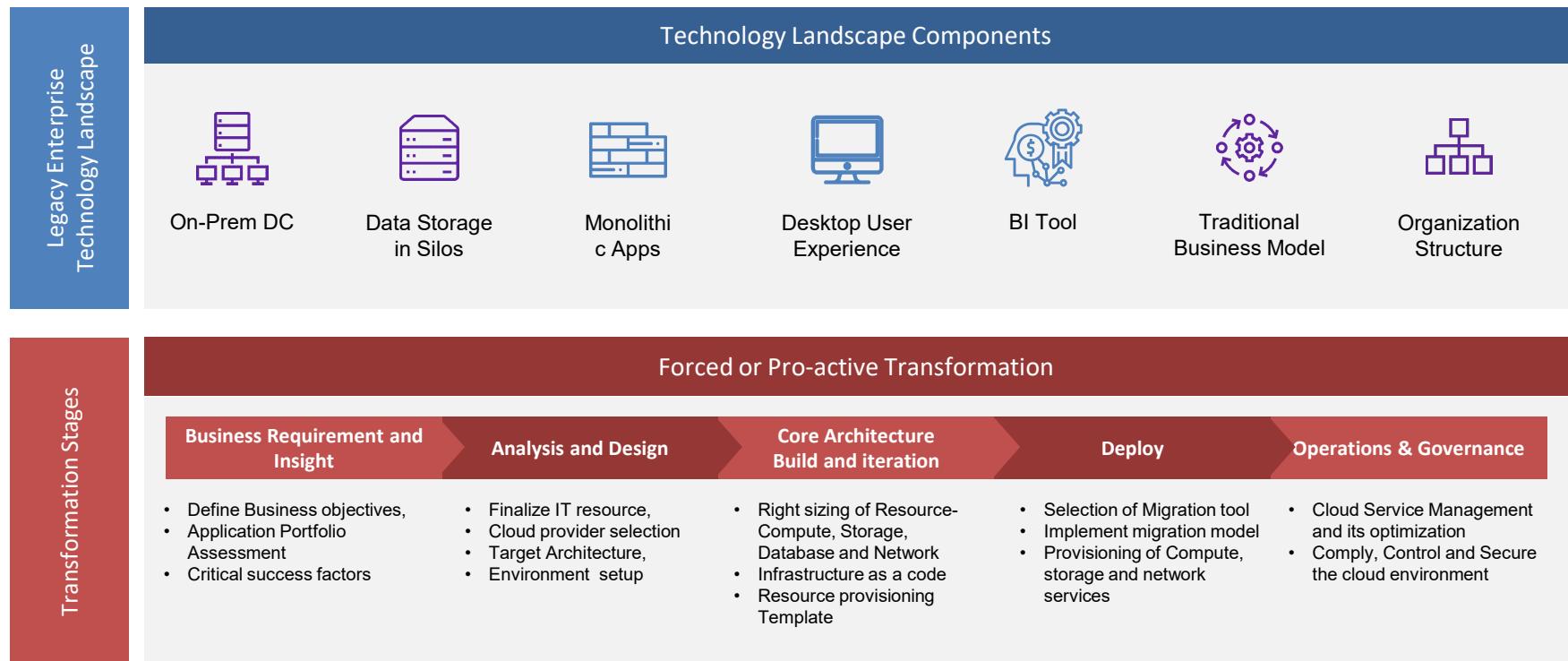
These work, but they are still deployment models



Topics of Interest as GF

Course ID	Course Description
DSE*ZG522	Big Data Systems What is big data - are existing systems sufficient?; Data Warehouse v/s Data Lakes; Hadoop – Components; Storage - Relational DBs/ NoSQL dbs / HDFS / HBase / Object Data stores - S3; Serialization; Interfaces - Hive/ Pig; Stream Processing; Spark; Mahout.
SS ZG527	Cloud Computing Concurrency and distributed computing, message passing over the network, connectivity and failure models, local vs remote connectivity, distributed resource modeling, distributed data models; replication & consistency; virtualization; CPU virtualization, memory and storage virtualization, virtualized networks, computing over WAN and Internet; computing on the cloud, computing models, service models and service contracts, programming on the cloud; Cloud infrastructure, LAN vs Wan issue, resource scaling and resource provisions, performance models, scalability, performance measurement and enhancement techniques; cloud applications and infrastructure services.
IS ZC446	Data Storage Technologies & Networks Storage Media and Technologies – Magnetic, Optical and Semiconductor media, techniques for read/write operations, issues and limitations. Usage and Access – Positioning in the memory hierarchy, Hardware and Software Design for access, Performance issues. Large Storages – Hard Disks, Networked Attached Storage, Scalability issues, Networking issues. Storage Architecture. - Storage Partitioning, Storage System Design, Caching, Legacy Systems. Storage Area Networks – Hardware and Software Components, Storage Clusters/Grids. Storage QoS – Performance, Reliability, and Security issues.
SS ZG515/DSE*ZG515	Data Warehousing Introduction, evolution of data warehousing; decision support systems; goals, benefit, and challenges of data warehousing; architecture; data warehouse information flows; software and hardware requirements; approaches to data warehouse design; creating and maintaining a data warehouse; Online Analytical Processing (OLAP) and multi-dimensional data, multi-dimensional modeling; view materialization; data marts; data warehouse metadata; data mining.
CSI** ZG522	Design and Operation of Data Centers Data Center Design: Principles (Scalability, Reliability, and Elasticity), Components - Computing Infrastructure (Processing, Storage, and Networking) and Physical Infrastructure (Power, Cooling, and Physical Security); Servers – Server Hardening, Server Optimization, Server Deployment and Consolidation, Converged and Hyper-Converged Infrastructure. Application monitoring and maintenance. Networking for data centers – device hardening, bandwidth aggregation, traffic management, redundancy, network isolation, deployment of internal security and peripheral security; Contingency Planning & Disaster Recovery: Backup, recovery, and redundancy/replication technologies and approaches. Data Center Architecture: Private, Public, and Hybrid models; Distributed Data Centers; Introduction to Software Defined DataCenters. Costing and Pricing—Costing and Cost Optimization, Pricing and Economics of Data Center Operation.
	Ethics for Data Science Introduction to data ethics, perils of big data, foundations of data privacy, challenges of privacy in the digital age, data policies, consent and fair usage.
	Infrastructure Management Introduction to System Management IT Infrastructure, Introduction to System Management IT Infrastructure, Staffing Legislation , Ethics, Outsourcing for ITSM, Customer Service, Availability, 6. Performance and Tuning, Production Acceptance, Change Management, Problem Management, Storage Management, Network Management, Configuration Management, Capacity Planning, Strategic Security, Business Continuity, Facilities Management, Developing Robust Processes, Using Technology to Automate and Evaluate Robust Processes, Integrating Systems Management Processes, Special Considerations for Client-Server and Web-Enabled Environments
DSE*ZG523	Introduction to Data Science Context and use of Data Science. Highdimensional data, graphs, vectors in high dimensional space and large matrices; Algorithms for massive data problems, sampling techniques. Techniques for extracting information/patterns from data.
CSI** ZG515	Introduction to DevOps Continual Service - continuous integration and continuous delivery; Scaling: automating infrastructure and infrastructure-as-code; DevOps and Cloud: platform-as-a-service and DevOps, use of virtual machines and containers for deployment, Micro-services; application lifecycle management: deployment pipeline and application deployment,continuous deployment pipeline; stack management - life cycle of stack and events, resource and event monitoring, auto healing; Security: security of deployment pipeline, policy-as- code.
CSI** ZG511	IT Infrastructure Projects & Processes The course introduces infrastructure software project process as a structured methodology for professional software Maintenance and infrastructure. Introduction to Software process, Introduction to ITIL, CMMI – SVC – Capacity Maturity Model integrated for Services.

Cloud Adoption Journey – Enterprise Landscape



Key Cloud Adoption Drivers

 Global Scale

Worldwide reach, Hyperscale & Elasticity with **economic benefits**

 Digital User Experience

Omni channel experience on **Mobile** platforms & enabling user **Mobility**

 Reliability & Resiliency

Reliable trading - High Availability, Dynamic Failover, COVID lessons learnt

 Compliance & Security

Regularity Compliance, **Data Protection** & **confidentiality**

 IT Simplification

Accelerated Time to Market - DevOps, DevSecOps, Managed PaaS

 Driving Innovation

SaaS leverage - Microservices, Analytics, AI, Blockchain and IoT enabling business workflows



Cloud Adoption Journey Road Map



Cloud Adoption Delivery Framework

Customers do business in a heterogeneous World so DXC Luxoft is committed to delivering the best customer experience across software, services, and support in their cloud adoption journey



Cloud Adoption Goals

- Empower enterprise mobility
- Transform the datacenter
- Enable application innovation
- Unlock insights on any data
- Support application lifecycle management



Cloud Adoption Approach

- Define how to select and prioritize applications for migrations.
- Understand proven best practices to migrate applications to cloud Platform
- Get Exposure to variety of application migration patterns – ranging from cloud Server Migration/Data Migration, to App Code* for CI/CD.
- Validate Landing Zone, Operations run book and Security playbook by actually testing applications in Cloud platform



Cloud Adoption Phases

- Planning and Discovery
- Design and Build
- Migrate
- Operation
- Governance



Cloud Adoption Tools

- Discovery tools
- Migration tools
- Monitoring tools
- DevOps Tools
- Project Management Tools



Cloud Adoption Key Success

- Cost Optimization
- Automation and Orchestration
- Improve operation efficiency and performance
- Reduce Risk
- Increase agility

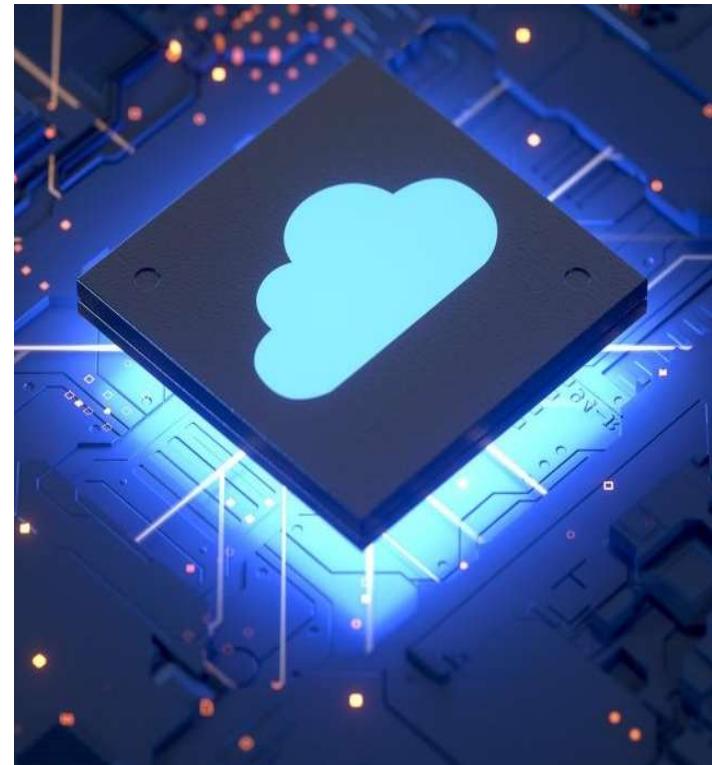
Cloud Adoption

Framework (CAF)

The Cloud Adoption Framework brings together cloud adoption best practices. It provides a set of tools, guidance, and narratives that would help shape technology, infrastructure, and people strategies for driving your desired business outcomes during your cloud adoption effort.

The cloud adoption framework objectives includes:

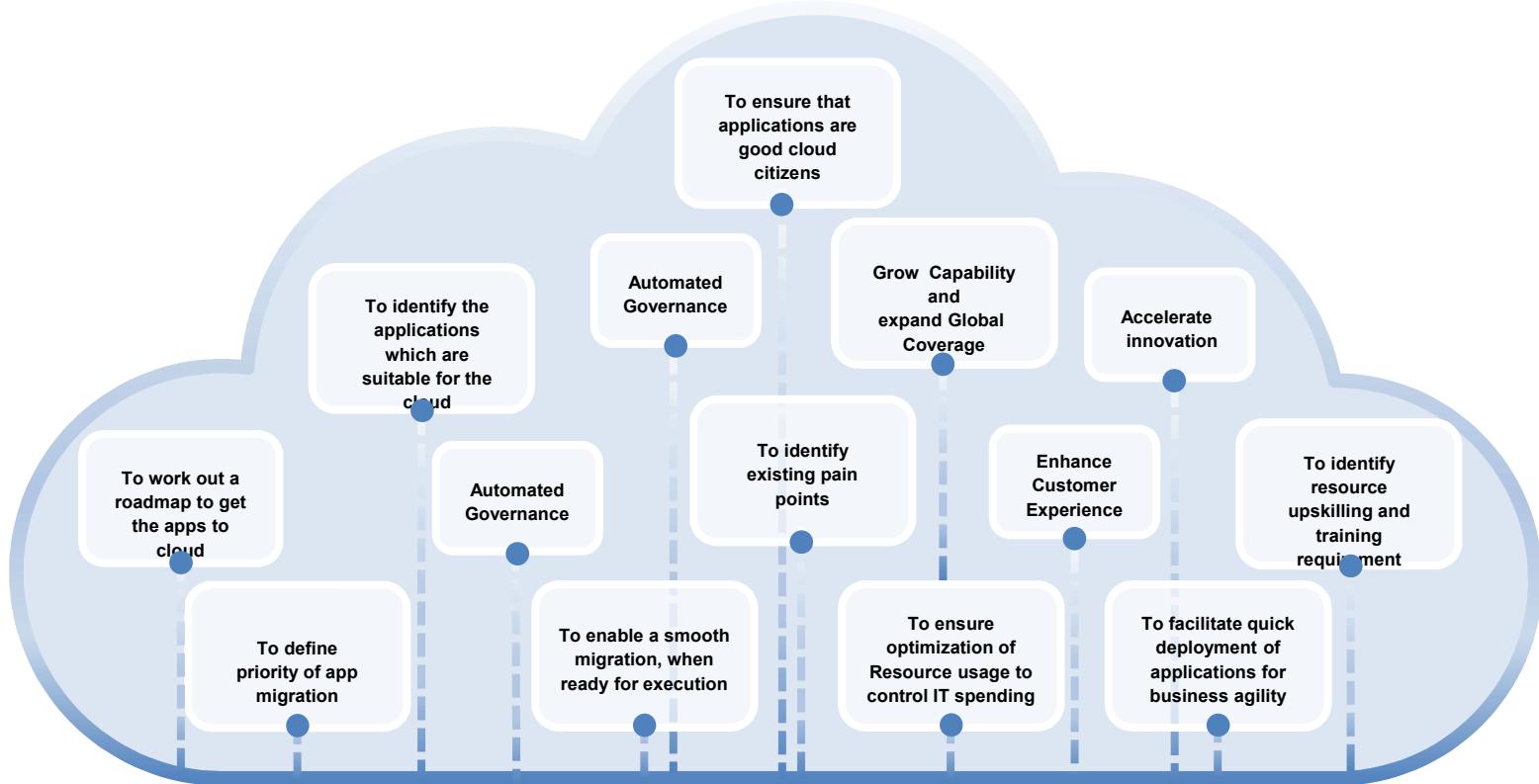
- The Cloud Adoption Framework provides tools and guidance for implementing cloud technologies to incorporate business, people and process changes,
- "Cloud Adoption framework" is used to describe collections of development tools to middleware to database services that ease the creation, deployment and management of cloud applications
- Aligns cloud adoption with business objectives across the cloud adoption stages.
- CAF Standardize technology adoption to reduce technology debt and streamlines cloud services management.
- CAF ensures security of infrastructure, applications, and data, while ensuring data sovereignty.
- CAF drives audit readiness for infrastructure applications.
- CAF allows periodical review of the reference architecture, approved list of services, security controls and cost optimization techniques.



Slide 59

- VL1** The framework should have objectives like:
 how to adopt cloud
 how to speed up adoption
 how to make applications compliant
I think we should incorporate and have a think about this. I think you have it right on slide 5 so we should decide which to use. I think I actually prefer this slide now, with some updated objectives.
The objectives we have currently are more around the benefits of a migration?
Vadgama, Vishal (DXC Luxoft), 7/15/2021
- ML1** made changes as per your suggestions
Moitra, Mridul (DXC Luxoft), 7/16/2021
- ML2** [@Vadgama, Vishal (DXC Luxoft)]
made changes as per your suggestions
Moitra, Mridul (DXC Luxoft), 7/16/2021

Why do we need a Cloud Adoption Framework





Evolution of Web

Explosive growth in applications:

biomedical informatics, space exploration, business analytics,
web 2.0 social networking: YouTube, Facebook

Extreme scale content generation: e-science and e-business data deluge

Extraordinary rate of digital content consumption: digital gluttony:

Apple iPhone, iPad, Amazon Kindle, Android, Windows Phone

Exponential growth in compute capabilities:

multi-core, storage, bandwidth, virtual machines (virtualization)

Very short cycle of obsolescence in technologies:

Windows 8, Ubuntu, Mac; Java versions; C → C#; Python

Newer architectures: web services, persistence models, distributed file systems/repositories (Google, Hadoop), multi-core, wireless and mobile

Diverse knowledge and skill levels of the workforce



In [software engineering](#), **SOA (service-oriented architecture)** is an architectural style that focus on discrete services instead of a monolithic design.^[1] By consequence, it is as well applied in the field of [software design](#) where services are provided to the other components by [application components](#), through a [communication protocol](#) over a network. A service is a discrete unit of functionality that can be accessed remotely and acted upon and updated independently, such as retrieving a credit card statement online. SOA is also intended to be independent of vendors, products and technologies.

A **web service** is any piece of software that makes itself available over the internet and uses a standardized XML messaging system



BITS Pilani
Pilani Campus

BITS Pilani presentation

Mridul Moitra
Cloud Computing



BITS Pilani

<CSI ZG527 / SS ZG527 / SE ZG527
Cloud Computing
Lecture No. 2

Overview

- **Introduction to Virtualization and Hypervisors**
- Origins of Hypervisors
- Hypervisors and its Classification
- Existing Solutions/Products
- Advantages and Disadvantages
- State of the Art
- Future of Hypervisors and Virtualization

Introduction

Virtualization is the simulation of the software and/or

- hardware upon which other software runs. This
- simulated environment is called virtual machine. Each
- VM can run its own operating systems and
- applications as if it were in a physical machine. So It is
- way to run multiple operating systems on the same
- hardware at the same time.
- ☐ For e.g., Windows and Linux both can run on the same
- laptop at the same time

Origin of Hypervisor

- IBM first developed CP/CMS operating system in 1967, an attempt to build time-sharing systems for mainframe systems
- In 1972, IBM's zSeries line featured Virtualization
- Early acceptance and rapid development by developers all over
- In 1985, IBM introduced the PR/SM hypervisor to manage logical partitions
- Other companies, Sun Microsystems, HP, and SGI joined the race and started selling virtualized software around 2000

What is Virtualization

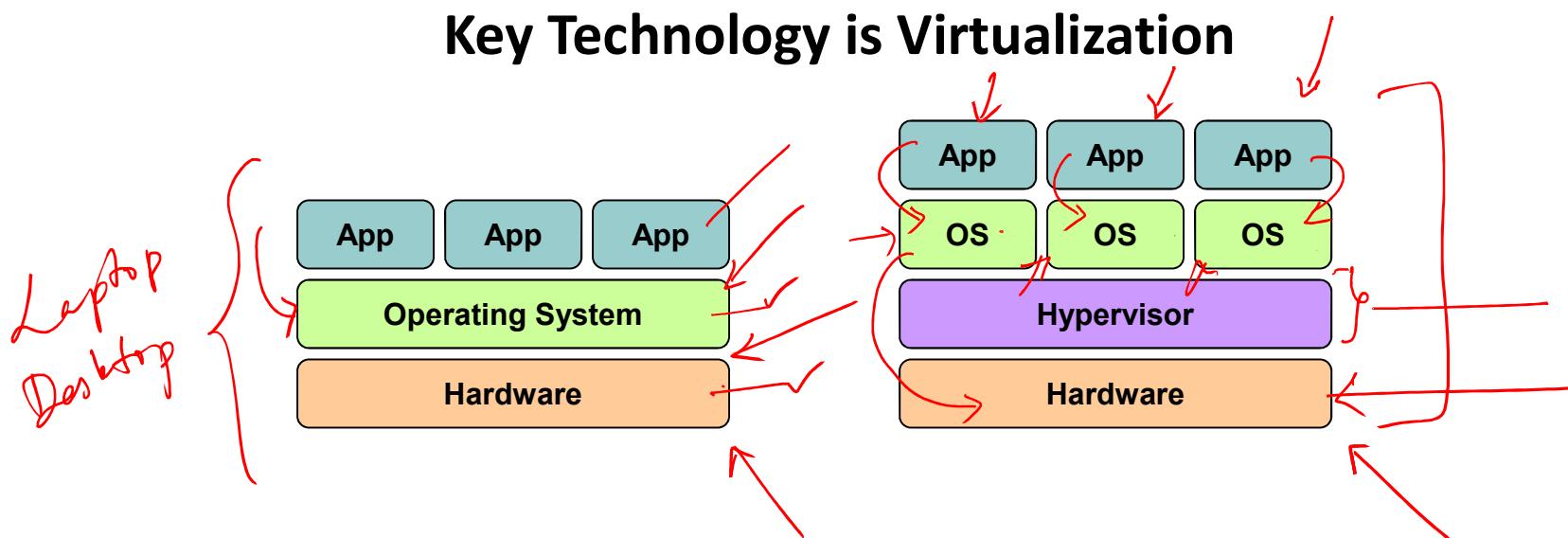
- Before using Virtualization, we had:
 - Single OS per machine
 - Software and hardware tightly coupled
 - Underutilized resources (idle time)
 - Inflexibility
- Virtualization gives you:
 - Hardware independence of operation system and applications
 - Ability to encapsulate OS and applications in to virtual machines
 - Ability to provision virtual machines to any system

Classification of Hypervisors

There are two types of hypervisor based on architecture:

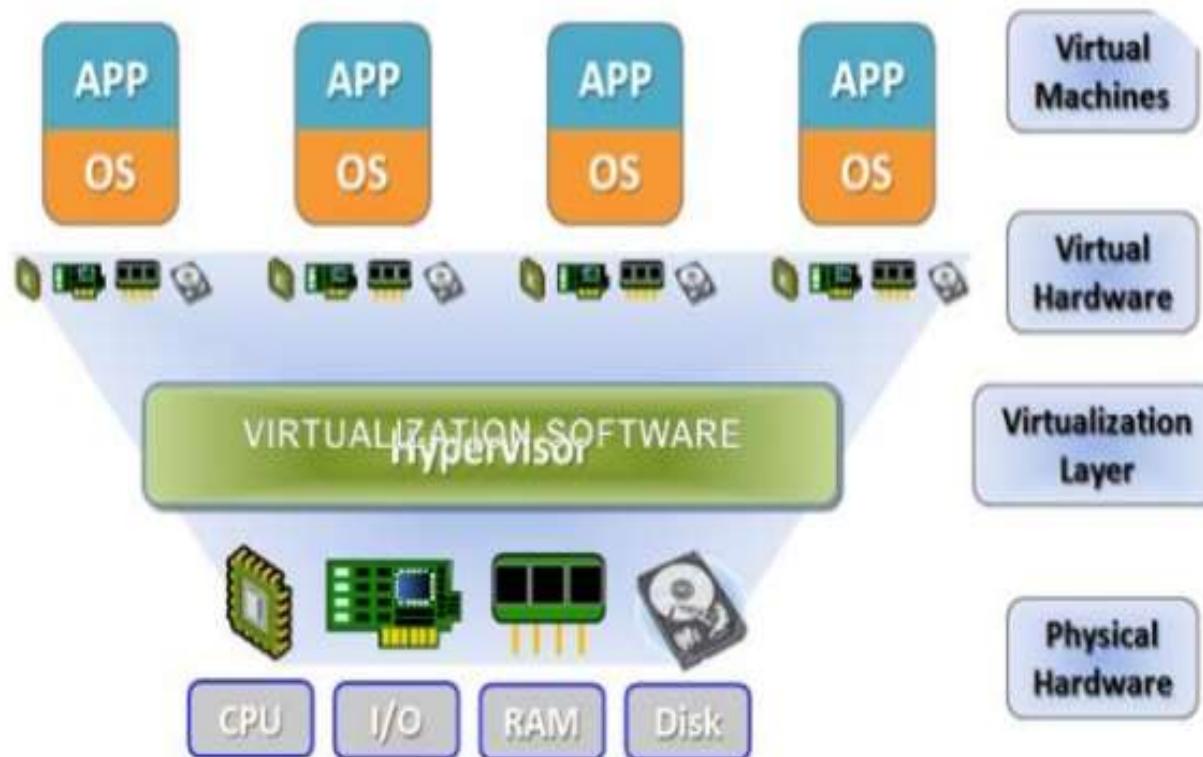
- **Type 1 Hypervisor**
 - also known as native or bare metal hypervisor
- **Type 2 Hypervisor**
 - also known as hosted hypervisor

Technology made cloud possible



Virtualization plays an important role as an enabling technology for datacentre implementation by abstracting compute, network, and storage service platforms from the underlying physical hardware

What is Virtualization



How do we Get Virtualization

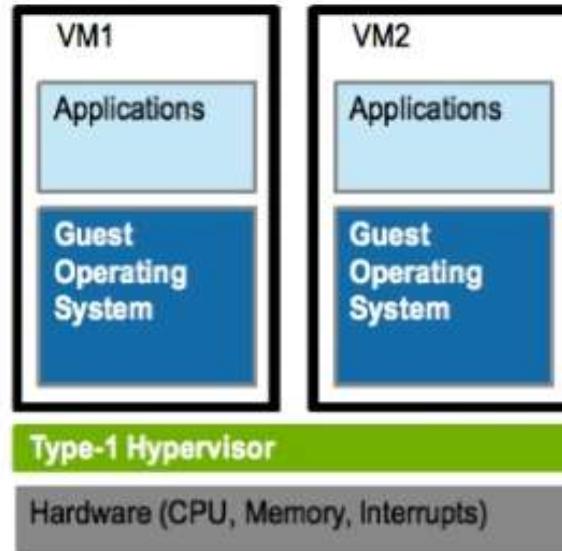
How do we get Virtualization?

- **Hypervisors:** The approach to virtualization
- **Hypervisor** also known as the Virtual Machine Monitor
- Software that allows multiple operating systems to share a single hardware host
- Emulates hardware resources to guest operating systems
- Each operating system appears to have host's processor, memory, and other resources all to itself
- In reality, hypervisor controls the resources and allocates them as needed by each guest OS in a synchronized

Classification of Hypervisors

Type 1 Hypervisor

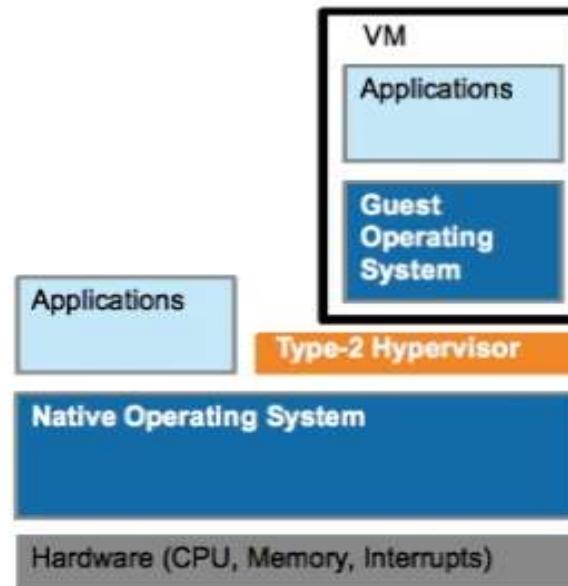
- Runs directly on the host's hardware to manage guest operating systems
- Does not require any base server operating system
- Direct access to hardware resources
- Better performance, scalability, and stability
- However, hardware support is limited



Classification of Hypervisor

Type 2 Hypervisor

- Hosted on the main operating system
- Basically a software installed on an OS
- Hypervisor asks OS to make hardware calls
- Better compatibility with hardware
- Increased overhead affects performance

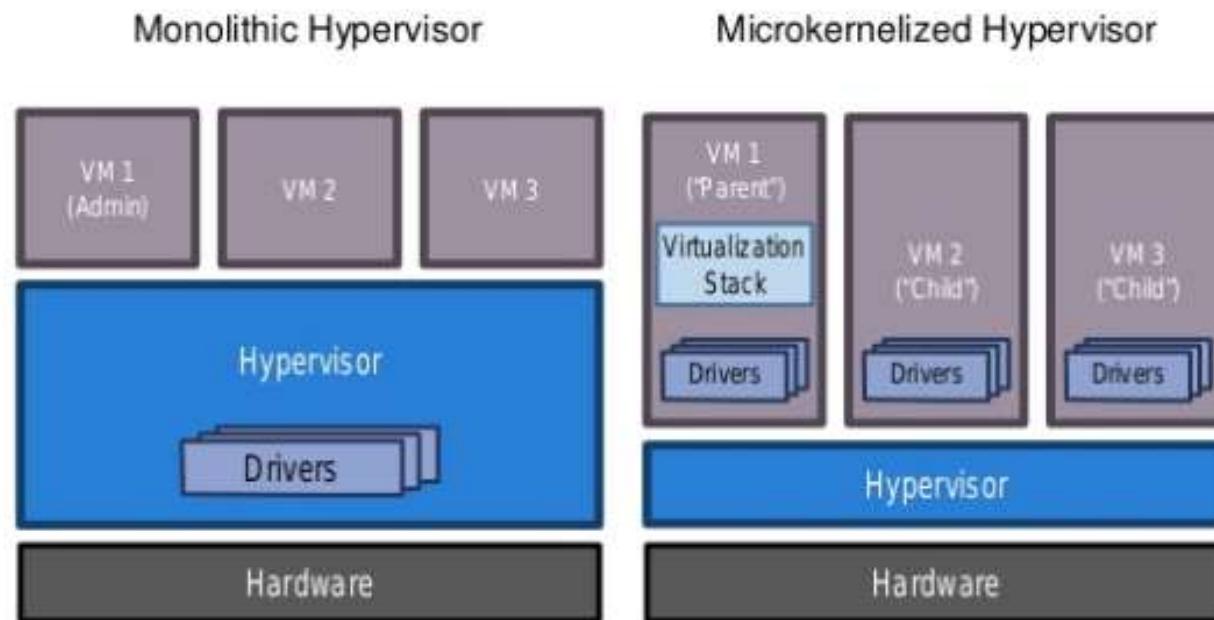


Classification of type I VMM

Type 1 Hypervisors can be further classification into two main ways to architect the hypervisor solutions:

- **Monolithic**
 - Hosts the hypervisor/VMM in a single layer that also includes most of the required components, such as the kernel, device drivers, and the I/O stack
- **Microkernelized**
 - Uses a very thin, specialized hypervisor that only performs the core tasks of ensuring partition isolation and memory management. This layer does not include the I/O stack or device drivers.
 - Virtualization stack and hardware-specific device drivers are located in a specialized partition called the parent partition.

Classification of Type I VMM



Virtualization Techniques

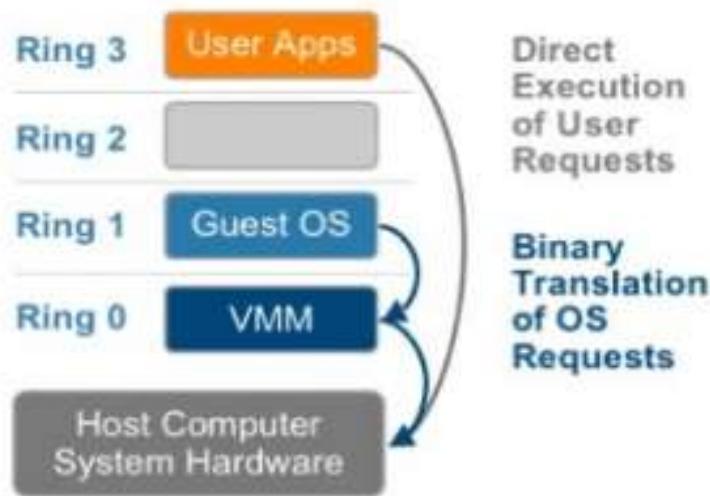
There are multiple approaches to running the guest operating system. These include:

- **Full Virtualization**
- **Paravirtualization**
 - Also known as OS assisted virtualization
- **Hardware-assisted virtualization**
 - Also known as accelerated virtualization, hardware virtual machine (HVM)

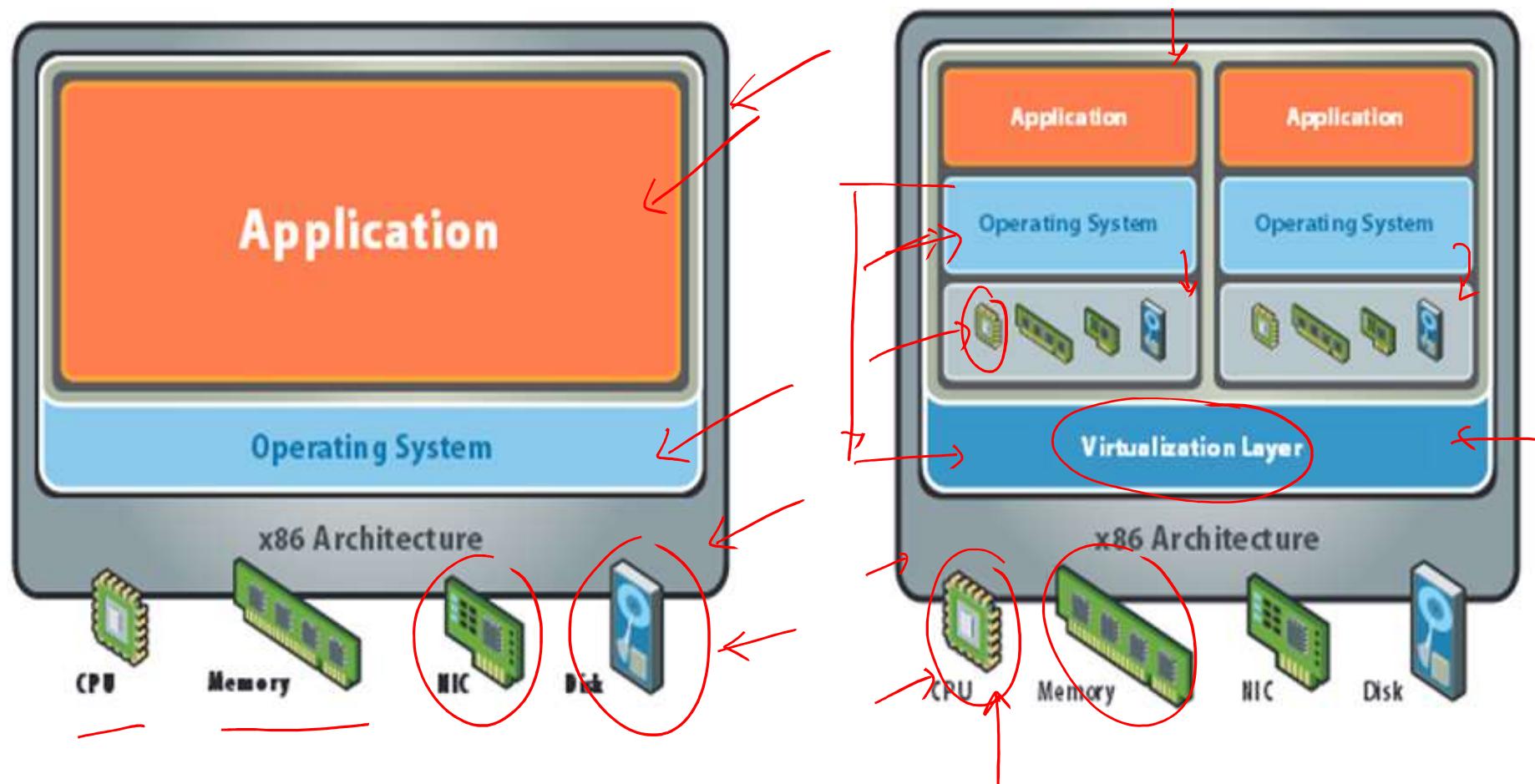
Virtualization Techniques

Full Virtualization

- Completely abstracted from the underlying hardware by virtualization layer
- Guest OS unaware that it is a guest
- Hypervisor translates all OS calls on-the-fly
- No hardware assistance or modification; flexibility



What is Virtualization



Importance of Virtualization in Cloud Computing

- Cloud makes notion of “Pay for what you use”, “infinite availability- use as much you want”.
- These notions are practical only if we have
 - lot of flexibility
 - efficiency in the back-end.
- This efficiency is readily available in Virtualized Environments and Machines

What does Virtualization do?

- Virtualization allows multiple operating system instances to run concurrently on a single computer
- It is a means of separating hardware from a single operating system.
- Each “guest” OS is managed by a Virtual Machine Monitor (VMM), also known as a hypervisor.
- Because the virtualization system sits between the guest and the hardware, it can control the guests’ use of CPU, memory, and storage, even allowing a guest OS to migrate from one machine to another.
- Instead of purchasing and maintaining an entire computer for one application, each application can be given its own operating system, and all those operating systems can reside on a single piece of hardware.
- Virtualization allows an operator to control a guest operating system’s use of CPU, memory, storage, and other resources, so each guest receives only the resources that it needs.

Changes after Virtualization

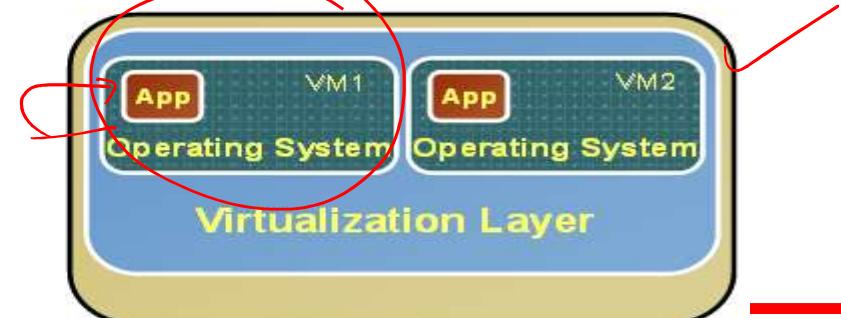
Before Virtualization

- Single OS image per machine
- Software and hardware tightly coupled
- Running multiple applications on same machine often creates conflict
- Underutilized resources
- Inflexible and costly infrastructure



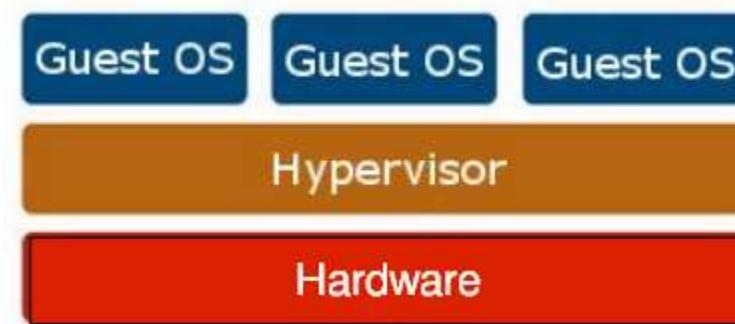
After Virtualization

- Hardware-independence of operating system and applications
- Virtual machines can be provisioned to any system
- Can manage OS and application as a single unit by encapsulating them into virtual machines



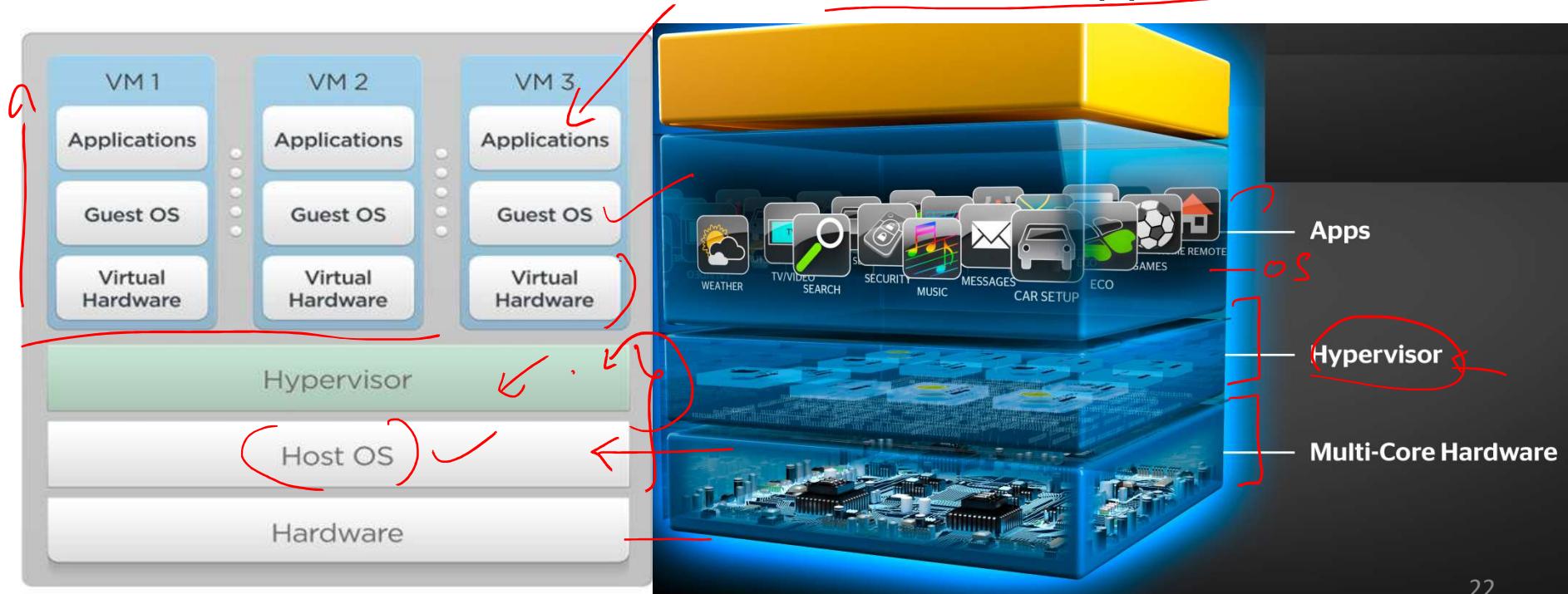
Virtualization Architecture

- OS assumes complete control of the underlying hardware.
- Virtualization architecture provides this illusion through a hypervisor/VMM.
- Hypervisor/VMM is a software layer which:
 - Allows multiple Guest OS (Virtual Machines) to run simultaneously on a single physical host
 - Provides hardware abstraction to the running Guest OSs and efficiently multiplexes underlying hardware resources



Hypervisor

A thin layer of software that generally provides virtual partitioning capabilities which runs directly on hardware, but underneath higher-level virtualization services. Sometimes referred to as a “bare metal” approach.



Hypervisor Design Goals

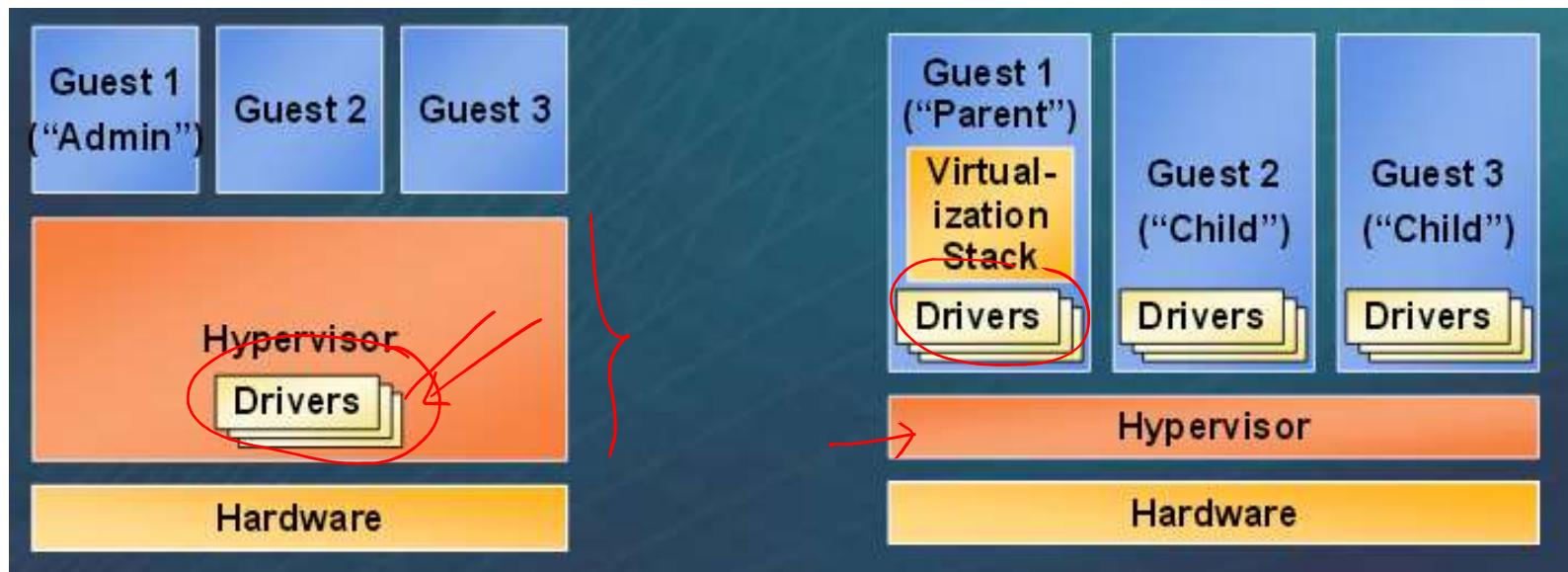
- Isolation ✓
 - Security isolation ✓
 - Fault isolation ✓
 - Resource isolation ✓✓
- Reliability ✓
 - Minimal code base
 - Strictly layered design
 - Not extensible ✓
- Scalability ✓
 - Scale to large number of cores
 - Large memory systems ✓

How Hypervisor goals are achieved?

- Partitioning Kernel ✓
 - “Partition” is isolation boundary]
 - Few virtualization functions; relies on virtualization stack
- Very thin layer of software✓
 - Microkernel ✓
 - Highly reliable ✓
 - Basis for smaller Trusted Computing Base (TCB)
- No device drivers ✓
 - Drivers run in a partition ✓
- Well-defined interface]
 - Allow others to create support for their OSes as guests

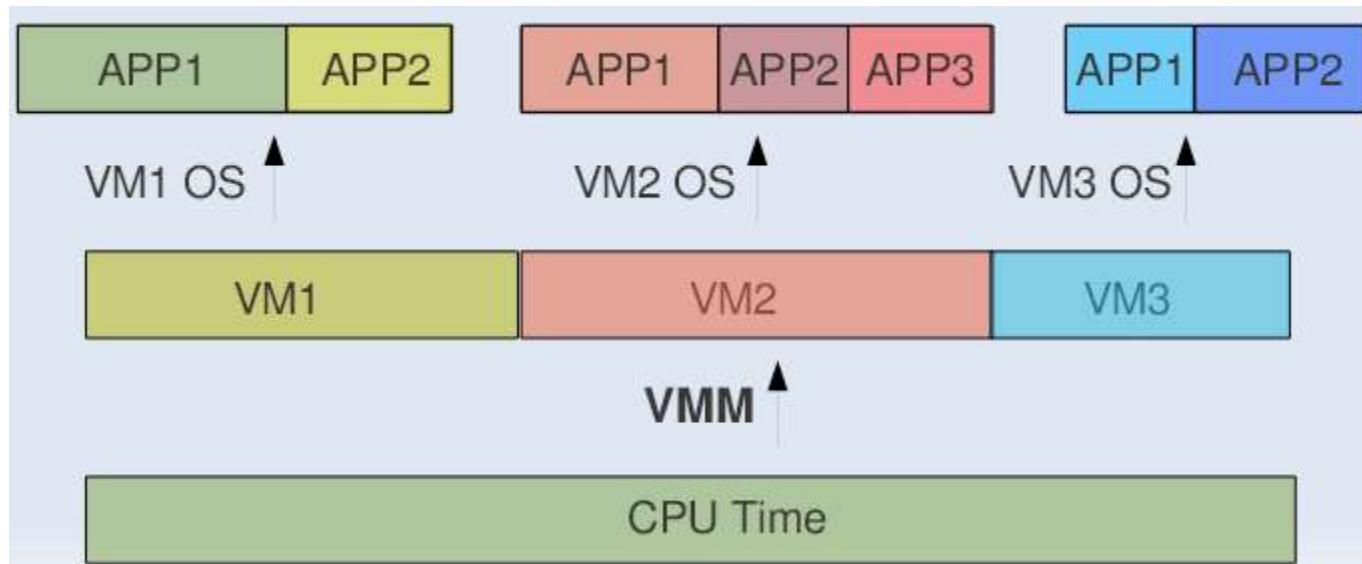
Hypervisor Monolithic versus Microkernelized

- Monolithic hypervisor ✓
 - Simpler than a modern kernel, but still complex
 - Contains its own drivers model
- Microkernelized hypervisor
 - Simple partitioning functionality
 - Increase reliability and minimize lowest level of the TCB
 - No third-party code ✓
 - Drivers run within guests ✓



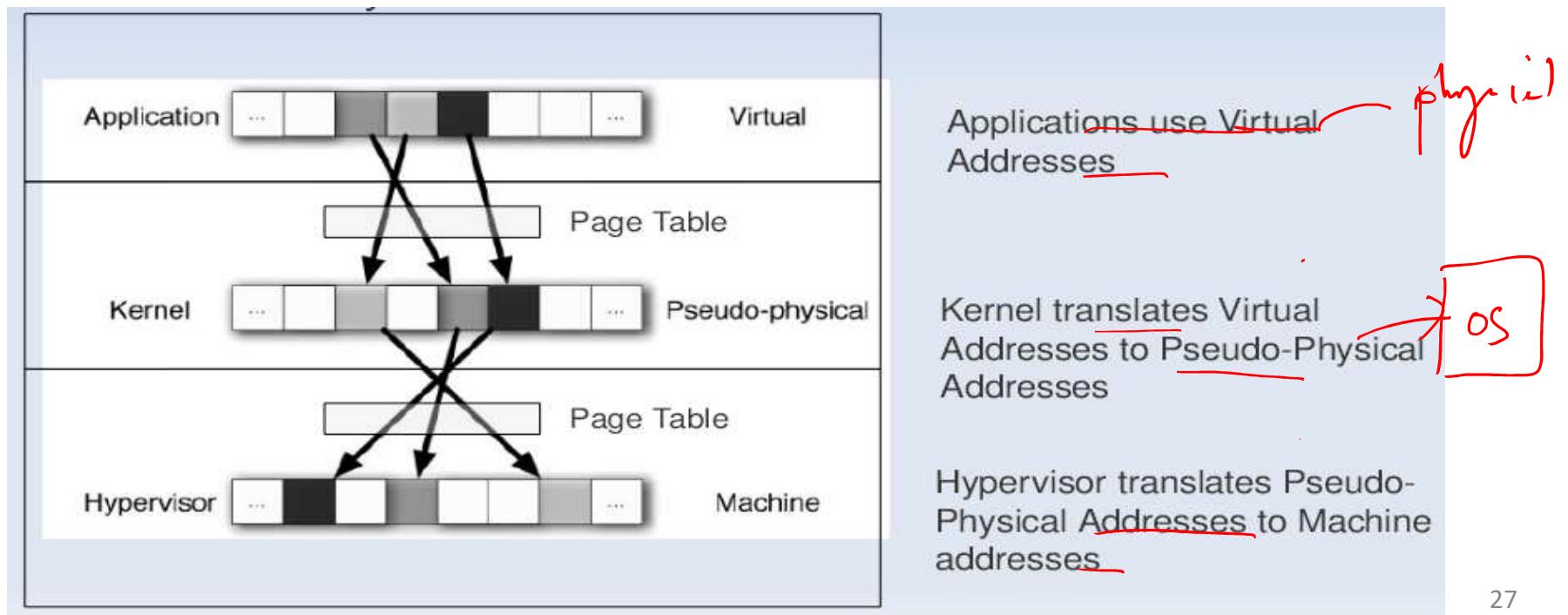
CPU Sharing

- VMM or Hypervisor provides a virtual view of CPU to VMs.
- In multi processing, CPU is allotted to the different processes in form of time slices by the OS.
- Similarly VMM or Hypervisor allots CPU to different VMs.



Memory Sharing

- In Multiprogramming there is a single level of indirection maintained by Kernel.
- In case of Virtual Machines there is one more level of indirection maintained by VMM



IO Sharing

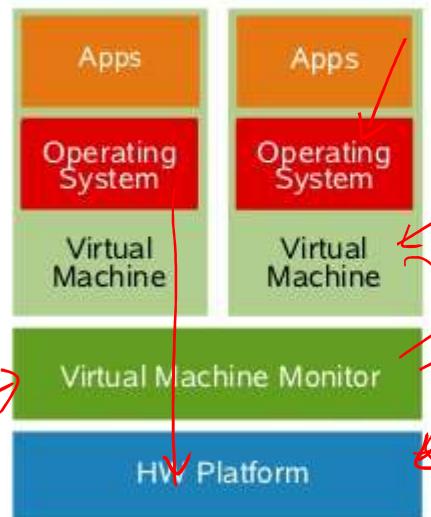
- Device needs to use Physical Memory location.
- In a virtualized environment, the kernel is running in a hypervisor-provided virtual address space
- Allowing the guest kernel to convey an arbitrary location to device for writing is a serious security hole
- Each device defines its own protocol for talking to drivers

Approaches for Virtualization

Full & Paravirtualization Overview

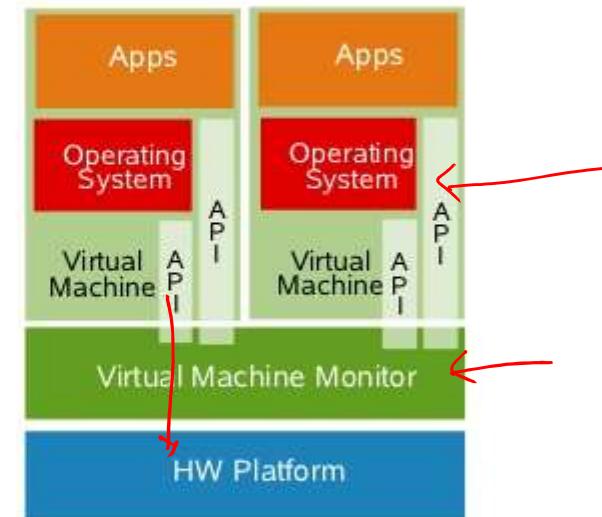
N

Full Virtualization ✓



Runtime modification of Guest OS:
VMM manages the conflict, then
returns to OS

Paravirtualization ✓



Static modification of Guest OS prior to
runtime: Privileged instruction calls are
exchanged with API functions provided
by the VMM
– Almost no performance degradation
– Significant scalability

Full Virtualization

❑ Full virtualization

- In its basic form known as “full virtualization” the hypervisor provides a fully emulated machine in which an operating system can run. VMWare is a good example.
- The biggest advantage to this approach is its flexibility: one could run a RISC-based OS as a guest on an Intel-based host.
- While this is an obvious approach, there are significant performance problems in trying to emulate a complete set of hardware in software.

VM Implementation Techniques

- Binary Translation
- Paravirtualization
- Hardware Supported Virtualization

Binary Translation

- Used in VMWare

Binary image of OS is manipulated at the runtime.

- Privileged instructions are rewritten to point to their emulated versions.
- Performance from this approach is not ideal particularly when doing anything I/O intensive.
- Caching of the locations of unsafe instructions can speed Up

Paravirtualization

- Used in XEN
- Make OS aware of underlying Virtualization env.
- OS's code is manipulated.
- Important system calls are changed to point to the implementation provided by the VMM.

HW Supported Virtualization

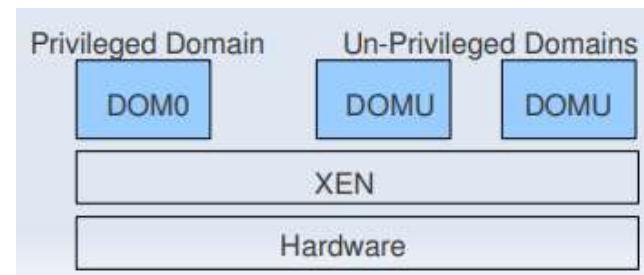
- Added new instructions which makes Virtualization considerably easier for x86.
- Intel – IVT(Intel Virtualization Technology)
- AMD – introduced AMD-V
- OS stays in its original privilege level 0.
- Attempts to access the hardware directly are caught and passed to VMM.
- In other words a new privilege ring is setup for the VMM

XEN

- XEN Domains
- CPU Sharing
- Hyper Calls
- Memory Sharing
- IO Sharing
- XEN Split Driver Technique
- IO Ring

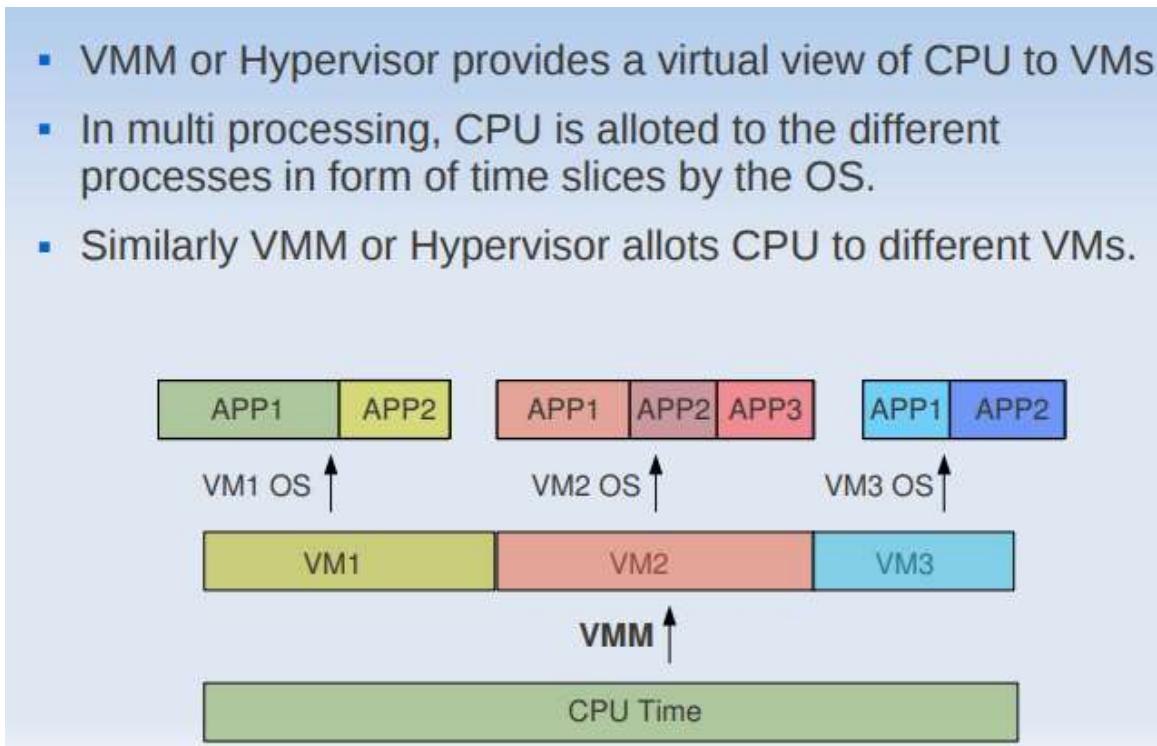
XEN Domains

- Xen runs guests in environments known as domains which encapsulate a complete running virtual environment
- There are two types pf Domains:
- DomU -the “U” stands for unprivileged.
- Guest OSs run in this domain.
- Dom0 has elevated privileges
- Provides device drivers
- Provides tools/mechanisms to configure Virtualization environment

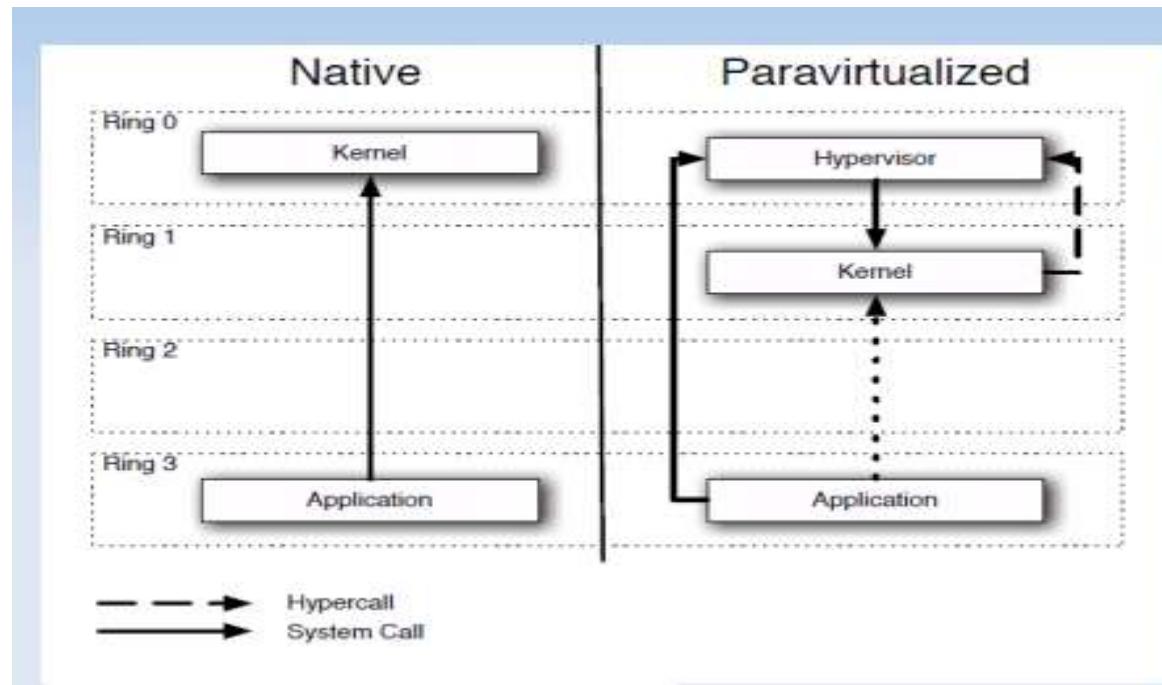


CPU Sharing

- VMM or Hypervisor provides a virtual view of CPU to VMs.
- In multi processing, CPU is allotted to the different processes in form of time slices by the OS.
- Similarly VMM or Hypervisor allots CPU to different VMs.

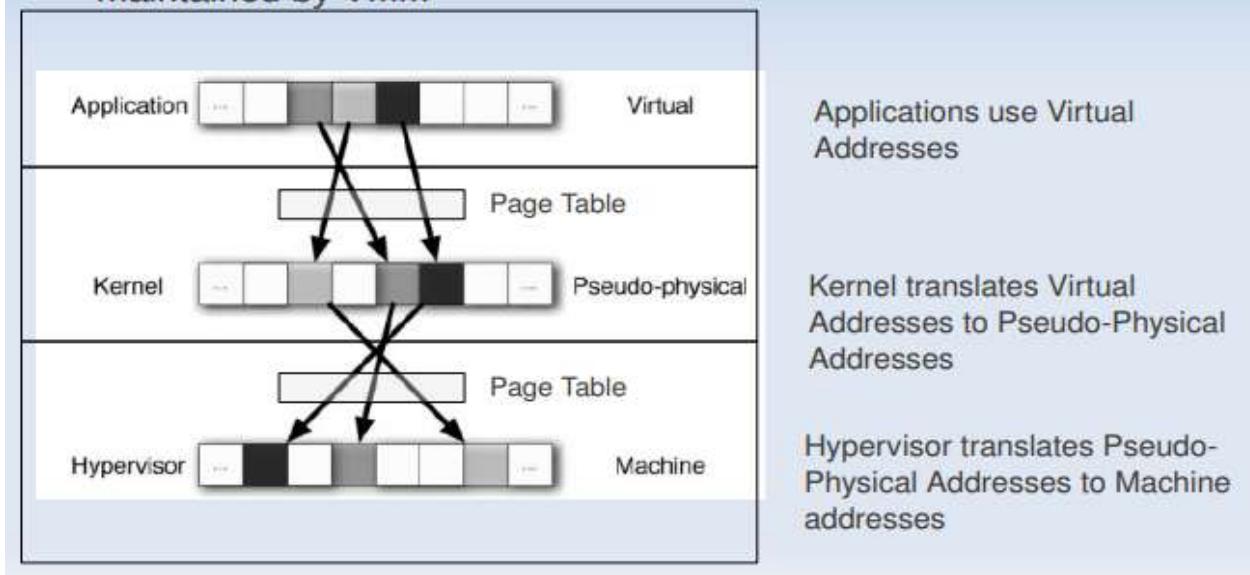


XEN Hypervisor



Memory Sharing

- In Multiprogramming there is a single level of indirection maintained by Kernel.
- In case of Virtual Machines there is one more level of indirection maintained by VMM

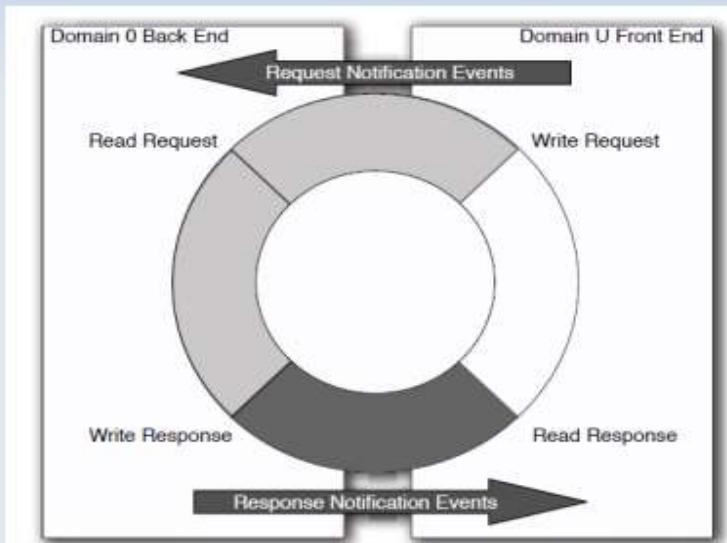


IO Sharing

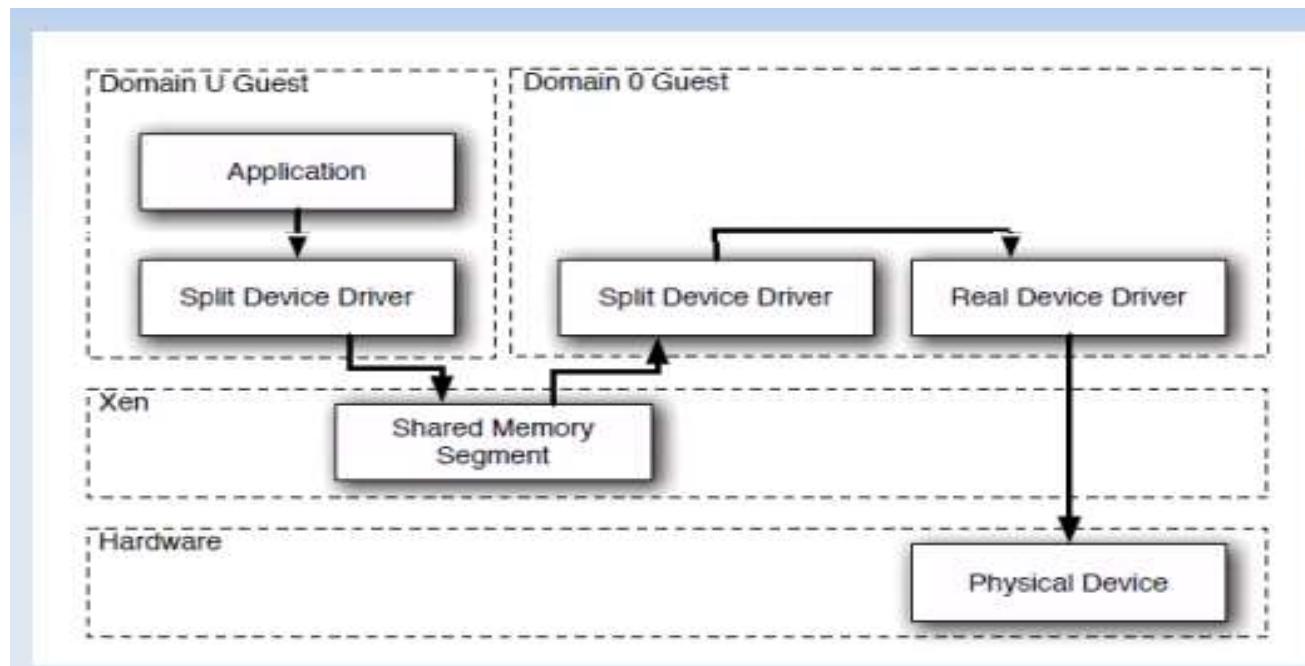
- DMA Problem
- Device needs to use Physical Memory location.
- In a virtualized environment, the kernel is running in a hypervisor provided virtual address space
- Allowing the guest kernel to convey an arbitrary location to device for writing is a serious security hole
- Detecting a DMA instruction is nontrivial. Each device defines its own protocol for talking to drivers.
- XEN Follows Split Driver Model: Dom 0 does the IO on behalf of all the other guests.
- As DOM0 is privileged the IO has no problem

IO Ring

Shared memory is used with event based synchronization



XEN IO Split Device Driver

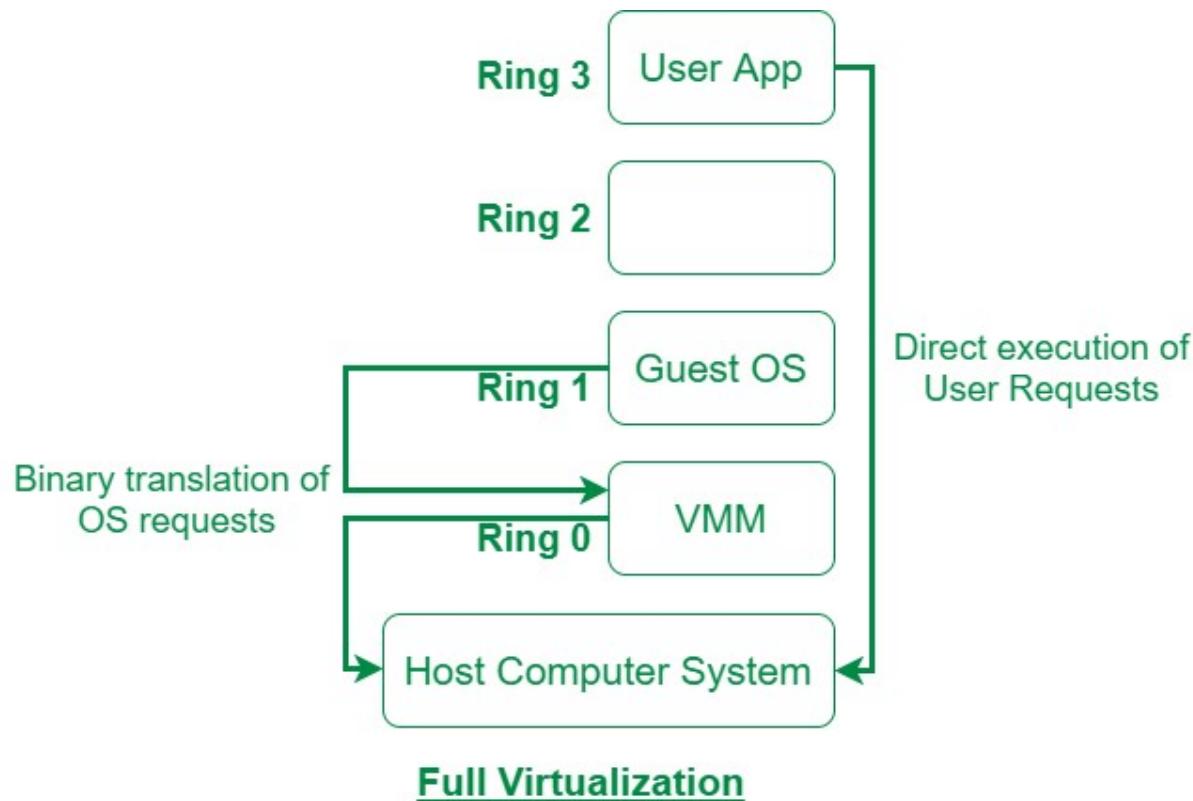


Conclusions

- Notion of Cloud is possible without Virtualization, but it will be inefficient and inflexible.
- Virtualization is an attempt to manage OS.
- There are many levels and many ways to
- implement Virtualization.

Full Virtualization

Full Virtualization: Full Virtualization was introduced by IBM in the year 1966. It is the first software solution for server virtualization and uses binary translation and direct approach techniques. In full virtualization, guest OS is completely isolated by the virtual machine from the virtualization layer and hardware



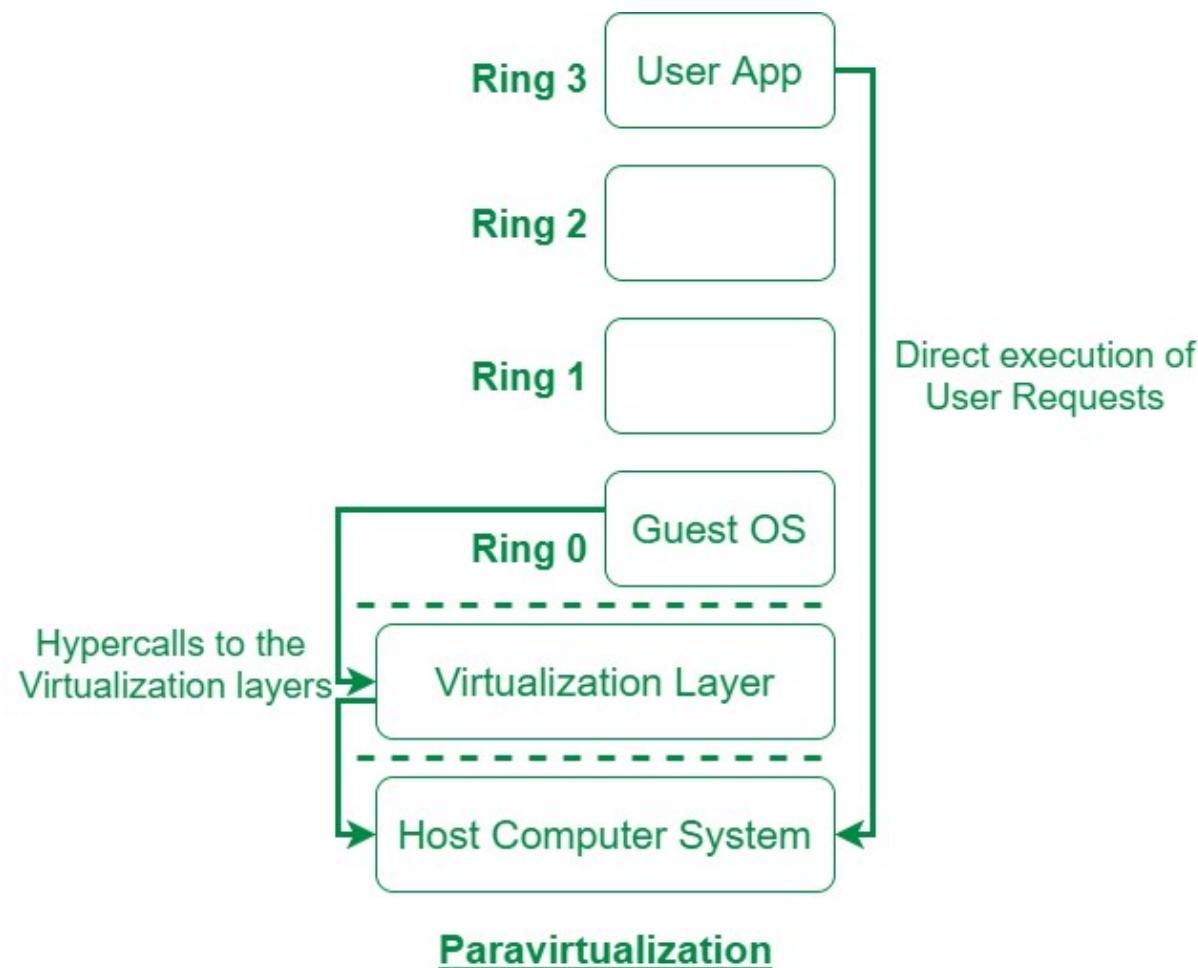
ParaVirtualization

□ Paravirtualization

- “Paravirtualization,” found in the XenSource, open source Xen product, attempts to reconcile these two approaches. Instead of emulating hardware, paravirtualization uses slightly altered versions of the operating system which allows access to the hardware resources directly as managed by the hypervisor.
- This is known as hardware-assisted virtualization, and improves performance significantly
- In order to retain flexibility, the guest OS is not tied to its host OS. Drastically different operating systems can be running in a hypervisor at the same time, just as they can under full virtualization.
- In this way, paravirtualization can be thought of as a low-overhead full virtualization

Para Virtualization

Paravirtualization is the category of CPU virtualization which uses hypercalls for operations to handle instructions at compile time. In paravirtualization, guest OS is not completely isolated but it is partially isolated by the virtual machine from the virtualization layer and hardware

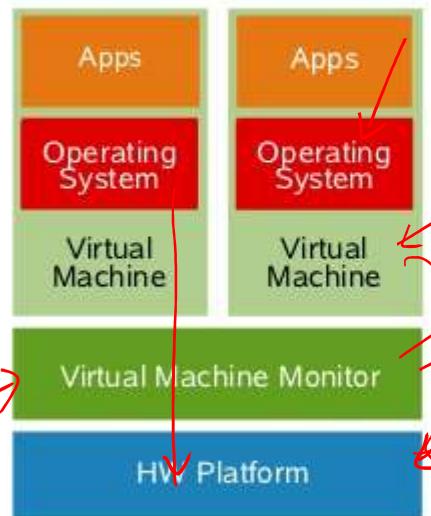


Approaches for Virtualization

Full & Paravirtualization Overview

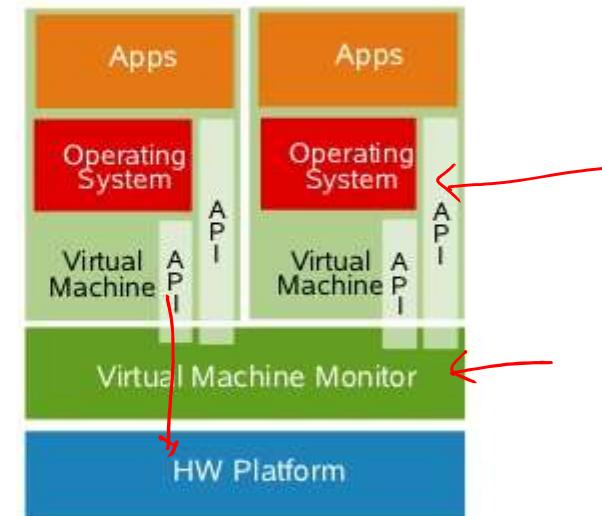
N

Full Virtualization ✓



Runtime modification of Guest OS:
VMM manages the conflict, then
returns to OS

Paravirtualization ✓



Static modification of Guest OS prior to runtime: Privileged instruction calls are exchanged with API functions provided by the VMM
– Almost no performance degradation
– Significant scalability

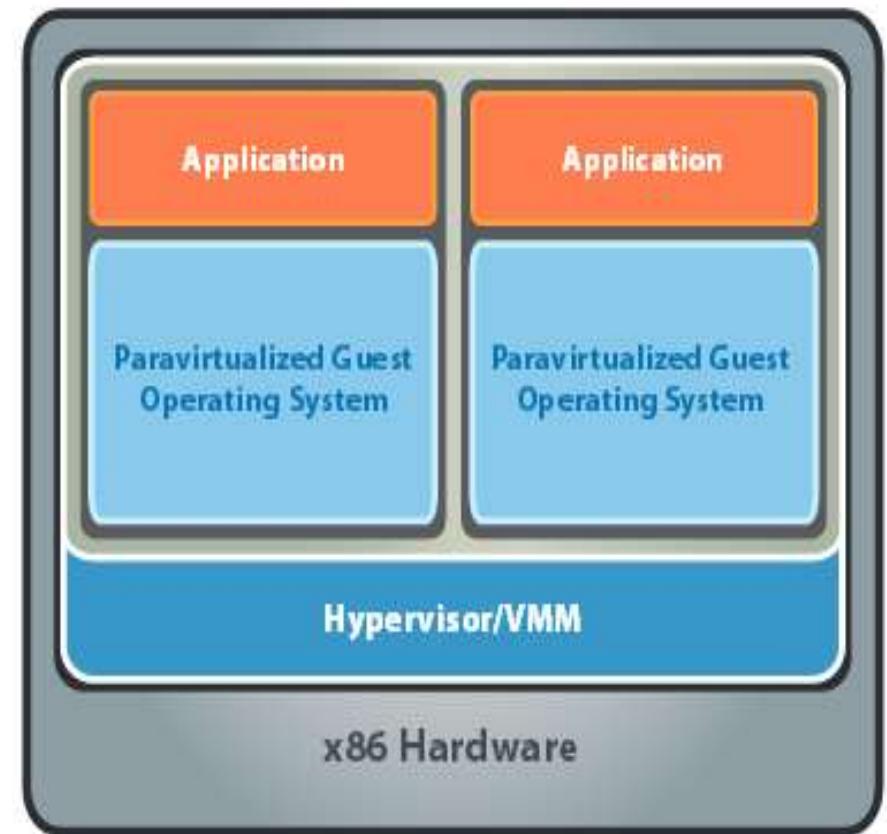
SKI Virtualization

- Single Kernel Image (SKI),
 - Single Kernel Image (SKI), in which the host OS spawns additional copies of itself. This kind of virtualization can be found in Swsoft Virtuozzo and Sun Solaris, Zones. SKI can be thought of as “lightweight” virtualization.
 - While this approach avoids the performance problems with pure emulation, it does so at the expense of flexibility.
 - It is not possible, for instance, to run different versions or even different patch levels of a particular operating system on the same machine.
 - Whatever versions exist in the host, that same software will be provided in the guest. SKI also sacrifices the security and reliability provided by other virtualization methods.

S.No.	Full Virtualization	Paravirtualization
1.	In Full virtualization, virtual machines permit the execution of the instructions with the running of unmodified OS in an entirely isolated way.	In paravirtualization, a virtual machine does not implement full isolation of OS but rather provides a different API which is utilized when OS is subjected to alteration.
2.	Full Virtualization is less secure.	While the Paravirtualization is more secure than the Full Virtualization.
3.	Full Virtualization uses binary translation and a direct approach as a technique for operations.	While Paravirtualization uses hypercalls at compile time for operations.
4.	Full Virtualization is slow than paravirtualization in operation.	Paravirtualization is faster in operation as compared to full virtualization.
5.	Full Virtualization is more portable and compatible.	Paravirtualization is less portable and compatible.
6.	Examples of full virtualization are Microsoft and Parallels systems.	Examples of paravirtualization are Microsoft Hyper-V, Citrix Xen, etc.
7.	It supports all guest operating systems without modification.	The guest operating system has to be modified and only a few operating systems support it.
8.	The guest operating system will issue hardware calls.	Using the drivers, the guest operating system will directly communicate with the hypervisor.
9.	It is less streamlined compared to para-virtualization.	It is more streamlined.
10.	It provides the best isolation.	It provides less isolation compared to full virtualization.

x86 Hardware Virtualization

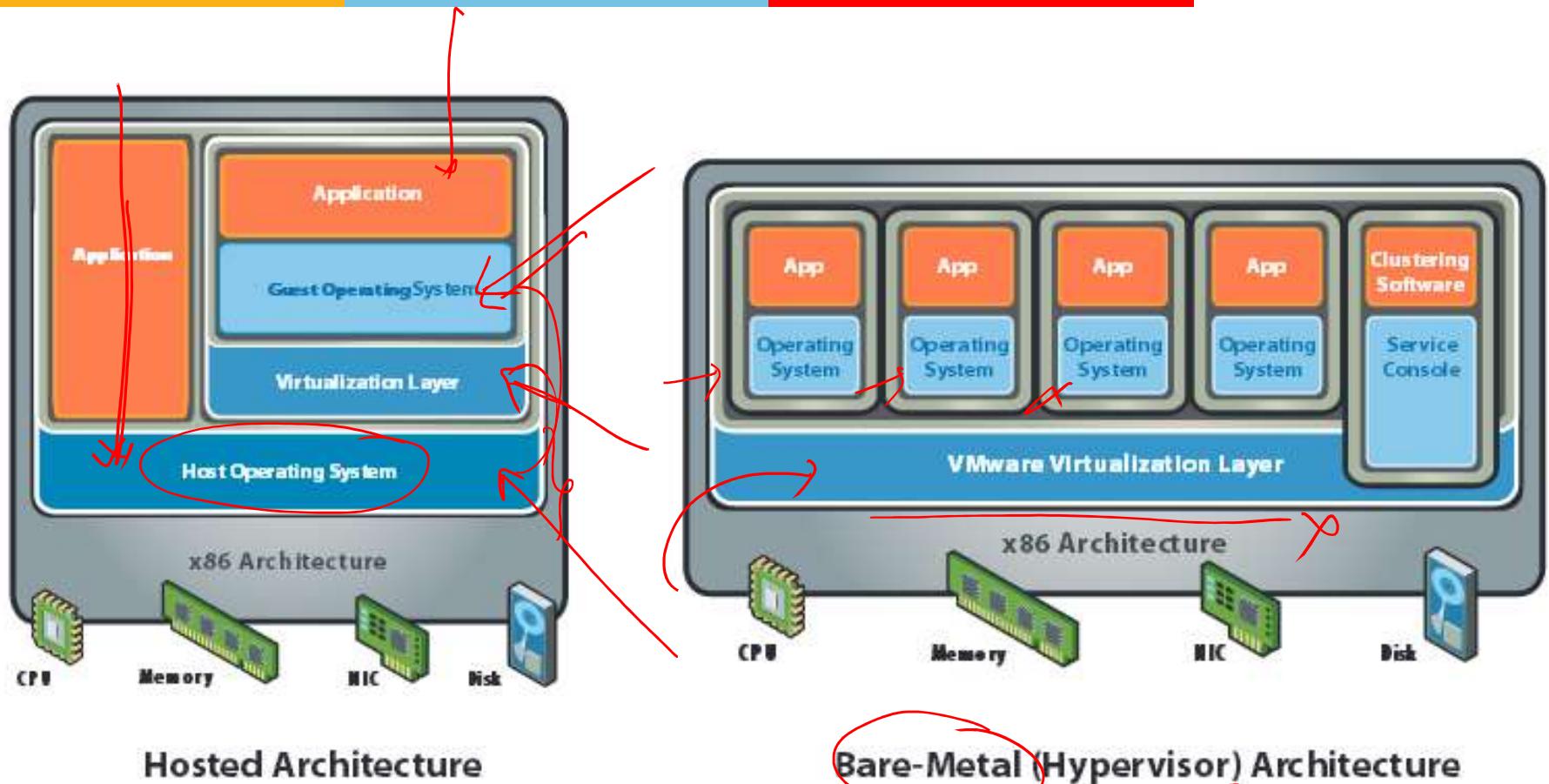
- The latest generation of x86-based systems feature processors with 64-bit extensions supporting very large memory capacities.
- This enhances their ability to host large, memory-intensive applications, as well as allowing many more virtual machines to be hosted by a physical server deployed within a virtual infrastructure.
- The continual decrease in memory costs will further accelerate this trend.



x86 Hardware Virtualization

- For Industry-standard x86 systems, the two approaches typically used with software-based partitioning are
 - hosted and
 - hypervisor architectures
- A hosted approach provides partitioning services on top of a standard operating system and supports the broadest range of hardware configurations.
- In contrast, a hypervisor architecture is the layer of software installed on a clean x86-based system (hence it is often referred to as a “bare metal” approach). Since it has direct access to the hardware resources, a hypervisor is more efficient than hosted architectures, enabling greater scalability, robustness and performance

x86 Hardware Virtualization



Advantages of Virtualization

- Instant provisioning - fast scalability
- Live Migration is possible
- Load balancing and consolidation in a Data Center
is possible.
- Low downtime for maintenance ✓
- Virtual hardware supports legacy operating
systems efficiently
- Security and fault isolation ✓

Advantages of Virtualization

Security: by compartmentalizing environments with different security requirements in different virtual machines one can select the guest operating system and tools that are more appropriate for each environment. For example, we may want to run the Apache web server on top of a Linux guest operating system and a backend MS SQL server on top of a guest Windows XP operating system, all in the same physical platform. A security attack on one virtual machine does not compromise the others because of their isolation.

Advantages of Virtualization

Reliability and availability: A software failure in a virtual machine does not affect other virtual machines.

Cost: It is possible to achieve cost reductions by consolidation smaller servers into more powerful servers. Cost reductions stem from hardware cost reductions (economies of scale seen in faster servers), operations cost reductions in terms of personnel, floor space, and software licenses. VMware cites overall cost reductions ranging from 29 to 64%

Advantages of Virtualization

Adaptability to Workload Variations: Changes in workload intensity levels can be easily taken care of by shifting resources and priority allocations among virtual machines. Autonomic computing-based resource allocation techniques, such as the ones in can be used to dynamically move processors from one virtual machine to another.

Load Balancing: Since the software state of an entire virtual machine is completely encapsulated by the VMM, it is relatively easy to migrate virtual machines to other platforms in order to improve performance through better load balancing

Advantages of Virtualization

Legacy Applications: Even if an organization decides to migrate to a different operating system, it is possible to continue to run legacy applications on the old OS running as a guest OS within a VM. This reduces the migration cost.

Issues to be aware of

- **Software licensing** ✓
One of the most significant virtualization-related issues to be aware of is software licensing. Virtualization makes it easy to create new servers, but each VM requires its own separate software license. Organizations using expensive licensed applications could end up paying large amounts in license fees if they do not control their server sprawl.
- **IT training** ✓
IT staff used to dealing with physical systems will need a certain amount of training in virtualization. Such training is essential to enable the staff to debug and troubleshoot issues in the virtual environment, to secure and manage VMs, and to effectively plan for capacity.
- **Hardware investment** ✓
Server virtualization is most effective when powerful physical machines are used to host several VMs. This means that organizations that have existing not-so-powerful hardware might still need to make upfront investments in acquiring new physical servers to harvest the benefits of virtualization

Issues to be aware of

- Interoperability among vendor products is still evolving.
- Failure of the virtualization device, leading to loss of the mapping table.

Applications of Virtualization

- Today, virtualization can apply to a range of system layers, including hardware-level virtualization, operating system-level virtualization, and high-level language virtual machines.
- **Maximize resources** — Virtualization can reduce the number of physical systems you need to acquire, and you can get more value out of the servers. Most traditionally built systems are underutilized. Virtualization allows maximum use of the hardware investment.
- **Multiple systems** — With virtualization, you can also run multiple types of applications and even run different OS for those applications on the same physical hardware.
- **IT budget integration** — When you use virtualization, management, administration and all the attendant requirements of managing your own infrastructure remain a direct cost of your IT operation.

Technology Trends

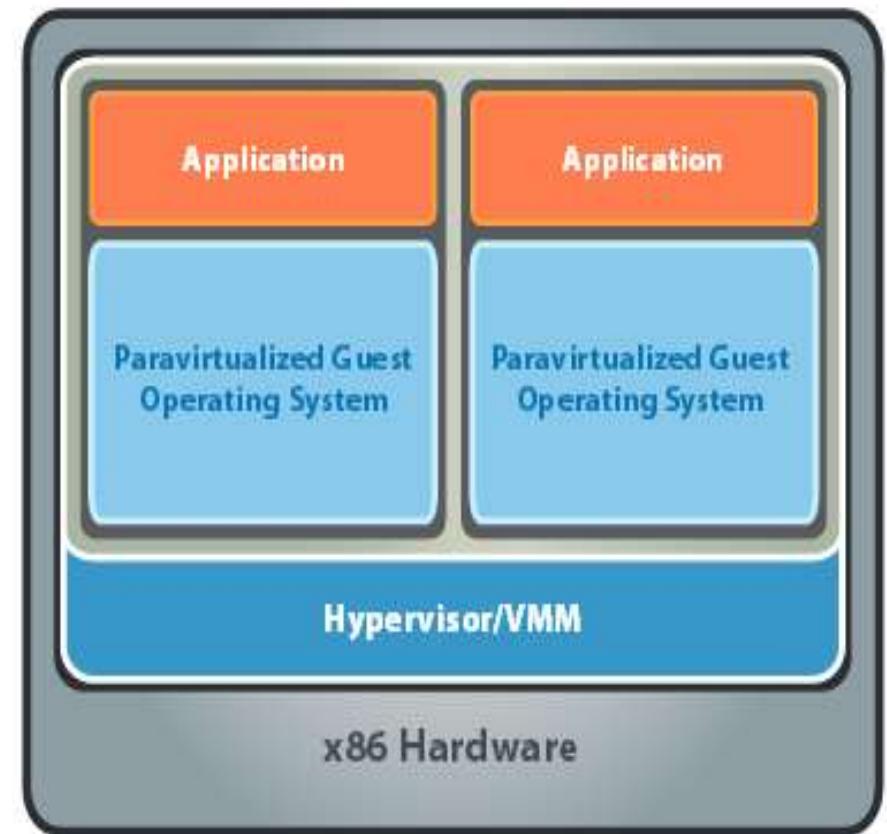
- Virtualization is Key to Exploiting Trends
- Allows most efficient use of the compute resources
 - Few apps take advantage of 16+ CPUs and huge memory as well as virtualization
 - Virtualization layer worries about NUMA, not apps
- Maximize performance per watt across all servers
 - Run VMs on minimal # of servers, shutting off the others
 - Automated, live migration critical:
 - Provide performance guarantees for dynamic workloads
 - Balance load to minimize number of active servers
- Stateless, Run-anywhere Capabilities
 - Shared network and storage allows flexible mappings
 - Enables additional availability guarantees

SKI Virtualization

- Single Kernel Image (SKI),
 - Single Kernel Image (SKI), in which the host OS spawns additional copies of itself. This kind of virtualization can be found in Swsoft Virtuozzo and Sun Solaris, Zones. SKI can be thought of as “lightweight” virtualization.
 - While this approach avoids the performance problems with pure emulation, it does so at the expense of flexibility.
 - It is not possible, for instance, to run different versions or even different patch levels of a particular operating system on the same machine.
 - Whatever versions exist in the host, that same software will be provided in the guest. SKI also sacrifices the security and reliability provided by other virtualization methods.

x86 Hardware Virtualization

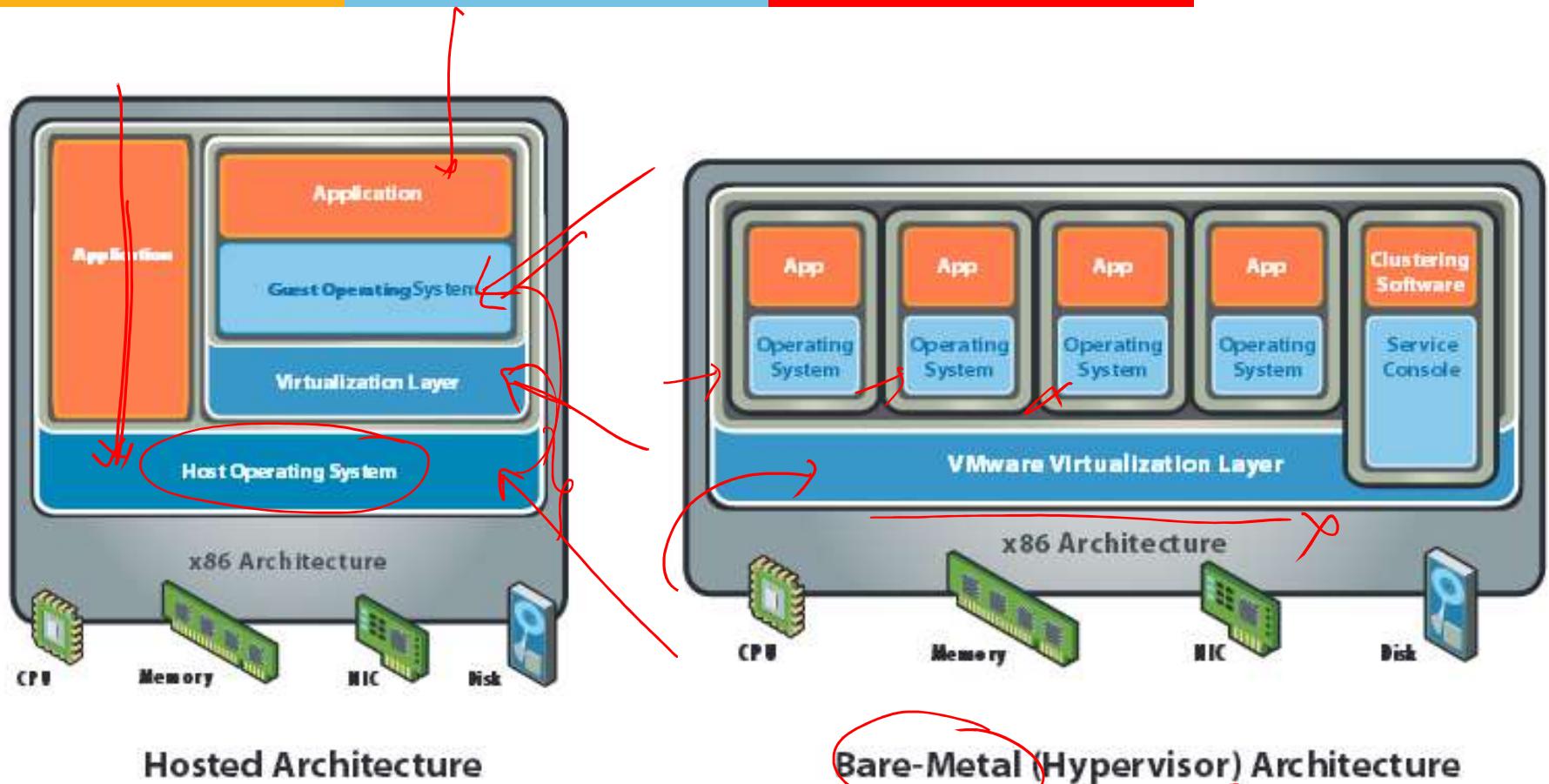
- The latest generation of x86-based systems feature processors with 64-bit extensions supporting very large memory capacities.
- This enhances their ability to host large, memory-intensive applications, as well as allowing many more virtual machines to be hosted by a physical server deployed within a virtual infrastructure.
- The continual decrease in memory costs will further accelerate this trend.



x86 Hardware Virtualization

- For Industry-standard x86 systems, the two approaches typically used with software-based partitioning are
 - hosted and
 - hypervisor architectures
- A hosted approach provides partitioning services on top of a standard operating system and supports the broadest range of hardware configurations.
- In contrast, a hypervisor architecture is the layer of software installed on a clean x86-based system (hence it is often referred to as a “bare metal” approach). Since it has direct access to the hardware resources, a hypervisor is more efficient than hosted architectures, enabling greater scalability, robustness and performance

x86 Hardware Virtualization



Advantages of Virtualization

- Instant provisioning - fast scalability
- Live Migration is possible
- Load balancing and consolidation in a Data Center
is possible.
- Low downtime for maintenance ✓
- Virtual hardware supports legacy operating
systems efficiently
- Security and fault isolation ✓

Terminologies to remember

Basic technique of virtualization is – Trap and Emulate
Virtualization

Privilege levels or protection ring

CPL – Current Privilege Level (Guest OS)

Instruction Types – Behavior sensitive, Control sensitive
Control sensitive (privileged instructions)

Method of virtualization – Binary Translation

Para virtualization (Hypervisor dependent)

Advantages of Virtualization

Security: by compartmentalizing environments with different security requirements in different virtual machines one can select the guest operating system and tools that are more appropriate for each environment. For example, we may want to run the Apache web server on top of a Linux guest operating system and a backend MS SQL server on top of a guest Windows XP operating system, all in the same physical platform. A security attack on one virtual machine does not compromise the others because of their isolation.

Advantages of Virtualization

Reliability and availability: A software failure in a virtual machine does not affect other virtual machines.

Cost: It is possible to achieve cost reductions by consolidation smaller servers into more powerful servers. Cost reductions stem from hardware cost reductions (economies of scale seen in faster servers), operations cost reductions in terms of personnel, floor space, and software licenses. VMware cites overall cost reductions ranging from 29 to 64%

Advantages of Virtualization

Adaptability to Workload Variations: Changes in workload intensity levels can be easily taken care of by shifting resources and priority allocations among virtual machines. Autonomic computing-based resource allocation techniques, such as the ones in can be used to dynamically move processors from one virtual machine to another.

Load Balancing: Since the software state of an entire virtual machine is completely encapsulated by the VMM, it is relatively easy to migrate virtual machines to other platforms in order to improve performance through better load balancing

Advantages of Virtualization

Legacy Applications: Even if an organization decides to migrate to a different operating system, it is possible to continue to run legacy applications on the old OS running as a guest OS within a VM. This reduces the migration cost.

Issues to be aware of

- **Software licensing** ✓
One of the most significant virtualization-related issues to be aware of is software licensing. Virtualization makes it easy to create new servers, but each VM requires its own separate software license. Organizations using expensive licensed applications could end up paying large amounts in license fees if they do not control their server sprawl.
- **IT training** ✓
IT staff used to dealing with physical systems will need a certain amount of training in virtualization. Such training is essential to enable the staff to debug and troubleshoot issues in the virtual environment, to secure and manage VMs, and to effectively plan for capacity.
- **Hardware investment** ✓
Server virtualization is most effective when powerful physical machines are used to host several VMs. This means that organizations that have existing not-so-powerful hardware might still need to make upfront investments in acquiring new physical servers to harvest the benefits of virtualization

Issues to be aware of

- Performance can be a concern, especially for in-band deployments, where the virtualization controller or appliance can become a bandwidth bottleneck.
- Interoperability among vendor products is still evolving.
- Failure of the virtualization device, leading to loss of the mapping table.

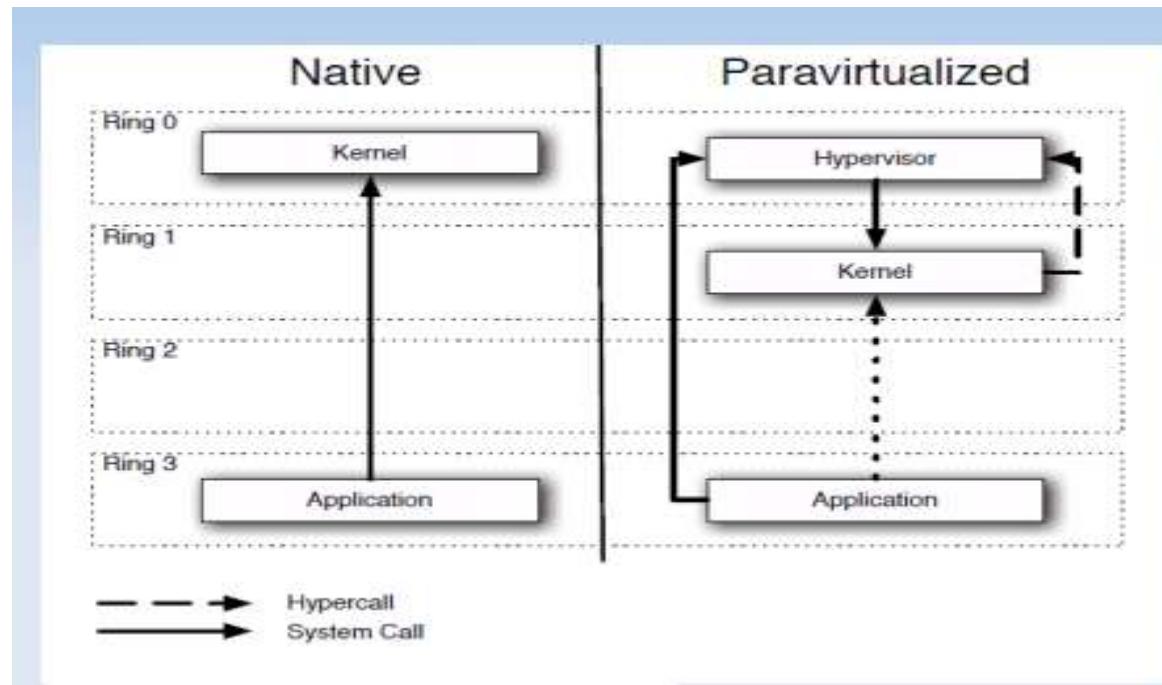
Applications of Virtualization

- Today, virtualization can apply to a range of system layers, including hardware-level virtualization, operating system-level virtualization, and high-level language virtual machines.
- **Maximize resources** — Virtualization can reduce the number of physical systems you need to acquire, and you can get more value out of the servers. Most traditionally built systems are underutilized. Virtualization allows maximum use of the hardware investment.
- **Multiple systems** — With virtualization, you can also run multiple types of applications and even run different OS for those applications on the same physical hardware.
- **IT budget integration** — When you use virtualization, management, administration and all the attendant requirements of managing your own infrastructure remain a direct cost of your IT operation.

Benefits of using Virtual Machines

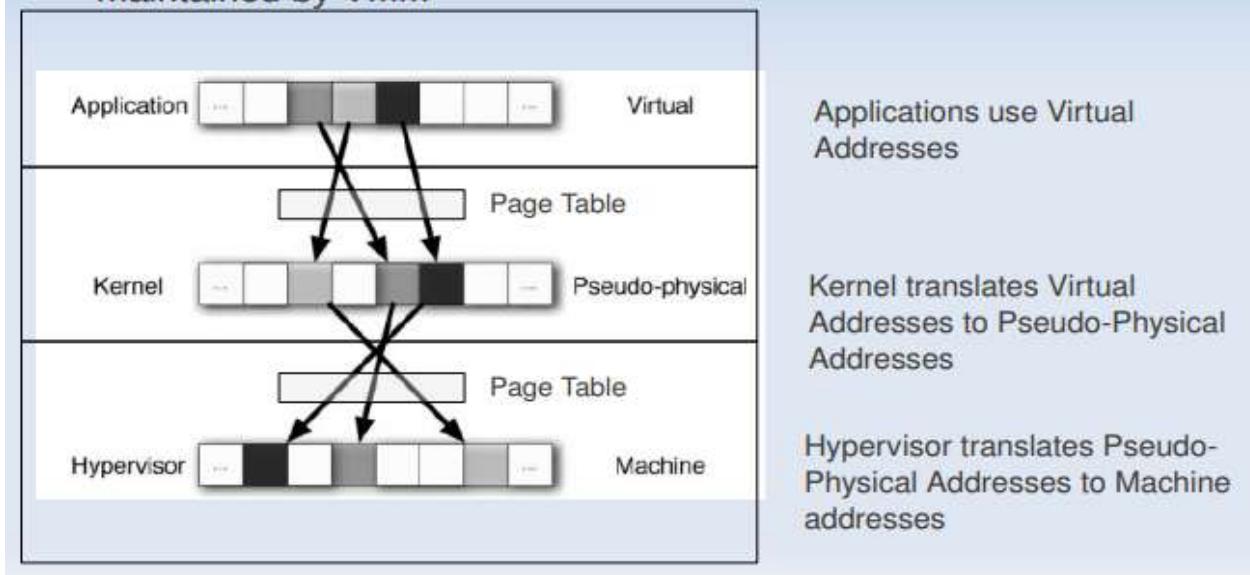
- Instant provisioning - fast scalability
- Live Migration is possible
- Load balancing and consolidation in a Data Center is possible.
- Low downtime for maintenance
- Virtual hardware supports legacy operating systems efficiently
- Security and fault isolation

XEN Hypervisor



Memory Sharing

- In Multiprogramming there is a single level of indirection maintained by Kernel.
- In case of Virtual Machines there is one more level of indirection maintained by VMM

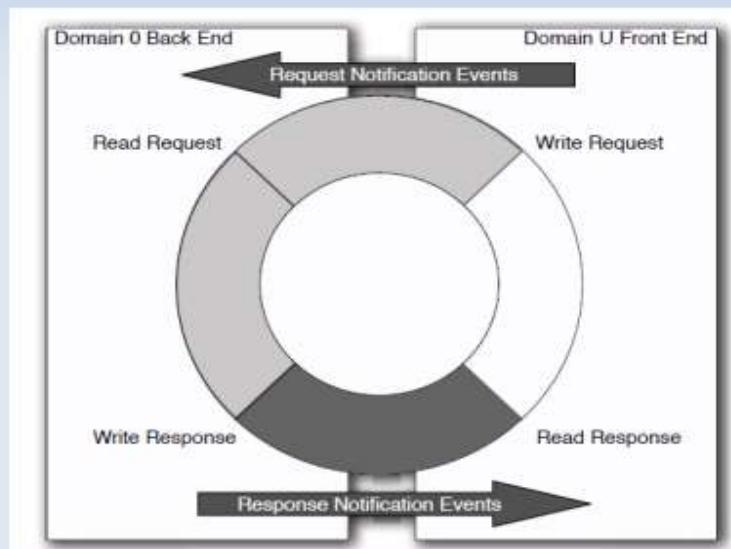


IO Sharing

- DMA Problem
- Device needs to use Physical Memory location.
- In a virtualized environment, the kernel is running in a hypervisor provided virtual address space
- Allowing the guest kernel to convey an arbitrary location to device for writing is a serious security hole
- Detecting a DMA instruction is nontrivial. Each device defines its own protocol for talking to drivers.
- XEN Follows Split Driver Model: Dom 0 does the IO on behalf of all the other guests.
- As DOM0 is privileged the IO has no problem

IO Ring

Shared memory is used with event based synchronization



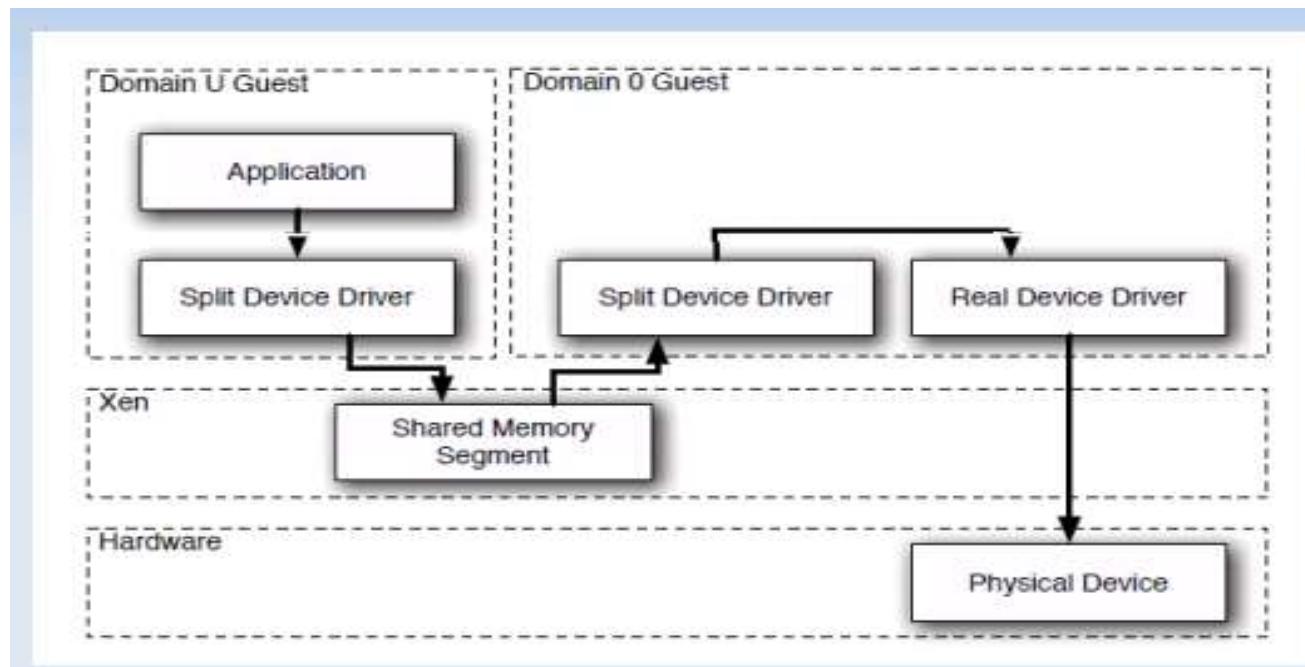
Privilege Rings

- Memory page has a 2 bit code which is checked by CPU before executing the instruction.

If privilege level is insufficient the CPU does not execute the Instruction.



XEN IO Split Device Driver



Conclusions

- Notion of Cloud is possible without Virtualization, but it will be inefficient and inflexible.
- Virtualization is an attempt to manage OS.
- There are many levels and many ways to
- implement Virtualization.

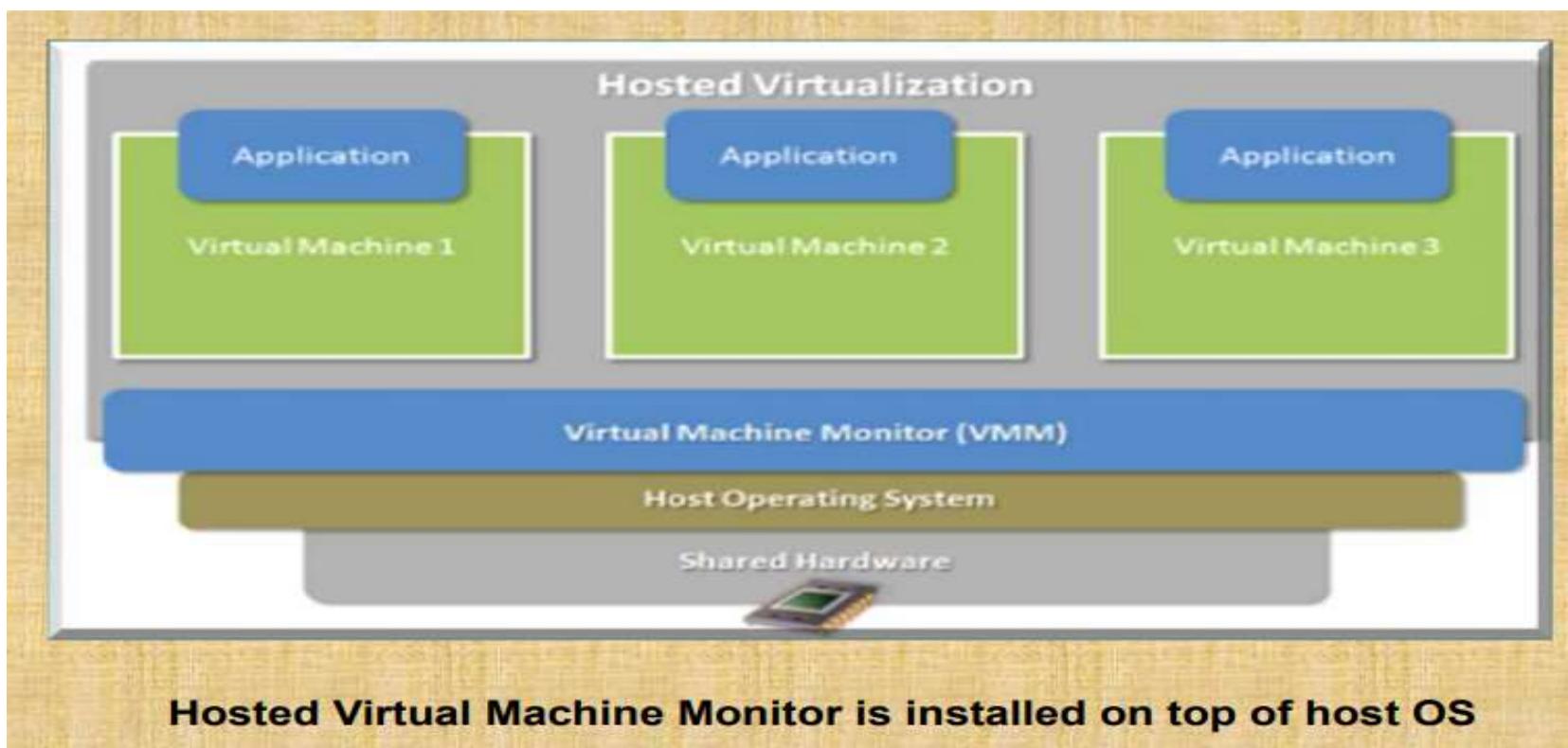
Virtualization Architecture

- Hosted Architecture.
- Bare-Metal Architecture

Hosted Architecture

- In this architecture, host operating system is first installed.
- A piece of software called a hypervisor or virtual machine monitor (VMM) is installed on top of the host OS.
- It allows users to run various guest operating systems within their own application windows.
- Eg. VMware Workstation, Oracle Virtual Box , Microsoft Virtual PC

Hosted Architecture



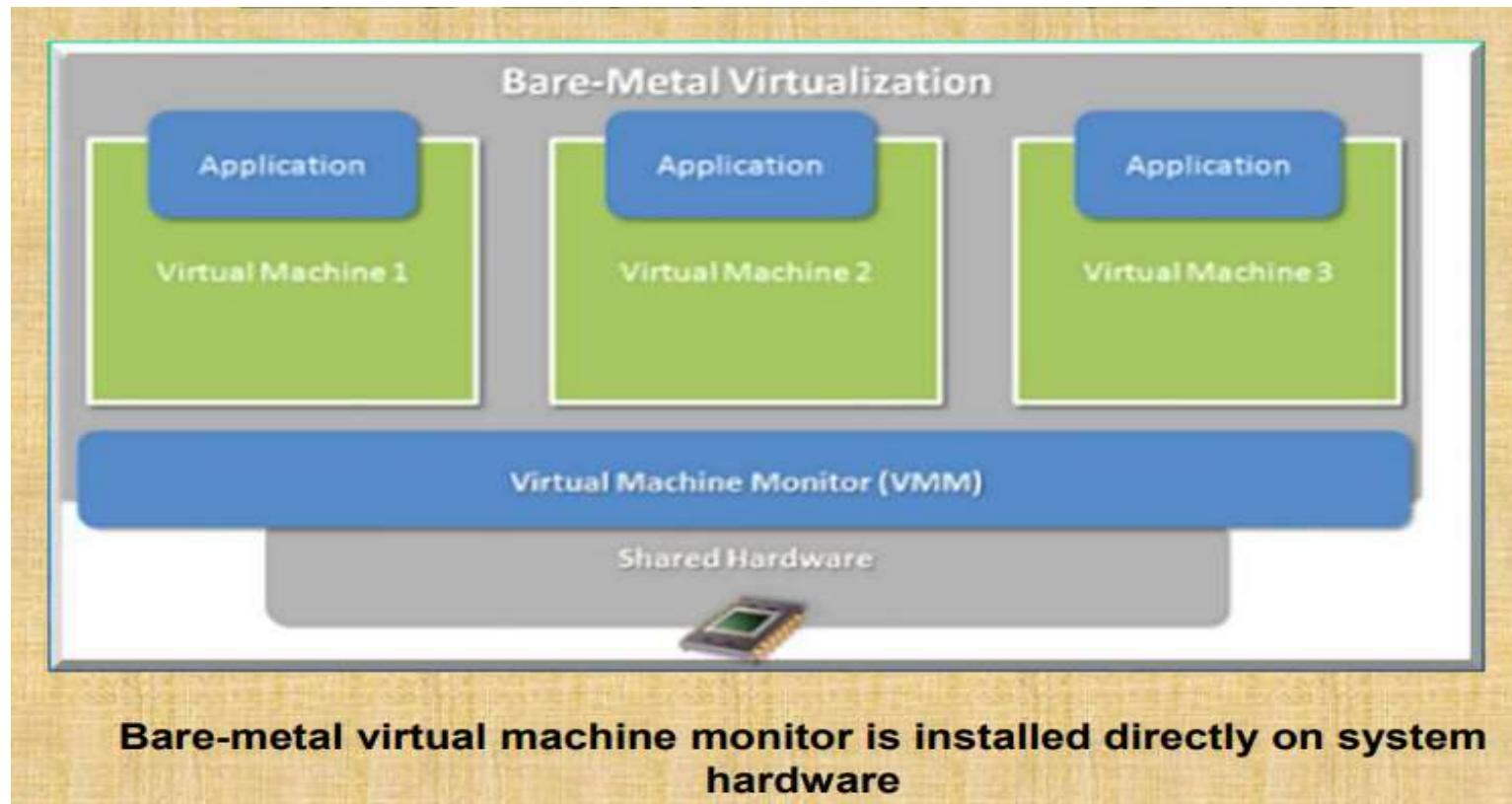
Hosted Architecture(Pros & Cons)

- Advantage
- Ease of installation and configuration.
- Unmodified Host OS & Guest OS.
- Run on a wide variety of pc.
- Disadvantage
- performance degradation.
- lack of support for real-time operating systems.

Bare-Metal Architecture

- In this architecture, type1 hypervisor or VMM is installed on the bare hardware.
VMM communicates directly with system hardware rather than relying on a host operating system.
- E.g: VMWARE ESX, VMWARE ESXi, Microsoft Hyper-V

Bare-Metal Architecture



Bare-Metal Architecture (Pros & Cons)

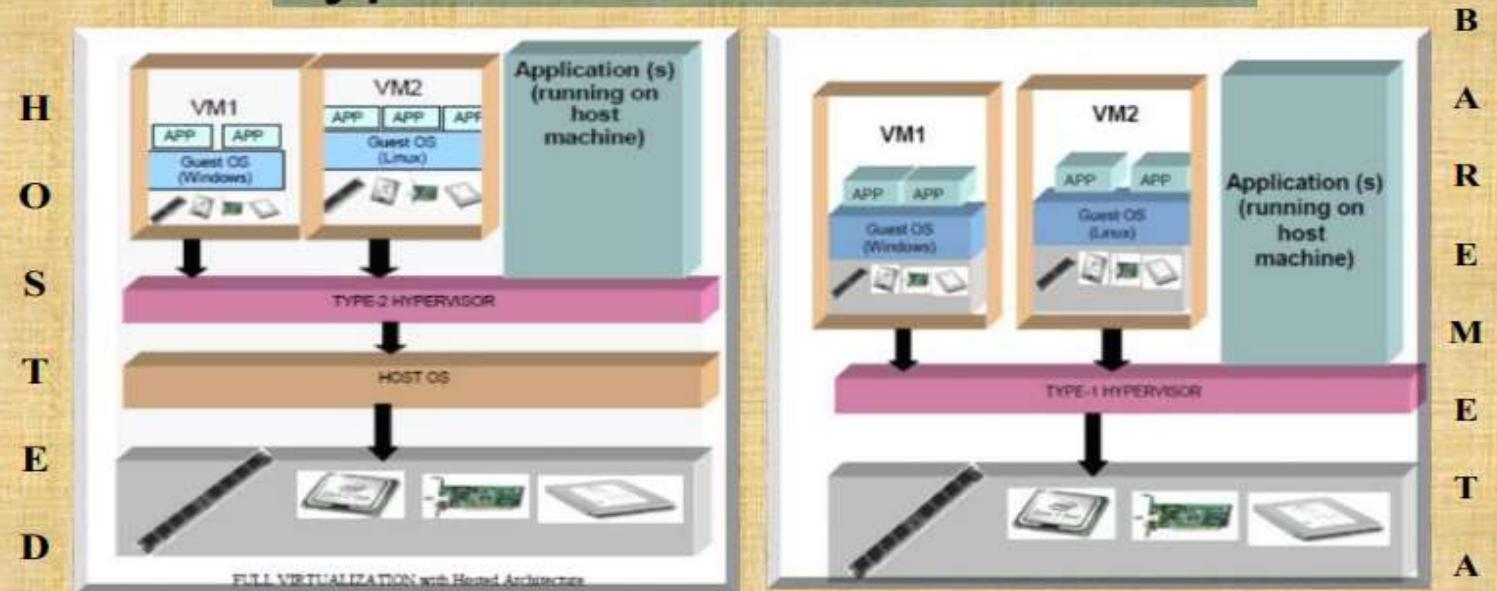
Advantages

- Improved I/O Performance.
- Support Real Time OS.
- Disadvantage
- Difficult to install & Configure.
- Depends upon hardware platform.

FULL VIRTUALIZATION (What is ?)

- It is a virtualization technique used to provide a virtual machine environment which is a complete simulation of the underlying hardware.
- All operating systems and applications which can run natively on the hardware can also be run in the virtual machine.
- The guest OS need not be modified.
- Guest OS do not aware the existence of VM.
- Each VM is independent of each other

Types of Full Virtualization

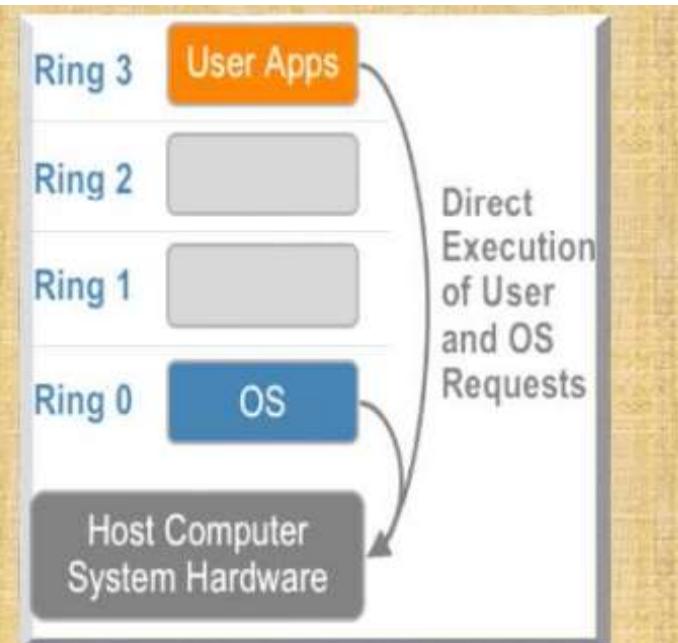


➤ Hypervisor or Virtual Machine Monitor (VMM)

- ✓ SW component that implements virtual machine hardware abstraction.
- ✓ Responsible for hosting and managing virtual machines & running of guest OS.

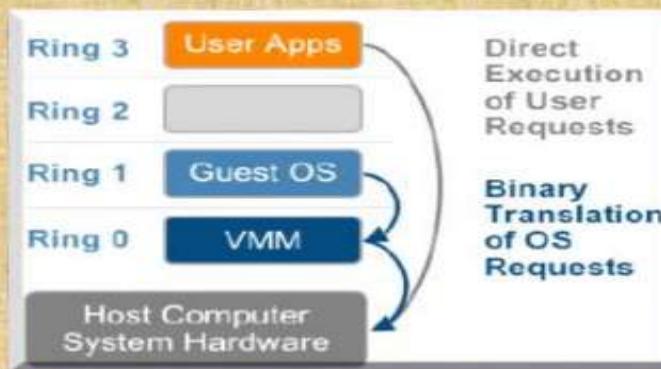
virtualization – Challenges (X86)

- CPU provide 4 protection level(Ring 0 to Ring 3) to OS to execute code.
- OS kernel is designed to run at ring 0 to execute the code directly on the hardware and handle privileged instruction .
- User Application(s) run at ring 3 (less privileged)



So Where Hypervisor resides?

Binary Translation in Full Virtualization



- VMM runs at Ring 0 & Guest OS at Ring 1 (with more privilege than application executing in user space).
- VMM executes
 - **privileged instruction** by dynamically **translating** the instruction of guest OS into a sequence of instruction appropriate to execute in real h/w.
 - It executes the **user level instruction** directly.

Full Virtualization – Advantages & Disadvantage

Advantage

- Secure
- The emulation layer isolates VMs from the host OS & other application (s).
- Total VM portability
- The emulating h/w interface & guest Os forms a standard package that can be ported & run in any platform.
- Run unmodified OS
- Guest OS do not aware of being virtualized.

Disadvantage

- Performance degradation in hosted full virtualization.
- Hardware dependency in bare-metal full virtualization.

Application of Full Virtualization

- ❑ Hosted Full Virtualization is used for Desktop Virtualization.
❑ Eg: Microsoft Virtual PC & Oracle VM Virtual Box.
- ❑ Bare-Metal Full Virtualization is used for Server Virtualization.
❑ Microsoft Hyper-V and VMware ESX Server.
- ❑ Server Virtualization is used in Cloud Computing.

Implementation of Full virtualization (Hosted Architecture)

Platform

- Hardware
- Intel® Core™2 Duo CPU
- 2 GB RAM
- 160 GB HDD
- Software
- Windows 7 as Host OS.
- Windows XP & LINUX as Guest OS.
- Oracle VM Virtual Box.

Implementation of Full virtualization (Hosted Architecture)

- Step1: Install Microsoft Virtual PC (type2 Hypervisor).
- Step2: Create VM1 with winxp (sp2) as guest OS &
- VM2 with Linux as guest OS.
- Step3: Install your desired application on guest OS

Conclusion

- The future of enterprise IT management will be based
- on virtual computing. Intel® VT makes it possible to
- maximize computer utilization while minimizing all
- associated overheads of management, power
- consumption, maintenance and physical space.
- Virtualization also allows the industry to run business
- with fewer machine and at reduced cost while
- providing the infrastructure to meet customer needs.

OS and Virtualization Difference

Host OS uses the actual hardware for the working whereas the Guest OS uses the virtual hardware like number of cores and type and size of hard drive defined by the user while adding a virtual machine. ... Linux operating systems are multi-threaded operating system. The host OS would consider virtual box as a thread.

First of all there aren't any specific number of process for an OS, its called as cores or threads, technically you can define how many cores or threads you want to use on your virtual machine and it depends on the system configuration you use.

Secondly Guest OS is what you have created in the virtual machine and host is what your laptop or pc actually run. Host OS uses the actual hardware for the working whereas the Guest OS uses the virtual hardware like number of cores and type and size of hard drive defined by the user while adding a virtual machine.

Third, as I mentioned earlier Guest and Host OS works on the configurations used by you, if you user higher amount of cores/ threads in setting your virtual machine the Guest OS will get higher speed

Kernel - a program whose purpose is control and multiplexing of hardware for the benefit of other programs.
... Hypervisor - a program whose purpose is control and multiplexing of hardware for other kernels. Typically runs at an even higher privileged level than a kernel, which was invented for this purpose

Comparison of different Server Virtualization software

- Citrix XenServer
- XenServer is an open sourced product from Citrix, based on Xen Project Hypervisor. It's a bare-metal virtualization platform with enterprise-grade features that can easily handle workloads, combined OS, and networking configurations. XenServer delivers application performance for x86 workloads in Intel and AMD environments.
- It can cater to XenApp and XenDesktop deployments, and offer customers the enhanced virtualized graphics with NVIDIA and Intel. XenServer services allow multiple computer operating systems to execute on same computer hardware

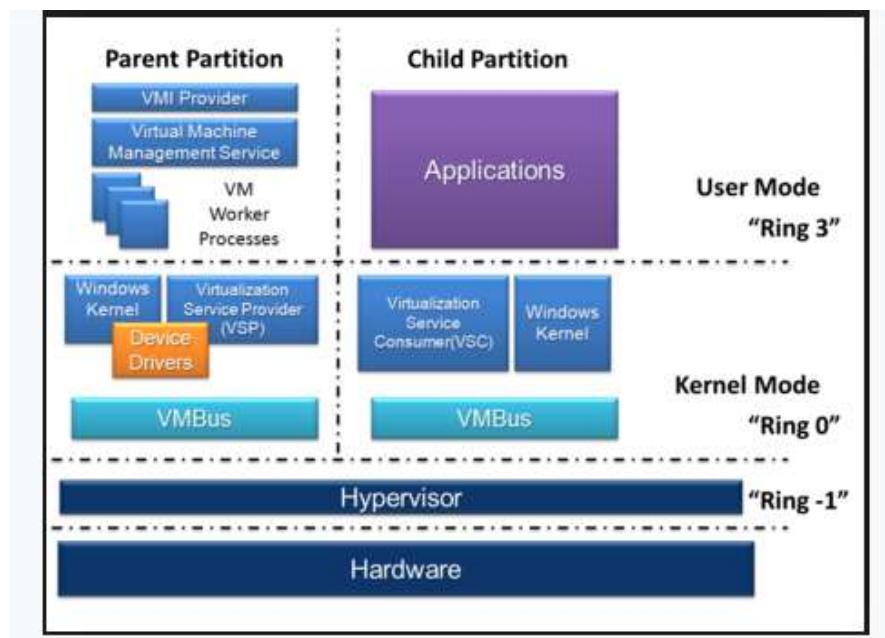
Microsoft Hyper-V

Microsoft introduced its hypervisor in 2008, and has continued to release new versions along with the new Windows servers. Hyper -V helps one expand or establish a private cloud environment, promotes effective hardware utilization, improves business continuity and makes development and testing more efficient. We have discussed some features for Windows Server 2019 here.

Microsoft Hyper V

Features:

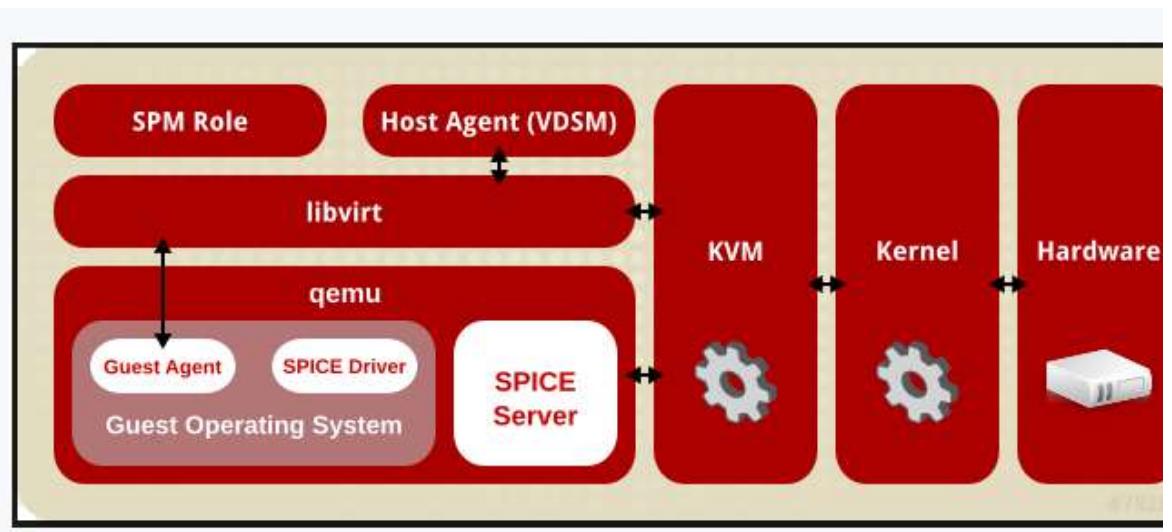
- Persistent memory support.
- Shielded VM updates.
- Simple Two-Node clusters.
- ReFS Deduplication.
- Storage Spaces Direct improvements.
- Windows Admin Center.
- Encrypted subnets.



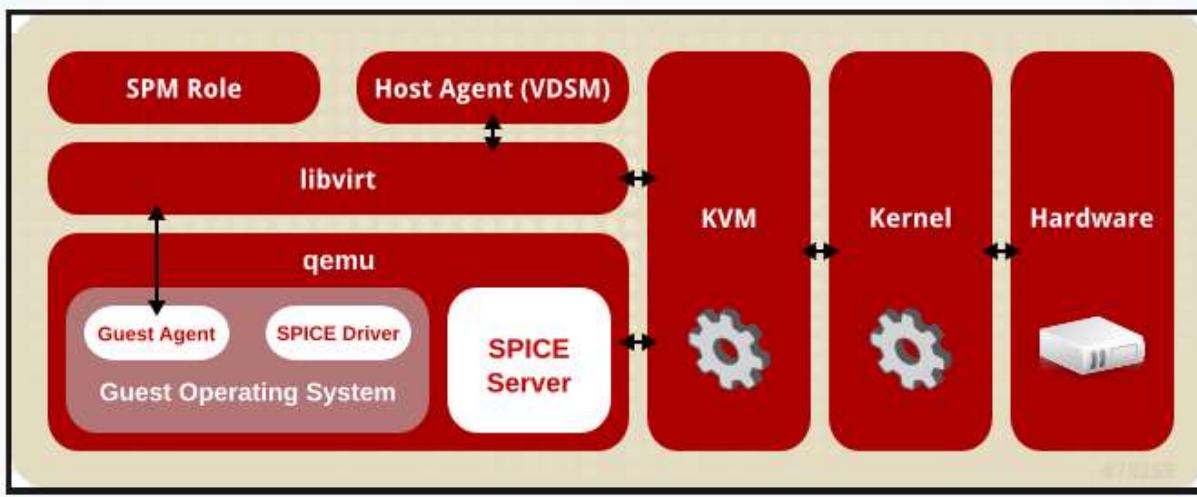
• Red Hat KVM (Kernel-based Virtual Machine)

Red Hat's KVM is a complete virtualization infrastructure solution. Kernel-based Virtual Machine turns Linux kernel into a hypervisor. A part of Red Hat Virtualization suite, it was merged into the Linux kernel mainline in kernel version 2.6.20.

- Here is an overview of the features of KVM:
- Scalability.
- Overcommit resources.
- Disk I/O throttling.
- Hot plug of virtual resources.
- Low cost virtualization solution.
- Red Hat Enterprise Virtualization programming & API.
- Live Migration & Storage Migration.
- Assign any PCI device to virtual machines.
- Container support.
- Disaster Recovery support.
- Red Hat Satellite integration



RedHat KVM



VMware Vsphere

- VMware vSphere
- VMware vSphere is a set of server virtualization products that includes virtualization, management, and interface layers. It comprises following core components- infrastructure services, including VMware vCompute, vStorage and vNetwork; application services; vCenter Server – single point control across datacenter services; and clients who can access the data center via vSphere Client or via a web browser.

- What Is a Bare Metal Server?
- A **bare metal server** is a physical hosting device dedicated to a single client (tenant). Typically set up on-prem or at a third-party **data center** (either rented or via **colocation**), a bare metal server can process more data than any other hosting solution as the user has exclusive use of all computing resources, including:
 - CPU.
 - RAM.
 - Disk space.
 - **Bandwidth**.
- Besides fully dedicated computing resources, other main reasons why companies choose a bare metal server are:
 - High levels of processing power.
 - Consistent input/output operations per second (IOPS).
 - High data privacy due to the lack of other tenants.
 - Complete control over the server's hardware and the software stack.
 - Predictable billing (typically monthly).
 - If your app is sensitive to performance and you wish to store data at a single-tenant device, the benefits of bare metal are hard to beat.

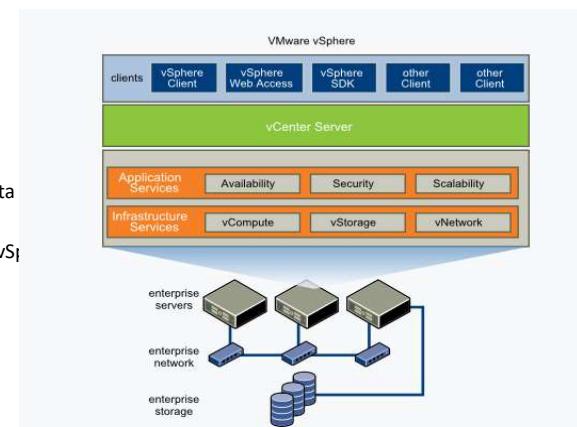
- What Is a Virtual Machine (VM) Server?
- A VM server is a software-based hosting setup that runs in a multi-tenant environment within a single device. Splitting a machine into individual VMs requires [server virtualization](#), a process that enables a device to host multiple systems while sharing the same physical resources (disk space, RAM, and CPU).
- Data centers create shared-resource servers using a hypervisor. A hypervisor parcels the server into distinct units that rely on the same components but have separate OSs, apps, and networking. To learn more about how hypervisors and [virtualization](#) work, refer to our article on [types of hypervisors](#).
- Most companies that choose a VM server over other hosting solutions do so because of the following reasons:
 - Quick and simple deployments.
 - Flexibility to add more server resources in times of high usage and match fluctuating traffic demands.
 - Ability to control and optimize costs through a pay-as-you-go model.
 - Little to no hardware-related [server management](#).
 - Quick and reliable [snapshots and backups](#).
 - VM servers are ideal for dynamic workloads and non-mission-critical apps that prioritize flexibility over consistently high performance

WHY HOST ON A BARE METAL SERVER?

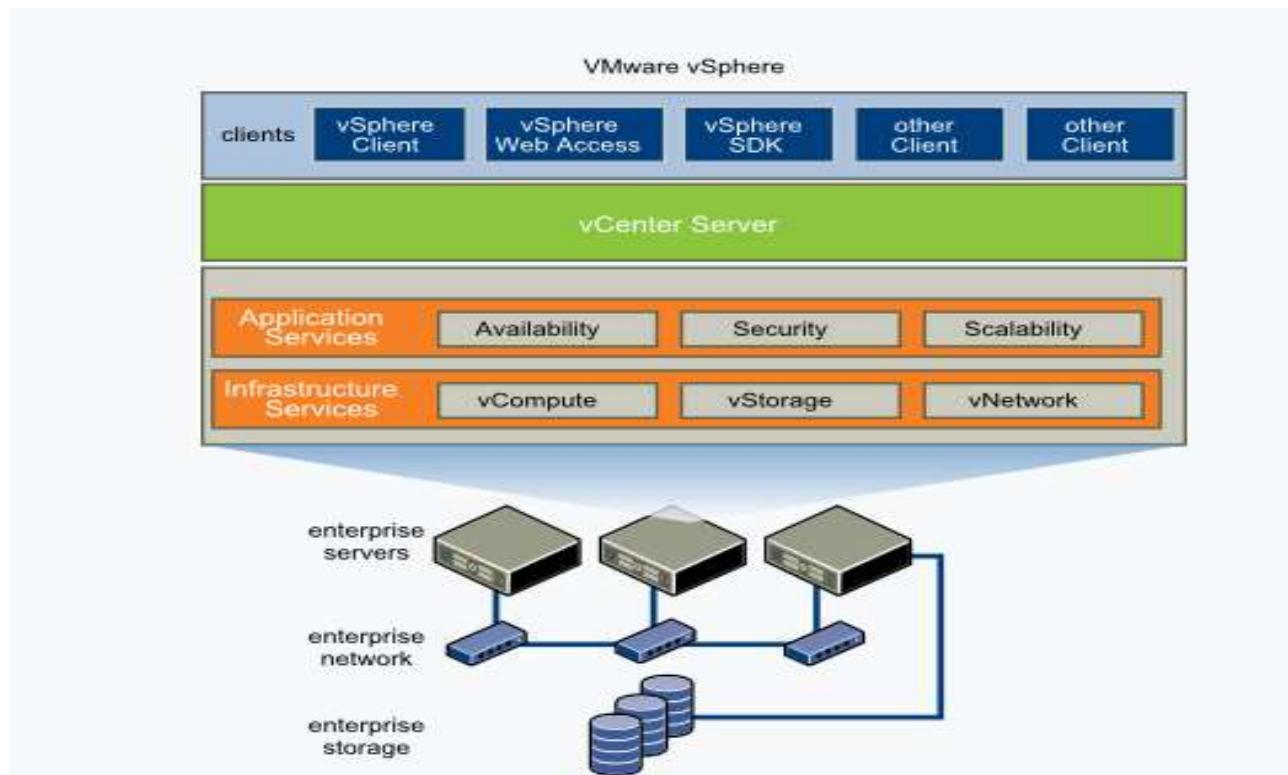
- No resource sharing with other server tenants
- Top, consistent performance
- High levels of security and data privacy
- The ability to set up hardware from scratch
- Complete freedom over the software stack



- Features and components:
- It abstracts memory, processors, storage and other resources into multiple VMs.
- vCenter Server: Centralized management tool to configure, provision and manage virtual IT environments. Provides data hosts.
- vSphere Client: vSphere 6.7 has the final version of Flash-based vSphere Web Client. Newer workflows in the updated vSphere library, vSAN, Storage policies, Host profiles, VMware vSphere Distributed Switch™ topology diagram and Licensing.
- vSphere SDKs: Provides interfaces for third-party solutions to access vSphere.
- VM File System: A cluster file system for VMs.
- Virtual SMP: Enables a single VM to use multiple physical processors at a time.
- vMotion: Enables live migration with transaction integrity.
- Storage vMotion: Enables VM file migration from one place to other without service interruption.
- High Availability: If one server fails, VM is shifted to another server with spare capacity to enable business continuity.
- Distributed Resource Scheduler (DRS): Assigns and balances compute automatically across hardware resources available for VMs.
- Fault Tolerance: Generates copy of primary VM to ensure its continuous availability.
- Distributed Switch (VDS): Spans multiple ESXi hosts and enables considerable reduction of network maintenance activities and increases network capacity.
- Network & Storage I/O Control.
- Hot add CPU and RAM resources



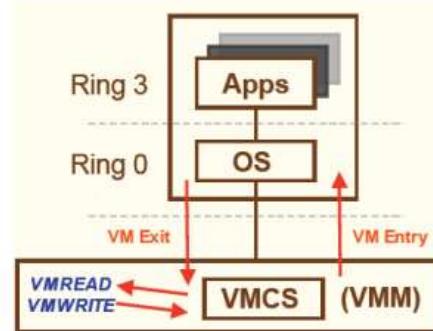
VMware vSphere



Hypervisor Comparison 2019: KVM vs Hyper-V vs XenServer vs vSphere

Feature	Windows Hyper-V 2019	vSphere 6.7	XenServer 7.6	KVM
RAM/Host	24TB	12 TB	5TB	12TB
RAM/VM	12 TB for generation 2; 1 TB for generation 1	6 TB	1.5TB	6 TB
CPU/VM	240 for generation 2; 64 for generation 1;	128	32	240
VM Disk	64 TB for VHDX format; 2040 GB for VHD format	62TB	2TB	10TB
VM Live Migration	Yes	Yes	Yes	Yes
VM Replication supports	Yes	Yes	Yes	Yes
Overcommit resources	No	Yes	No	Yes
Disk I/O Throttling	Yes	Yes	Yes	Yes
Hot plug of virtual resources	Yes	Yes	Yes	Yes

VM Control Structure



The VM Control Structure (VMCS)

VM-execution controls	Determines what operations cause VM exits	CR0, CR3, CR4, Exceptions, IO Ports, Interrupts, Pin Events, etc.
Guest -state area	Saved on VM exits Reloaded on VM entry	EIP, ESP, EFLAGS, IDTR, Segment Regs, Exit info, etc.
Host -state area	Loaded on VM exits	CR3, EIP set to monitor entry point, EFLAGS hardcoded, etc.
VM-exit controls	Determines which state to save, load, how to transition	Example: MSR save -load list
VM-entry controls	Determines which state to load, how to transition	Including injecting events (interrupts, exceptions) on entry

Virtualization Benefits

REDUCE ENERGY COSTS AND GO GREEN WITH VMWARE VIRTUALIZATION

Reduce the energy demands of your datacenter by dynamic management of computer capacity across a pool of servers.

VMware infrastructure delivers the resources your infrastructure needs and enables you to:

- Reduce energy costs by 80%.**
- Power down servers without affecting applications or users.**
- Green your datacenter while decreasing costs and improving service levels.**

What Is a Bare Metal Server?

A **bare metal server** is a physical hosting device dedicated to a single client (tenant). Typically set up on-prem or at a third-party **data center** (either rented or via **colocation**), a bare metal server can process more data than any other hosting solution as the user has exclusive use of all computing resources, including:

- CPU.
- RAM.
- Disk space.
- Bandwidth.**

Besides fully dedicated computing resources, other main reasons why companies choose a bare metal server are:

- High levels of processing power.
- Consistent input/output operations per second (IOPS).
- High data privacy due to the lack of other tenants.
- Complete control over the server's hardware and the software stack.
- Predictable billing (typically monthly).

WHY HOST ON A BARE METAL SERVER?

- No resource sharing with other server tenants
- Top, consistent performance
- High levels of security and data privacy
- The ability to set up hardware from scratch
- Complete freedom over the software stack



What Is a Virtual Machine (VM) Server?

A VM server is a software-based hosting setup that runs in a multi-tenant environment within a single device. Splitting a machine into individual VMs requires [server virtualization](#), a process that enables a device to host multiple systems while sharing the same physical resources (disk space, RAM, and CPU). Data centers create shared-resource servers using a hypervisor. A hypervisor parcels the server into distinct units that rely on the same components but have separate OSs, apps, and networking. To learn more about how hypervisors and [virtualization](#) work, refer to our article on [types of hypervisors](#). Most companies that choose a VM server over other hosting solutions do so because of the following reasons:

- Quick and simple deployments.
- Flexibility to add more server resources in times of high usage and match fluctuating traffic demands.
- Ability to control and optimize costs through a pay-as-you-go model.
- Little to no hardware-related [server management](#).
- Quick and reliable [snapshots and backups](#).

VM servers are ideal for dynamic workloads and non-mission-critical apps that prioritize flexibility over consistently high performance.

WHY HOST ON A VIRTUAL SERVER?



- Near-instant server and component deployment
- Ideal for dynamic, unpredictable workloads
- On-demand scalability
- Pay only for resources you use
- Quick and reliable disaster recovery

Bare Metal Vs VM Servers (Comparison Table)

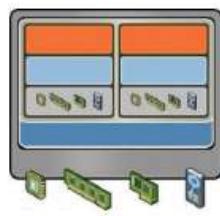
The table below offers a high-level overview of the main differences between bare metal and VM servers:

POINT OF COMPARISON	BARE METAL SERVER	VM SERVER
Main selling points	Consistent performance and complete data privacy	Near-instant scalability and cost optimization options
Hardware dedication	All server resources (CPU, RAM, memory, bandwidth) belong to a single user	Tenants host on the same device and share server resources
Performance capabilities	Consistently high performance	Less consistent performance due to multiple tenants
Customization options	The tenant has complete freedom when setting up both hardware and software	Fewer software customization options due to the shared nature of the server
Deployment time	Takes time to set up a new server (hours for a rented device, days for an on-prem server)	New deployments are a matter of a few minutes
Hardware maintenance	Complex without managed services	No hardware maintenance
Scalability	Scaling up or down requires months for on-prem servers, hours for rented devices	Near-instant, on-demand scalability (both up and down)
Capacity optimization	Limited capacity optimization	Advanced capacity optimization enabled by load balancing
Security	Customization options and single tenancy ensure a secure IT platform	Other tenants can cause security and privacy concerns
System recovery	Recovering from a mid-size disaster can take hours or even days	Recovering from a disaster happens in minutes
Server portability	Moving a physical server is a complex and lengthy task	You can easily move a VM across virtual environments or from one physical server to another
Typical billing methods (for a rented server)	A predictable (typically monthly) bill	Charges based on how much resources you use
On-prem expenses	High upfront costs for hardware and space <small>(but no need for licensing purchase)</small>	Smaller hardware costs but pricey VM software licenses

VMware Server Virtualization

Features

KEY FEATURES OF THE VMWARE SERVER VIRTUALIZATION



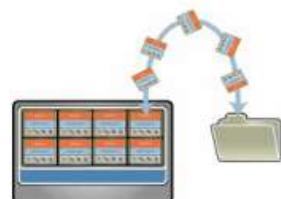
Partitioning

- Different OS can run on one physical machine
- System resources can be divided between virtual machines



Isolation

- Fault and security isolation on a hardware level
- Extended resource control for constant performance

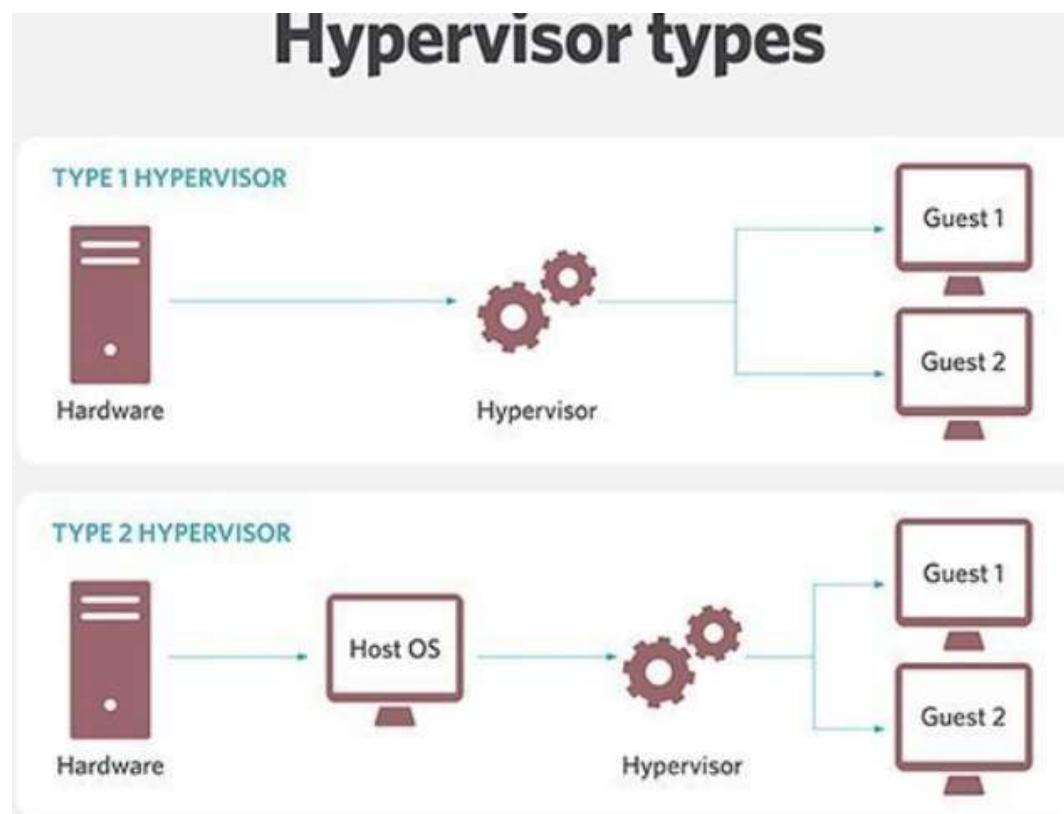


Encapsulation

- Complete status of a virtual machine can be stored in a file
- Move and copy of a virtual machine is as easy as it is with files



Hypervisor types





BITS Pilani
Pilani Campus

BITS Pilani presentation

Mridul Moitra
Cloud Computing

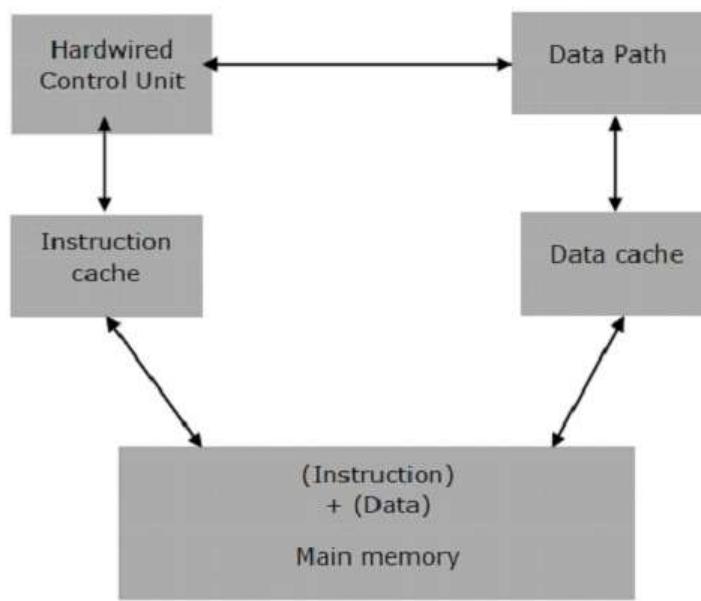


BITS Pilani

<CSI ZG527 / SS ZG527 / SE ZG527
Cloud Computing
Lecture No. 3

Difference between RISC and CISC –

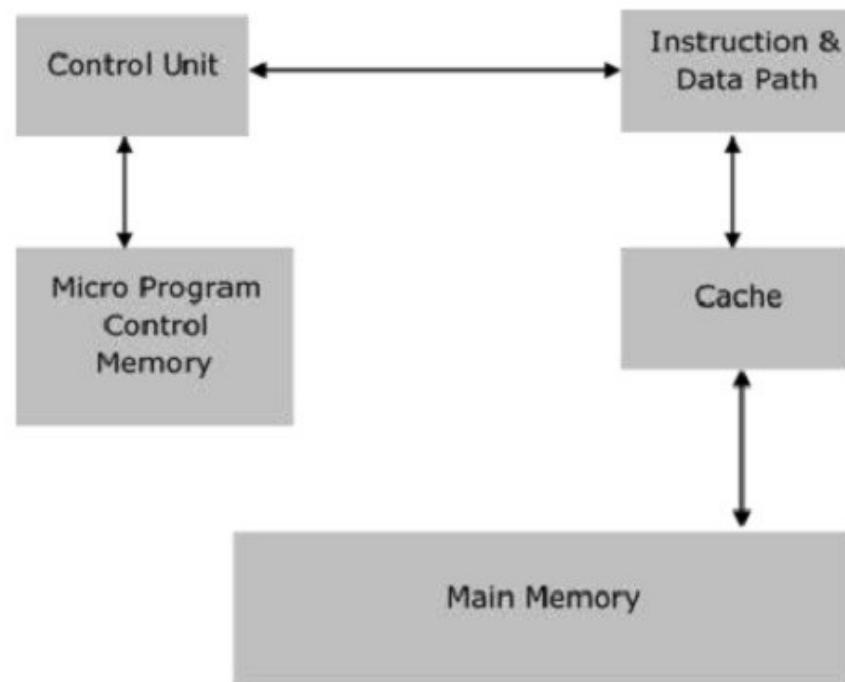
- Microprocessors are classified based on the architecture (instruction set) into RISC and CISC.
- RISC
- It stands for Reduced Instruction Set Computer.
- It is a microprocessor architecture that uses small instruction set of uniform length.
- These simple instructions are executed in one clock cycle.
- These chips are relatively simple to design.
- They are inexpensive.
- The disadvantage is that the computer has to repeatedly perform simple operations in order to execute a large program that has a large number of processing operations.
- Examples of RISC chips include SPARC, POWER PC.
- It has less number of instructions.
- It has fixed-length encodings for instructions.
- Simple addressing formats are supported.
- It doesn't support arrays.
- It doesn't use condition codes.
- Registers are used for procedure arguments and return addresses.
-



CISC

- It stands for Complex Instruction Set Computer.
- This offers hundreds of instructions of different sizes to the users.
- This architecture has a set of special purpose circuits which help execute the instructions at a high speed.
- The instructions interact with memory using complex addressing modes.
- These processors reduce the size of the program.
- Consequently, they take less number of memory cycles to execute the programs.
- The overall speed of execution is high.
- Examples of CISC include Intel architecture, AMD.
- It has variable-length encodings of instructions.
- It supports array.
- It has a large number of instructions.
- Arithmetic and logical operations can be applied to memory and register operands.

Architecture of CISC



Key Differences:

1. RISC machine focuses more on software and less on hardware, whereas; CISC machine focuses more on hardware and less on software.
2. RISC machine has greater use of registers so, they use transistors for more registers, whereas; CISC machine uses a greater number of complex instructions, so they use transistors to store all their complex instructions.
3. In RISC machine, as it follows a software-based approach that is why, the code part is large, whereas; in CISC machine, as it is complex hardware driven, this makes the code part much smaller.
4. In RISC machine, due to its great and reliable software approach, an instruction can execute in a single clock cycle, whereas; in CISC machine, due to its more hardware driven approach, an instruction lags a little bit and takes more than one clock cycle.
5. The RISC instructions are quite handy and easy as they can fit in a single word, whereas; the CISC instructions are quite larger than a typical word.

BASIS	RISC	CISC
Stands for	Reduced Instruction Set Computer	Complex Instruction Set Computer
Size of instructions	Smaller and simpler instructions	Larger and Complex instructions
Execution Time	1 cycle per instruction	Multiple number of cycles per instruction
Emphasis	On Software	On hardware
Instruction Formats	Fixed (4 bytes)	Variable Length (2-6 bytes)
Control Unit	Hardwired Control Unit	Microprogrammed Control Unit
Data and Instruction Cache	Separate	Combined
Example of processors	ARM processor and Qualcomm processor are some examples	AMD, VAX and Intel x86 CPUs are some examples
CPU size	Smaller	Larger as they have larger instruction libraries

Addressing Modes	Fewer	Many
Array	Not Supported	Supported
Pipeline	Easy	Hard
Power consumption	Less	More
Referred as	Machine Oriented	Programmer Oriented
Register sets	Has multiple register sets	Has Single register set
Memory unit	Doesn't have memory unit	Have memory unit

Cloud Security Architecture Patterns

Cloud Security Architecture Patterns

The right pattern can help you implement security across your organization. For example, it can help you protect the CIA (confidentiality, integrity, and availability) of your cloud data assets, as well as respond to security threats. You can implement security controls directly, or use security controls as a service offered by your cloud provider or third-party vendors.

The cloud security architecture model is usually expressed in terms of:

- **Security controls**—which can include technologies and processes. Controls should take into account the location of each service—company, cloud provider, or third party.
 - **Trust boundaries**—between the different services and components deployed on the cloud
 - **Standard interfaces and security protocols**—such as SSL, IPSEC, SFTP, LDAPS, SSH, SCP, SAML, OAuth, etc.)
 - **Techniques used for token management**—authentication, and authorization
 - **Encryption methods** including algorithms like 128-bit AES, Triple DES, RSA, Blowfish.
 - **Security event logging**—ensuring all relevant security events are captured, prioritized, and delivered to security teams.
- Here are a few best practices that you can follow to enhance the security of your cloud-based assets:

Enforce policies and data governance:

It is entirely the enterprise's responsibility to put in place and enforce policies for cloud data ownership and responsibility. At the most basic level, the enterprise must understand and classify its data so that the appropriate security measures can be implemented according to the varying levels of data sensitivity.

Diligently manage identity and access controls:

Identity and access management (IAM) in the cloud is substantially more complex than it is in closed, monolithic environments. Cloud providers offer best practice guidelines as well as tools and managed services to help organizations handle IAM, but it's up to the organization to use them effectively.

Security Control

Each security control should be clearly defined using the following attributes:

- **Service function**—what is the service's role? For example, encryption, authorization, event data collection.
- **Logical location**—public cloud service, third party service, or on-premises. Location affects performance, availability, firewall policies, and service management.
- **Protocol**—what protocol is used to access the service? For example, REST, HTTPS, SSH.
- **Input/Output** – what does the service receive and what is it expected to deliver? For example, input is a JSON feed and output is the same feed with encrypted payload data.
- **Control mechanisms**—what types of control does the service achieve? For example, data at rest

IaaS Cloud Computing Security Architecture

IaaS Cloud Computing Security Architecture

In an IaaS framework, the cloud provider is completely responsible for the physical resources and shares responsibility with the customer for the security of the host infrastructure and network; all the rest is the responsibility of the customer. This level brings the customer the most freedom, but also places the majority of the responsibility in their hands. The principles of IaaS closely follow the shared responsibility model for providers like AWS and Azure.

IaaS provides storage and network resources in the cloud. It relies heavily on APIs to help manage and operate the cloud. However, cloud APIs are often not secure, because they are open and easily accessible from the web.

The cloud service provider (CSP) is responsible for securing the infrastructure and abstraction layer used to access the resources. Your organization's security obligations cover the rest of the layers, mainly containing the business applications.

To better visualize cloud network security issues, deploy a Network Packet Broker (NPB) in an IaaS environment. The NPB sends traffic and data to a Network Performance Management (NPM) system, and to the relevant security tools. In addition, establish logging of events occurring on network endpoints.

IaaS cloud deployments require the following additional security features:

- Network segmentation
- Intrusion Detection System and Intrusion Prevention System (IDS/IPS)
- Virtual firewalls placed in front of web applications to protect against malicious code, and at the edge of the cloud network
- Virtual routers

PaaS Cloud Computing Security Architecture

PaaS Cloud Computing Security Architecture

In a PaaS framework, the provider also takes full responsibility for hosting physical infrastructure and network security, but it also shares responsibility with the customer at the application and access control levels. Application software, virtual machines and instances, and services such as AWS Elastic Beanstalk and AWS Lambda typically fall into the Platform-as-a-Service category.

PaaS platforms enable organizations to build applications without the overhead and complexity associated with managing hardware and back-end software. In a PaaS model, the CSP protects most of the environment. However, the company is still responsible for the security of the applications it is developing.

Therefore a PaaS security architecture is similar to a

SaaS Cloud Computing Security Architecture

SaaS Cloud Computing Security Architecture

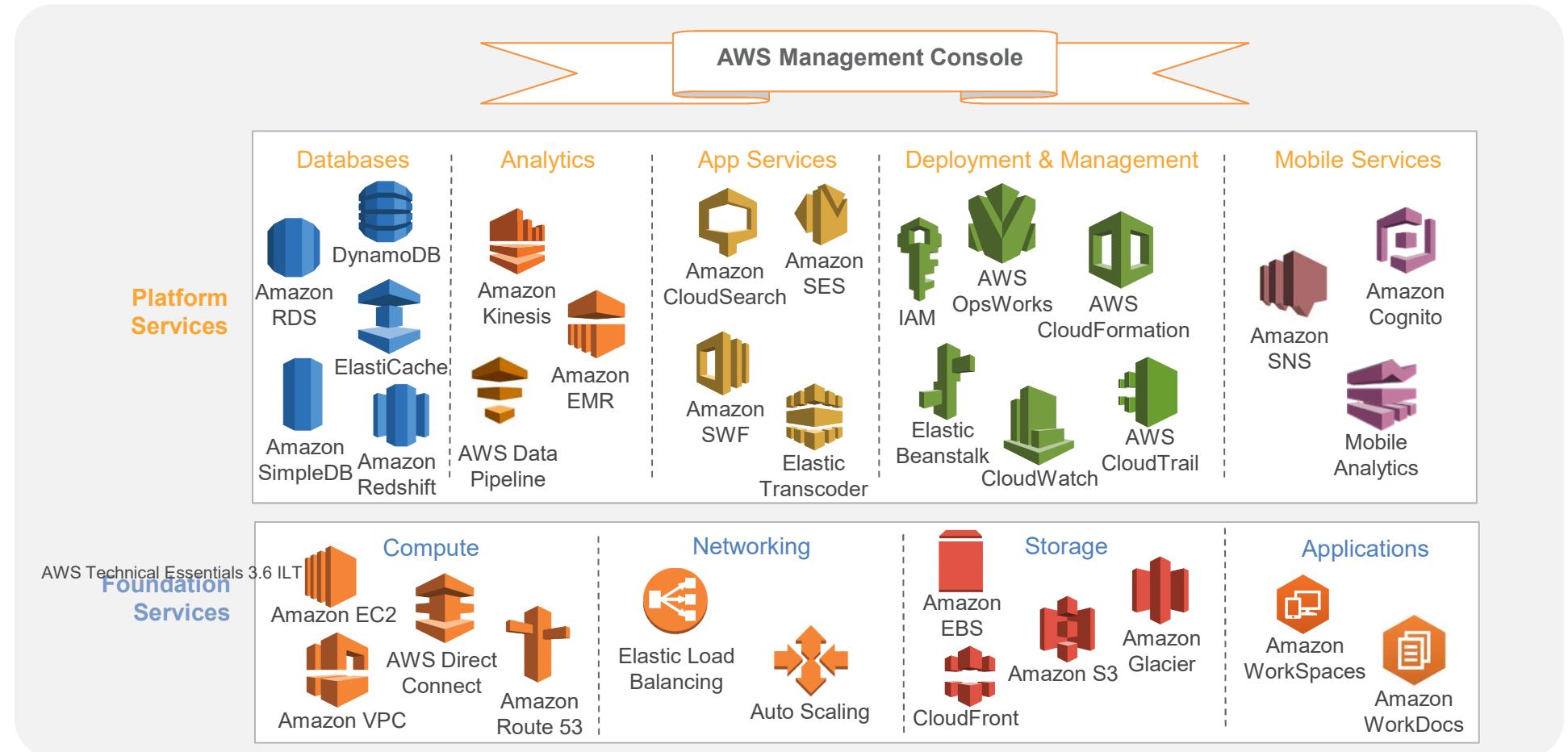
SaaS framework, the provider takes full responsibility for application controls while sharing responsibility with the customer for access control as well as client/endpoint protection.

Generally, SaaS companies provide business applications or other consumer apps over the internet that may run in the cloud, SaaS services provide access to software applications and data through a browser. The specific terms of security responsibility may vary between services, and are sometimes up for negotiation with the service provider.

Cloud Access Security Brokers (CASB) offers logging, auditing, access control and encryption capabilities that can be critical when investigating security issues in a SaaS product. In addition, make sure your SaaS environment has:

- Logging and alerting
- IP whitelists and/or blacklists
- API gateways, in case the service is accessed via API

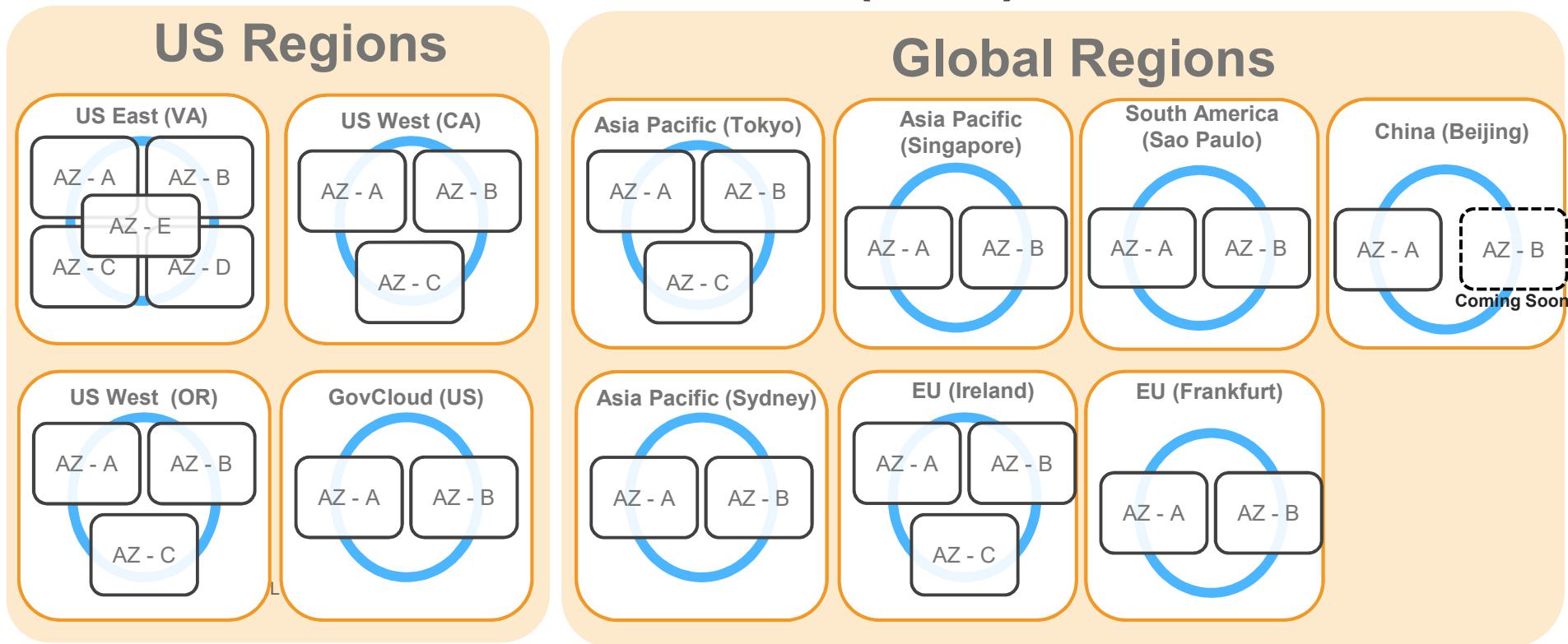
AWS Services



Regions and Edge Locations

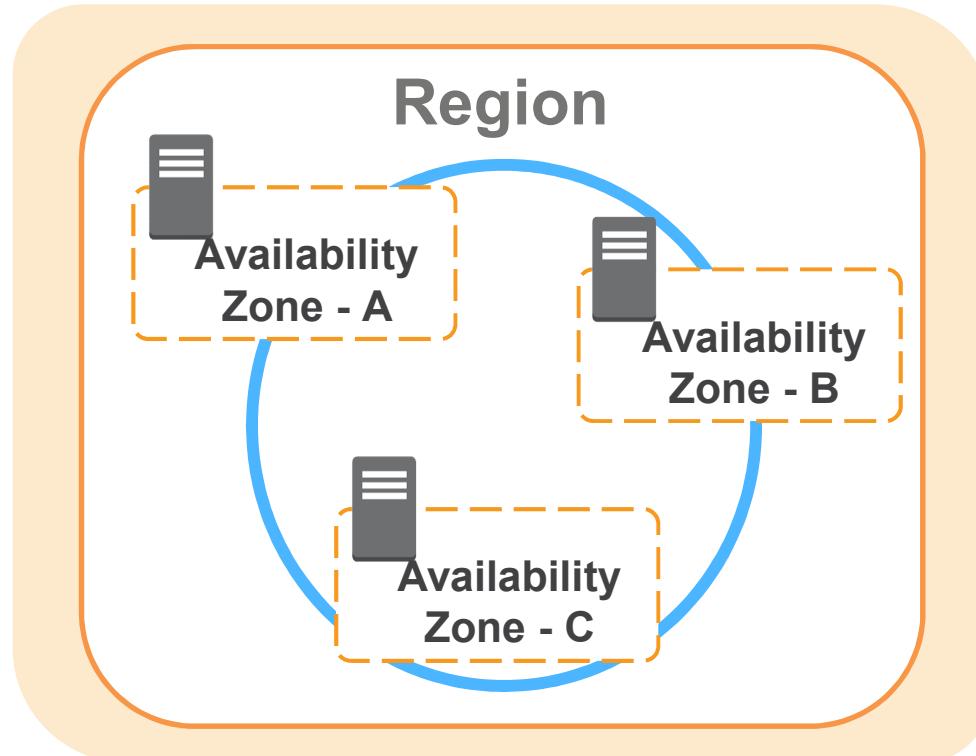


AWS Regions and Availability Zones (AZ)



Note: Conceptual drawing only. The number of Availability Zones (AZ) may vary.

Achieving High Availability Using Multi-AZ



Shared Security Responsibility—AWS

Customer

AWS

AWS Technical Essentials 3

Customer Data

Platform, Applications, Identity & Access Management

Operating System, Network & Firewall Configuration

Client-side Data Encryption &
Data Integrity Authentication

Server-side Encryption
(File System and/or Data)

Network Traffic Protection
(Encryption/Integrity/Identity)

Foundation Services

Compute

Storage

Database

Network

**AWS Global
Infrastructure**

Availability Zones

Regions

**Edge
Locations**

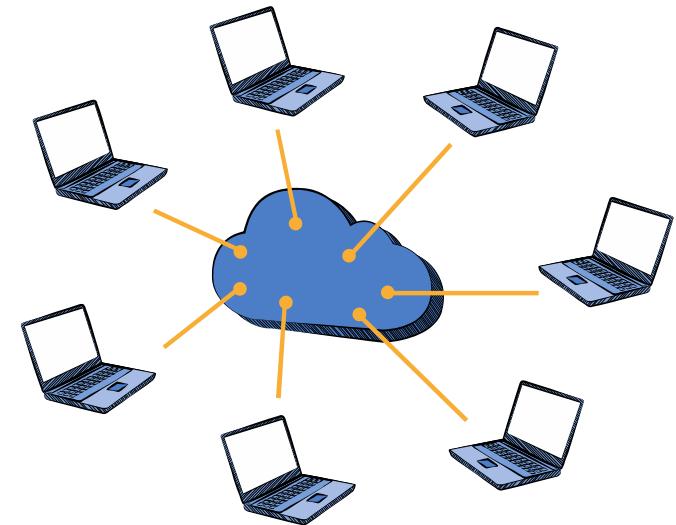
Physical Security

- 24/7 trained security guards
- Locations in nondescript, undisclosed facilities
- Two-factor authentication for ingress
- Authorization for datacenter access



Hardware, Software, and Network

- Automated change-control process
- Bastion servers that record all access attempts
- Firewall and other boundary devices
- AWS monitoring tools



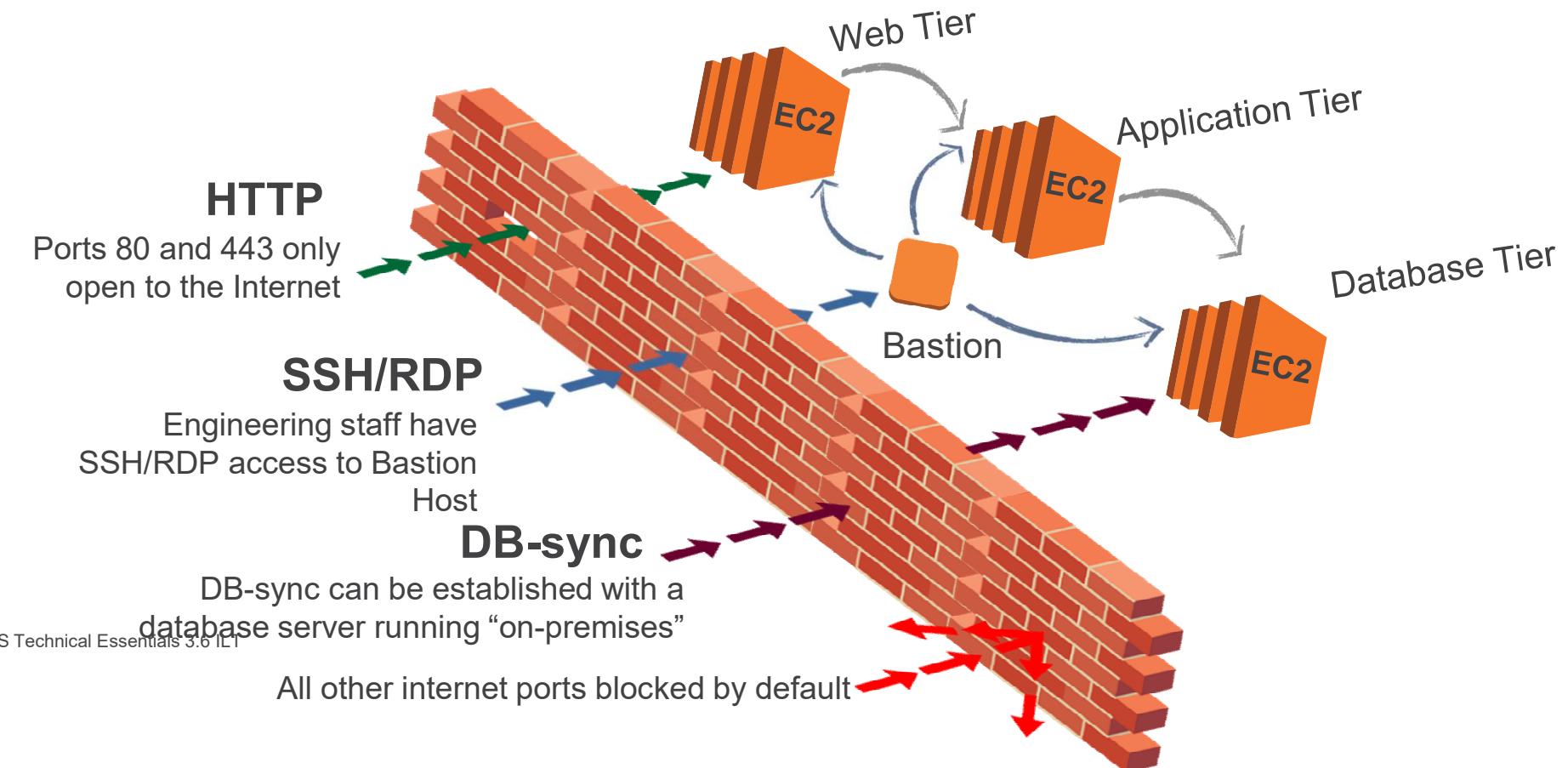
SSL Endpoints

SSL Endpoints	Security Groups	IAM	VPC
<p>Secure Transmission</p> <p>Establish secure communication sessions (HTTPS) using SSL</p>	<p>Instance Firewalls</p> <p>Configure firewall rules for instances using Security Groups</p>	<p>User Accounts</p> <p>Create individual IAM accounts so that each of your users has their own AWS security credentials</p>	<p>Subnet Control</p> <p>In your Virtual Private Cloud, create low-level networking constraints for resource access, such as public and private subnets, internet gateways, and NATs</p>

Security Groups

SSL Endpoints	Security Groups	IAM	VPC
<p>Secure Transmission</p> <p>Establish secure communication sessions (HTTPS) using SSL</p>	<p>Instance Firewalls</p> <p>Configure firewall rules for instances using Security Groups</p>	<p>User Accounts</p> <p>Create individual IAM accounts so that each of your users has their own AWS security credentials</p>	<p>Subnet Control</p> <p>In your Virtual Private Cloud, create low-level networking constraints for resource access, such as public and private subnets, internet gateways, and NATs</p>

AWS Multi-tier Security Groups



AWS Identity and Access Management (IAM)

SSL Endpoints	Security Groups	IAM	VPC
<p>Secure Transmission</p> <p>Establish secure communication sessions (HTTPS) using SSL</p>	<p>Instance Firewalls</p> <p>Configure firewall rules for instances using Security Groups</p>	<p>User Accounts</p> <p>Create individual IAM accounts so that each of your users has their own AWS security credentials</p>	<p>Subnet Control</p> <p>In your Virtual Private Cloud, create low-level networking constraints for resource access, such as public and private subnets, internet gateways, and NATs</p>

Account Control with IAM

- AWS Identity and Access Management (IAM):
 - Securely control access to AWS services and resources
 - Create and manage AWS users and groups
- AWS Master accounts should not be used for production systems!!
 - You should be using IAM user accounts.



Amazon Virtual Private Cloud (VPC)

SSL Endpoints	Security Groups	IAM	VPC
<p>Secure Transmission</p> <p>Establish secure communication sessions (HTTPS) using SSL</p>	<p>Instance Firewalls</p> <p>Configure firewall rules for instances using Security Groups</p>	<p>User Accounts</p> <p>Create individual IAM accounts so that each of your users has their own AWS security credentials</p>	<p>Subnet Control</p> <p>In your Virtual Private Cloud, create low-level networking constraints for resource access, such as public and private subnets, internet gateways, and NATs</p>

AWS Technical Essentials 3.6 ILT

EC2 Feature Consideration

Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides resizable compute capacity in the cloud.

Amazon EC2 provides a number of powerful features for building scalable, failure resilient, enterprise class applications

- Elastic Web-Scale Computing
 - Completely Controlled
 - Flexible Cloud Hosting Services
 - Designed for use with other Amazon Web Services
 - Reliable
 - Secure
 - Inexpensive
 - Multi Locations
 - Elastic IP address
 - Amazon Virtual Private Cloud
 - Auto Scaling
 - Amazon Cloud Watch
- AWS Technical Essentials 3.6 ILT
- Enhanced Networking
 - EC2 video link <https://www.youtube.com/watch?v=wNr7YqjjzOY>

EC2 Elastic Properties

- Elastic – Amazon EC2 enables you to increase or decrease capacity within minutes, not hours or days.
- You can commission one, hundreds or even thousands of server instances simultaneously.
- controlled with web service APIs, application can automatically scale itself up and down depending on its needs.
- Elastic Block Store vs. local Disk (not backup)
- Elastic IP Addresses vs. Static IP Addresses
- Interesting charging scheme; you are charged when not using it
- programmatically remapping your public IP addresses to any instance in your account

EC2 Instance Type

Instance Type	Instance Name	Application Suitability
General Purpose	T2,M3, M4	Development environments, build servers, code repositories, low-traffic websites and web applications, micro services, early product experiments, small databases
Compute Optimized	C3, C4	High performance front-end fleets, web-servers, batch processing, distributed analytics, high performance science and engineering applications, ad serving, MMO gaming, and video-encoding
Memory Optimized	R3	We recommend memory-optimized instances for high performance databases, distributed memory caches, in-memory analytics, genome assembly and analysis, larger deployments of SAP, Microsoft SharePoint, and other enterprise applications.
GPU	G2	3D application streaming, machine learning, video encoding, and other server-side graphics or GPU compute workload
Storage Optimized	I2	NOSQL Databases like Cassandra and MongoDB, scale out transactional databases, data warehousing, Hadoop, and cluster file systems.
Dense Storage	D2	Massively Parallel Processing (MPP) data warehousing, MapReduce and Hadoop distributed computing, distributed file systems, network file systems, log or data-processing applications

AW

AWS EC2 Pricing Model

- Free Usage Tier
- On-Demand Instances
 - Start and stop instances whenever you like, costs are rounded up to the nearest hour. (Worst price)
- Reserved Instances
 - Pay up front for one/three years in advance. (Best price)
 - Unused instances can be sold on a secondary market.
- Spot Instances
 - Specify the price you are willing to pay, and instances get started and stopped without any warning as the marked changes.

AWS EC2 Free UsageTier

- 750 hours of EC2 running Linux, RHEL, or SLES t2.micro instance usage
- 750 hours of EC2 running Microsoft Windows Server t2.micro instance usage
- 750 hours of Elastic Load Balancing plus 15 GB data processing
- 30 GB of Amazon Elastic Block Storage in any combination of General Purpose (SSD) or Magnetic, plus 2 million I/Os (with Magnetic) and 1 GB of snapshot storage
- 15 GB of bandwidth out aggregated across all AWS services
- 1 GB of Regional Data Transfer

Storage

- Instance –store : disappears with the instance (transient)
- Block storage: SAN-like, persists across time
- S3 is Object storage independent of an instance
- Glacier for archival purposes store it now and retrieve it at a later date
- Amazon: SimpleDB: Relational database better than MySQL or Oracle for reliability.

Elastic Block Storage (EBS)

- An EBS volume is a **virtual disk** of a fixed size with a block read/write interface. It can be **mounted** as a file system on a running EC2 instance where it can be **updated incrementally**. Unlike an instance store, an EBS volume is **persistent**.
- (Compare to an S3 object, which is essentially a file that must be accessed in its entirety.)
- Amazon EBS is particularly well-suited for use as the primary storage for a database or file system, or for any applications that require access to raw block-level storage
- Fundamental operations:
 - CREATE a new volume (1GB-1TB)
 - COPY a volume from an existing EBS volume or S3 object.
 - MOUNT on one instance at a time.
 - SNAPSHOT current state to an S3 object.

Simple Storage Service (S3)

- Simple Storage service is a storage for internet
- The number of objects you can store in S3 is unlimited
- A **bucket** is a container for objects and describes location, logging, accounting, and access control. A bucket can hold any number of **objects**, which are files of up to 5TB. A bucket has a name that must be **globally unique**.
- Fundamental operations corresponding to HTTP actions:
 - `http://bucket.s3.amazonaws.com/object`
 - POST a new object or update an existing object.
 - GET an existing object from a bucket.
 - DELETE an object from the bucket
 - LIST keys present in a bucket, with a filter.
- A bucket has a **flat directory structure** (despite the appearance given by the interactive web interface.)
- Amazon S3 works well for fast growing websites hosting data intensive, user-generated content, such as video and photo sharing sites.
- No set-up fee, No monthly minimum
- Video link <https://www.youtube.com/watch?v=1qrjFb0ZTm8>

S3 Bucket Naming

- Flat namespace
- Names may contain only lowercase letters, numbers, periods, underscores, and dashes, and must start with a number or letter
- Create your own namespace with your own buckets

Simple Storage Services (S3)

Bucket Properties

- Versioning – If enabled, POST/DELETE result in the creation of new versions without destroying the old.
- Lifecycle – Delete or archive objects in a bucket a certain time after creation or last access or number of versions.
- Access Policy – Control **when and where** objects can be accessed.
- Access Control – Control who **may** access objects in this bucket.
- Logging – Keep track of how objects are accessed.
- Notification – Be notified when failures occur.

Durability

- Amazon claims about S3:
 - Amazon S3 is designed to sustain the concurrent loss of data in two facilities, e.g. 3+ copies across multiple available domains.
 - 99.99999999% durability of objects over a given year.
- Amazon claims about EBS:
 - Amazon EBS volume data is replicated across multiple servers in an Availability Zone to prevent the loss of data from the failure of any single component.
 - Volumes <20GB modified data since last snapshot have an annual failure rate of 0.1% - 0.5%, resulting in complete loss of the volume.
 - Commodity hard disks have an AFR of about 4%.
- Amazon claims about Glacier is the same as S3:
 - Amazon S3 is designed to sustain the concurrent loss of data in two facilities, e.g. 3+ copies across multiple available domains PLUS periodic internal integrity checks.
 - 99.99999999% durability of objects over a given year.

AWS Auto Scaling Capability

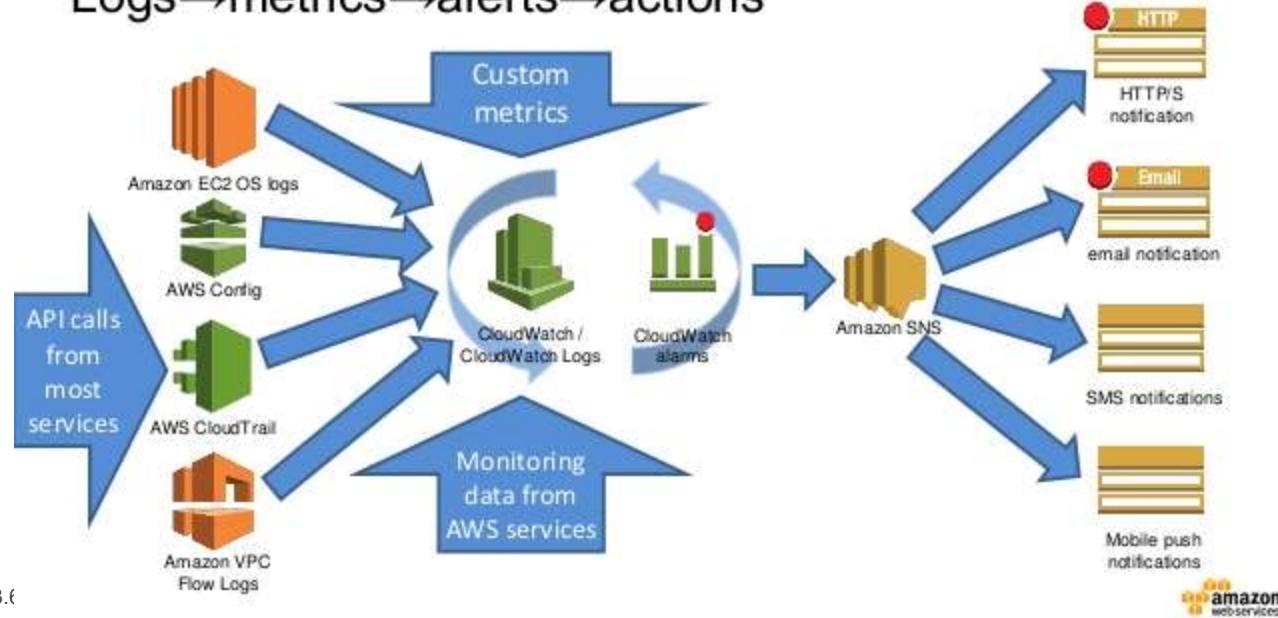
- Auto Scaling helps you maintain application availability and allows you to scale your [Amazon EC2](#) capacity up or down automatically according to conditions you define
 - Manage unhealthy EC2 compute instances
 - Ensure minimum number instances are always running
 - Launched new instances in event of failure or performance degradation (assume 30-120 seconds in most conditions)
 - Seamlessly attach auto scaled compute instances to load balancer (ELB)
- AWS Technical Essentials 3.6 ILT
- Video Link <https://www.youtube.com/watch?v=7SfVZqOVcCI>

AWS Elastic Load Balancer

- AWS ELB provides load balancing service with thousands of EC2 servers behind them
- AWS ELB will automatically Scale up /down the load balancing servers in backend
- The theoretical maximum response rate of AWS ELB is limitless
- It can handle 20,000+ concurrent requests easily

AWS Cloud Watch

Logs→metrics→alerts→actions



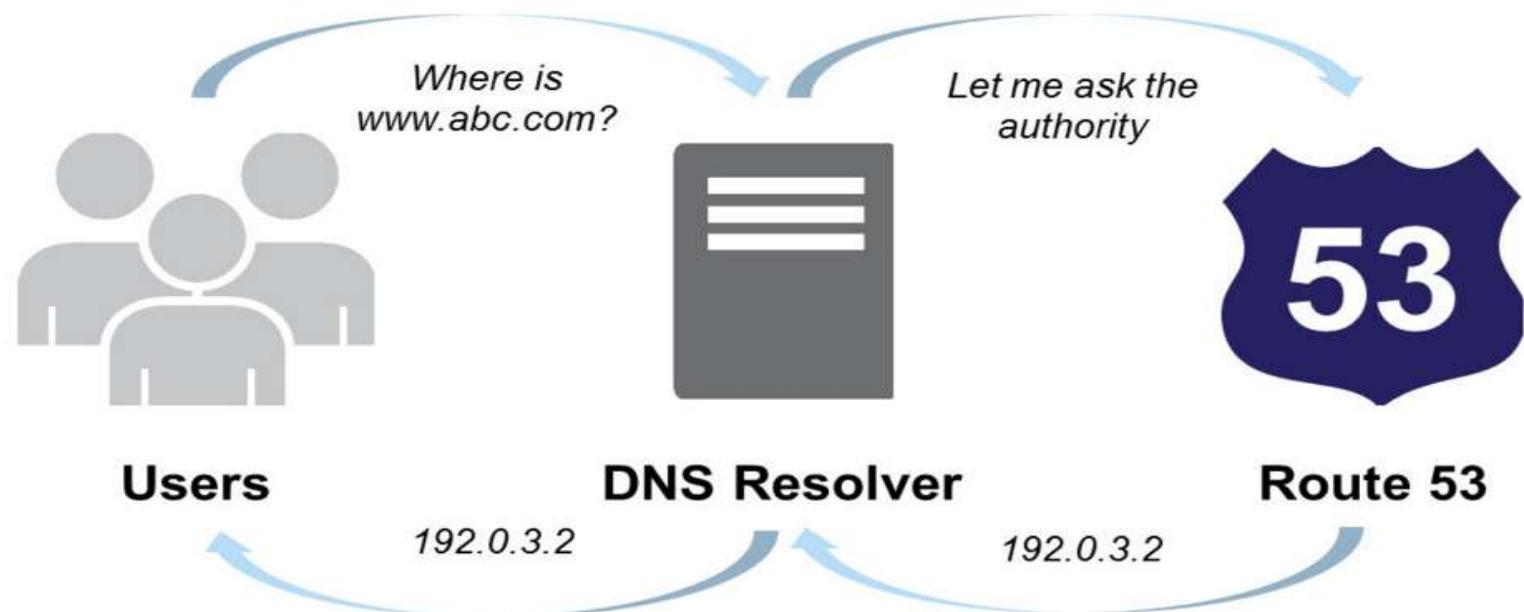
AWS Cloud Front

- Cloud-based content distributing network enables you to place the content at the edges of the network for rapid delivery.
- Place the contents in S3 and run the application from anywhere and the content is moved to where the application is (to the edges).
- Video Link

AWS Technical Essentials 3.6 ILT

https://www.youtube.com/watch?v=_oTj_3o1ceE

AWS Route 53



Amazon Relational Database Service (RDS)

- Amazon Relational Database Service a web service that provides the capabilities of MySQL, Oracle, or Microsoft SQL Server relational database as a managed, cloud-based service
- On-demand provisioning
- No operating system for you to access
- Platforms:
 - MySQL
 - Oracle
 - SQL Server
 - PostgreSQL
- “Mostly” pre-configured
- Basic monitoring / metrics provided
- Automated backups
- Automated recovery
- User initiated “snapshots”

AWS Technical **Automated replication**

- Software upgrades provided

RDS Features and Functionality

- Pre-configured Parameters
- Monitoring and Metrics
- Automatic Software Patching
- Automated Backups
- DB Snapshots
- Push-Button Scaling
- Automatic Host Replacement
- **Replication:** two features

AWS Technical Essentials 3.6 ILT

Multi-AZ Deployment, Read Replica

Multi-AZ Deployment

- **Availability Zones:** are distinct locations within a Region that are engineered to be isolated from failures in other Availability Zones.
- Run a DB Instance as a **Multi-AZ deployment**, the “primary” serves database writes and reads. Amazon RDS provisions and maintains a “standby” behind the scenes, which is an up-to-date replica of the primary. The standby is “promoted” in failover scenarios. After failover, the standby becomes the primary and accepts your database operations.

Oracle Deployment options

RDS

- Quick provisioning
- “Easy” management
- Simple environment setup
- Simplified replication strategy
- No OS-level control
- Limited granular fine tuning
- Limited platform / versions

EC2+Database

- You manage it yourself
- OS & storage overhead
- Software / version management
- Configuration
- Usually more costly
- More control over config / performance
- Allows more complex setups

Scalability- Scale Up vs. Scale Out

- Scale Up/Vertical Scaling
 - Select next available configuration (EC2, RDS)
 - Relatively simple but limited scalability
- Scale Out/Horizontal Scaling
 - Add additional resources
 - Complicated but high scalability
 - RDS Read replicas

RDS Limitation

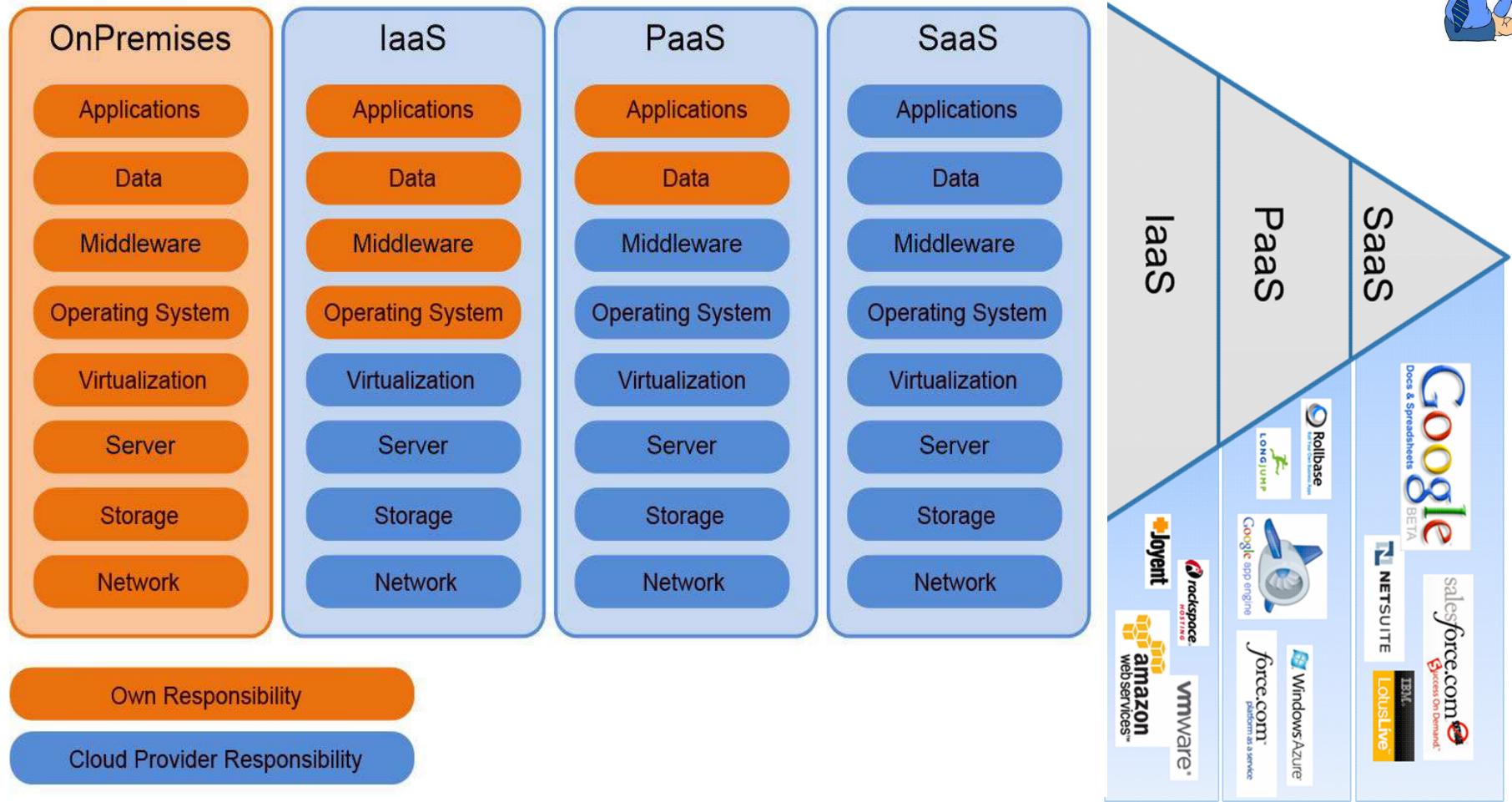
- Failovers are NOT instant.
 - Can take up to 6 minutes. Maybe more, depending on database size
 - Only limited platforms / versions are supported.
 - Upgrades / patching requires downtime.
 - You set an allowable maintenance window. Amazon will upgrade sometime during that window.
 - Can suffer from “noisy neighbor” syndrome.
 - Performance issues are sometimes hard to pinpoint
- Master-Master replication is NOT supported



heard of 3 models of Cloud Computing?



Yes, Yes, IaaS, PaaS and SaaS





BITS Pilani
Pilani Campus

BITS Pilani presentation

Mridul Moitra
Cloud Computing



BITS Pilani

<CSI ZG527 / SS ZG527 / SE ZG527
Cloud Computing
Lecture No. 3



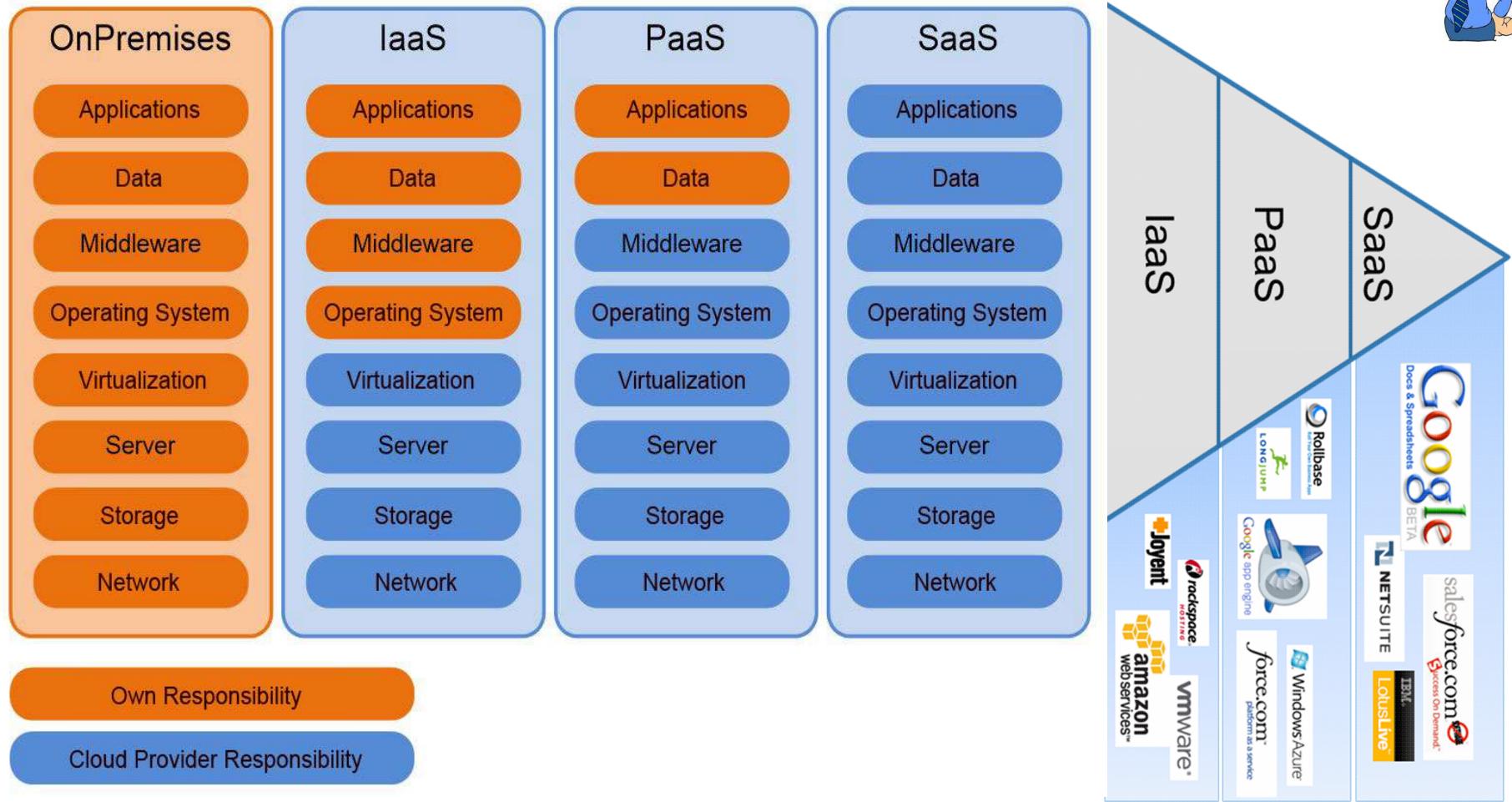
IaaS



heard of 3 models of Cloud Computing?



Yes, Yes, IaaS, PaaS and SaaS



Key concepts of IaaS

- Cloudbursting: The process of off-loading tasks to the cloud during times when the most compute resources are needed

- Multi-tenant computing
-
- The diagram illustrates the transition from a single-tenant architecture to a multitenant architecture. On the left, under the heading 'single-tenant', there is a box labeled 'Separate application and separate databases'. Inside this box, four colored human icons (green, yellow, orange, blue) each have a vertical stack of four boxes below them: 'Application' (green), 'Application' (yellow), 'Application' (orange), and 'Application' (blue). Each 'Application' box is connected by a downward arrow to a 'Database' box (green, yellow, orange, blue respectively). A large grey arrow points from this single-tenant section to the right, leading into the 'multitenant' section. In the 'multitenant' section, there is a box labeled 'One shared application and one shared database'. Inside this box, the same four colored human icons (green, yellow, orange, blue) all point to a single 'Application' box at the top, which in turn points down to a single 'Database' box at the bottom.
- Resource pooling: **Pooling** is a resource management term that refers to the grouping together of resources (compute(cpu), network(bandwidth), storage) for the purposes of **maximizing advantage** and/or **minimizing risk** to the users
 - Hypervisor

Two primary facets that make IaaS special

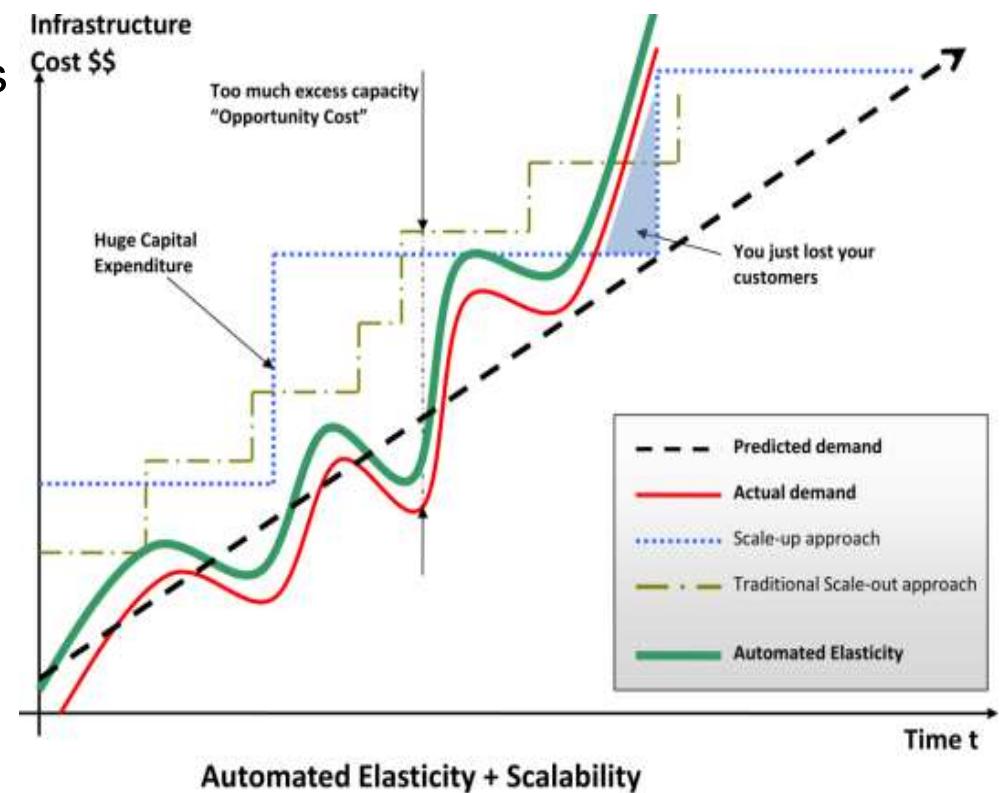
Elasticity:

Wikipedia: “In **cloud** computing, **elasticity** is defined as the degree to which a system (or a particular **cloud** layer) autonomously adapts its capacity to workload over time”

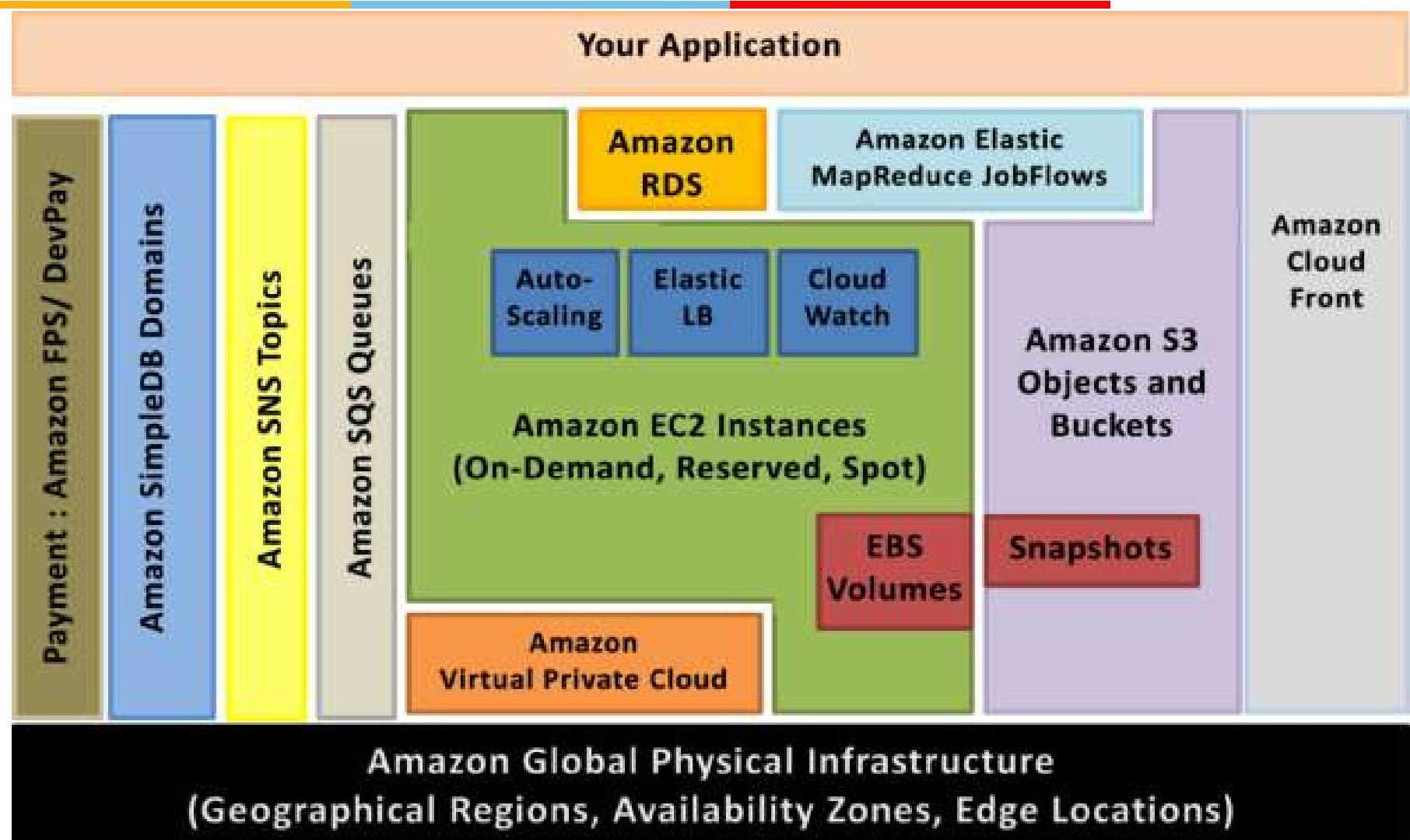
OR simply put “Ability of a system to **expand** or **contract** its dedicated resources to meet the demand”

&

Virtualization



AWS infrastructure services



Amazon Elastic Compute Cloud (Amazon EC2)

- Web service that provides resizable compute capacity in the cloud
- Can be bundled with OS, application software and associated configuration settings into an Amazon Machine Image (AMI).
- Use these AMIs to provision multiple virtualized instances
- Decommission them using simple web service calls to scale capacity up and down quickly, as capacity requirement changes.
- On-Demand Instances - pay for the instances by the hour
- Reserved Instances - pay a low, one-time payment and receive a lower usage rate to run the instance
- Spot Instances - bid for unused capacity and further reduce your cost.
- Instances can be launched in one or more geographical regions.
- Each region has multiple Availability Zones. Availability Zones are distinct locations that are engineered to be insulated from failures in other Availability Zones

EC2 Elastic Properties

- Elastic – Amazon EC2 enables you to increase or decrease capacity within minutes, not hours or days.
- You can commission one, hundreds or even thousands of server instances simultaneously.
- controlled with web service APIs, application can automatically scale itself up and down depending on its needs.
- Elastic Block Store vs. local Disk (not backup)
- Elastic IP Addresses vs. Static IP Addresses
- Interesting charging scheme; you are charged when not using it
- programmatically remapping your public IP addresses to any instance in your account

EC2 Instance Type

Instance Type	Instance Name	Application Suitability
General Purpose	T2,M3, M4	Development environments, build servers, code repositories, low-traffic websites and web applications, micro services, early product experiments, small databases
Compute Optimized	C3, C4	High performance front-end fleets, web-servers, batch processing, distributed analytics, high performance science and engineering applications, ad serving, MMO gaming, and video-encoding
Memory Optimized	R3	We recommend memory-optimized instances for high performance databases, distributed memory caches, in-memory analytics, genome assembly and analysis, larger deployments of SAP, Microsoft SharePoint, and other enterprise applications.
GPU	G2	3D application streaming, machine learning, video encoding, and other server-side graphics or GPU compute workload
Storage Optimized	I2	NOSQL Databases like Cassandra and MongoDB, scale out transactional databases, data warehousing, Hadoop, and cluster file systems.
Dense Storage	D2	Massively Parallel Processing (MPP) data warehousing, MapReduce and Hadoop distributed computing, distributed file systems, network file systems, log or data-processing applications

AW

AWS EC2 Pricing Model

- Free Usage Tier
- On-Demand Instances
 - Start and stop instances whenever you like, costs are rounded up to the nearest hour. (Worst price)
- Reserved Instances
 - Pay up front for one/three years in advance. (Best price)
 - Unused instances can be sold on a secondary market.
- Spot Instances
 - Specify the price you are willing to pay, and instances get started and stopped without any warning as the marked changes.

AWS EC2 Free UsageTier

- 750 hours of EC2 running Linux, RHEL, or SLES t2.micro instance usage
- 750 hours of EC2 running Microsoft Windows Server t2.micro instance usage
- 750 hours of Elastic Load Balancing plus 15 GB data processing
- 30 GB of Amazon Elastic Block Storage in any combination of General Purpose (SSD) or Magnetic, plus 2 million I/Os (with Magnetic) and 1 GB of snapshot storage
- 15 GB of bandwidth out aggregated across all AWS services
- 1 GB of Regional Data Transfer

Infrastructure Services

- **Amazon S3** is highly durable and distributed data store. With a simple web services interface, store and retrieve large amounts of data as objects in buckets (containers) at any time, using standard HTTP
- **AWS Identity and Access Management (IAM)** – enables multiple User creation with unique security credentials and manage the permissions for each of these Users

Storage

- Instance store : disappears with the instance (transient)
- Block storage: SAN-like, persists across time
- S3 is Object storage independent of an instance
- Glacier for archival purposes store it now and retrieve it at a later date

Elastic Block Storage (EBS)

- An EBS volume is a **virtual disk** of a fixed size with a block read/write interface. It can be **mounted** as a file system on a running EC2 instance where it can be **updated incrementally**. Unlike an instance store, an EBS volume is **persistent**.
- (Compare to an S3 object, which is essentially a file that must be accessed in its entirety.)
- Amazon EBS is particularly well-suited for use as the primary storage for a database or file system, or for any applications that require access to raw block-level storage
- Fundamental operations:
 - CREATE a new volume (1GB-1TB)
 - COPY a volume from an existing EBS volume or S3 object.
 - MOUNT on one instance at a time.
 - SNAPSHOT current state to an S3 object.

Simple Storage Service (S3)

- Simple Storage service is a storage for internet
- The number of objects you can store in S3 is unlimited
- A **bucket** is a container for objects and describes location, logging, accounting, and access control. A bucket can hold any number of **objects**, which are files of up to 5TB. A bucket has a name that must be **globally unique**.
- Fundamental operations corresponding to HTTP actions:
 - `http://bucket.s3.amazonaws.com/object`
 - POST a new object or update an existing object.
 - GET an existing object from a bucket.
 - DELETE an object from the bucket
 - LIST keys present in a bucket, with a filter.
- A bucket has a **flat directory structure** (despite the appearance given by the interactive web interface.)
- Amazon S3 works well for fast growing websites hosting data intensive, user-generated content, such as video and photo sharing sites.
- No set-up fee, No monthly minimum
- Video link <https://www.youtube.com/watch?v=1qrjFb0ZTm8>

S3 Bucket Naming

- Flat namespace
- Names may contain only lowercase letters, numbers, periods, underscores, and dashes, and must start with a number or letter
- Create your own namespace with your own buckets

Simple Storage Services (S3)

Bucket Properties

- Versioning – If enabled, POST/DELETE result in the creation of new versions without destroying the old.
- Lifecycle – Delete or archive objects in a bucket a certain time after creation or last access or number of versions.
- Access Policy – Control **when and where** objects can be accessed.
- Access Control – Control who **may** access objects in this bucket.
- Logging – Keep track of how objects are accessed.
- Notification – Be notified when failures occur.

Durability

- Amazon claims about S3:
 - Amazon S3 is designed to sustain the concurrent loss of data in two facilities, e.g. 3+ copies across multiple available domains.
 - 99.99999999% durability of objects over a given year.
- Amazon claims about EBS:
 - Amazon EBS volume data is replicated across multiple servers in an Availability Zone to prevent the loss of data from the failure of any single component.
- Amazon claims about Glacier is the same as S3:
 - Amazon S3 is designed to sustain the concurrent loss of data in two facilities, e.g. 3+ copies across multiple available domains PLUS periodic internal integrity checks.
 - 99.99999999% durability of objects over a given year.

AWS Auto Scaling Capability

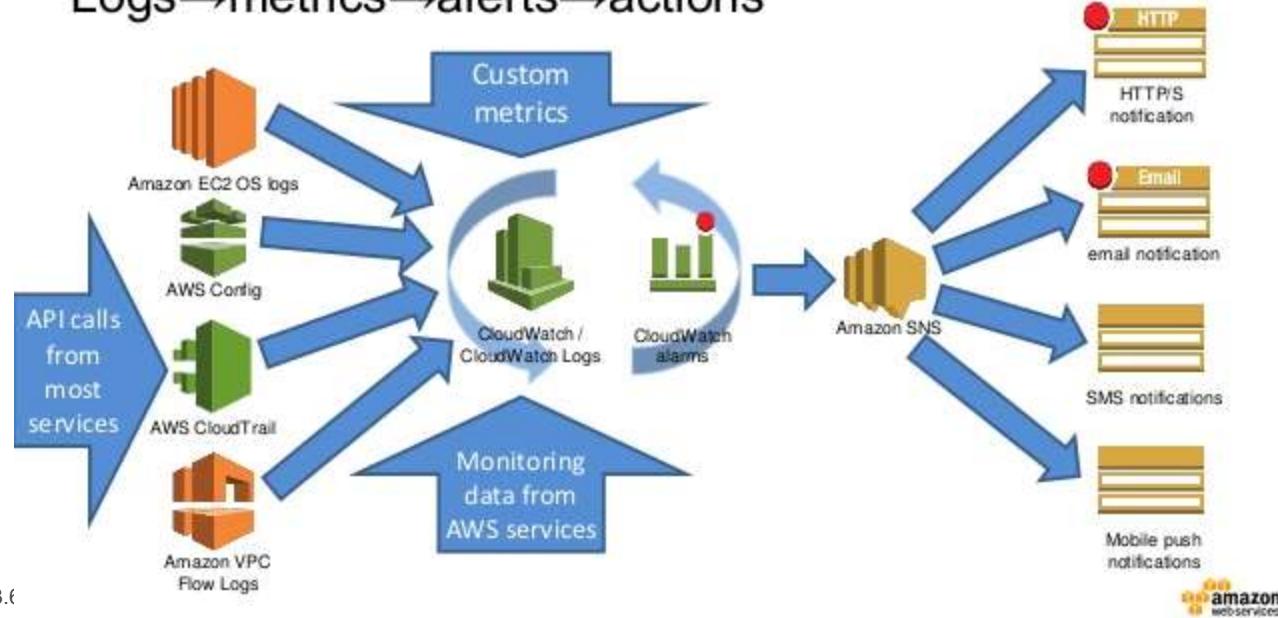
- Auto Scaling helps you maintain application availability and allows you to scale your [Amazon EC2](#) capacity up or down automatically according to conditions you define
- Manage unhealthy EC2 compute instances
- Ensure minimum number instances are always running
- Launched new instances in event of failure or performance degradation (assume 30-120 seconds in most conditions)
- Seamlessly attach auto scaled compute instances to load balancer (ELB)

AWS Elastic Load Balancer

- AWS ELB provides load balancing service with thousands of EC2 servers behind them
- AWS ELB will automatically Scale up /down the load balancing servers in backend
- The theoretical maximum response rate of AWS ELB is limitless
- It can handle 20,000+ concurrent requests easily

AWS Cloud Watch

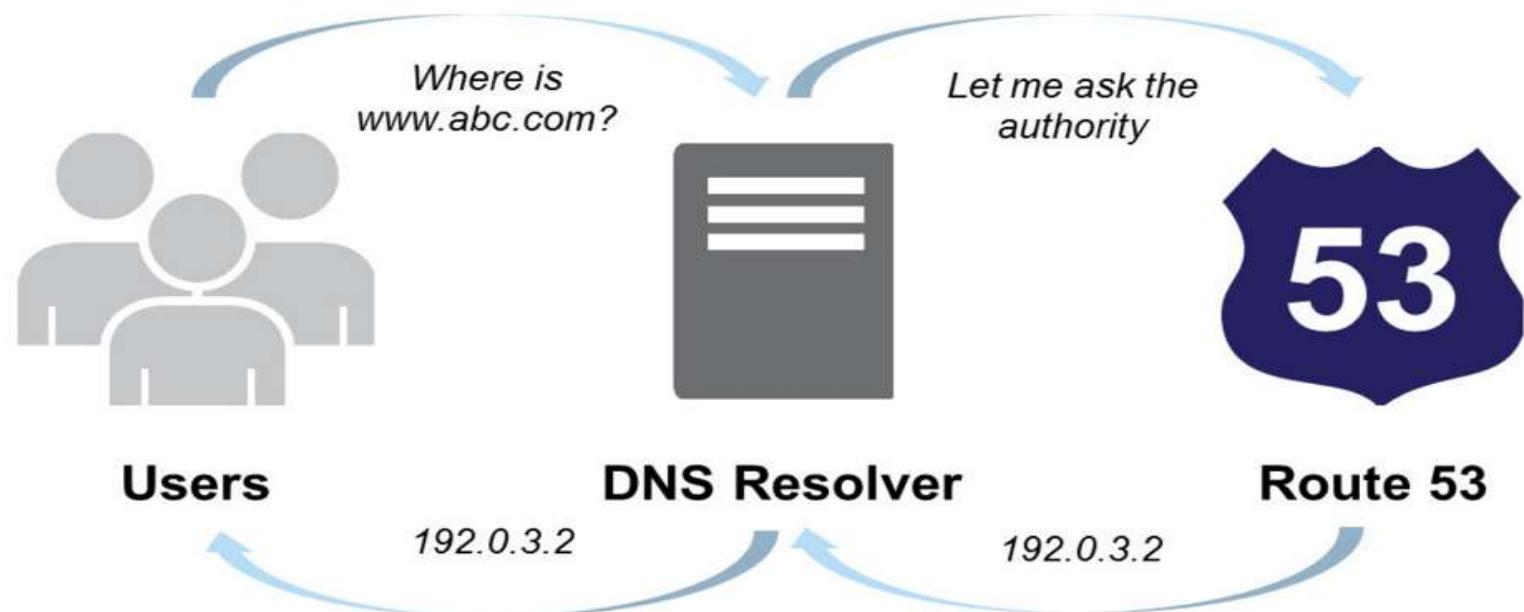
Logs→metrics→alerts→actions



AWS Cloud Front

- Cloud-based content distributing network enables you to place the content at the edges of the network for rapid delivery.
- Place the contents in S3 and run the application from anywhere and the content is moved to where the application is (to the edges).

AWS Route 53



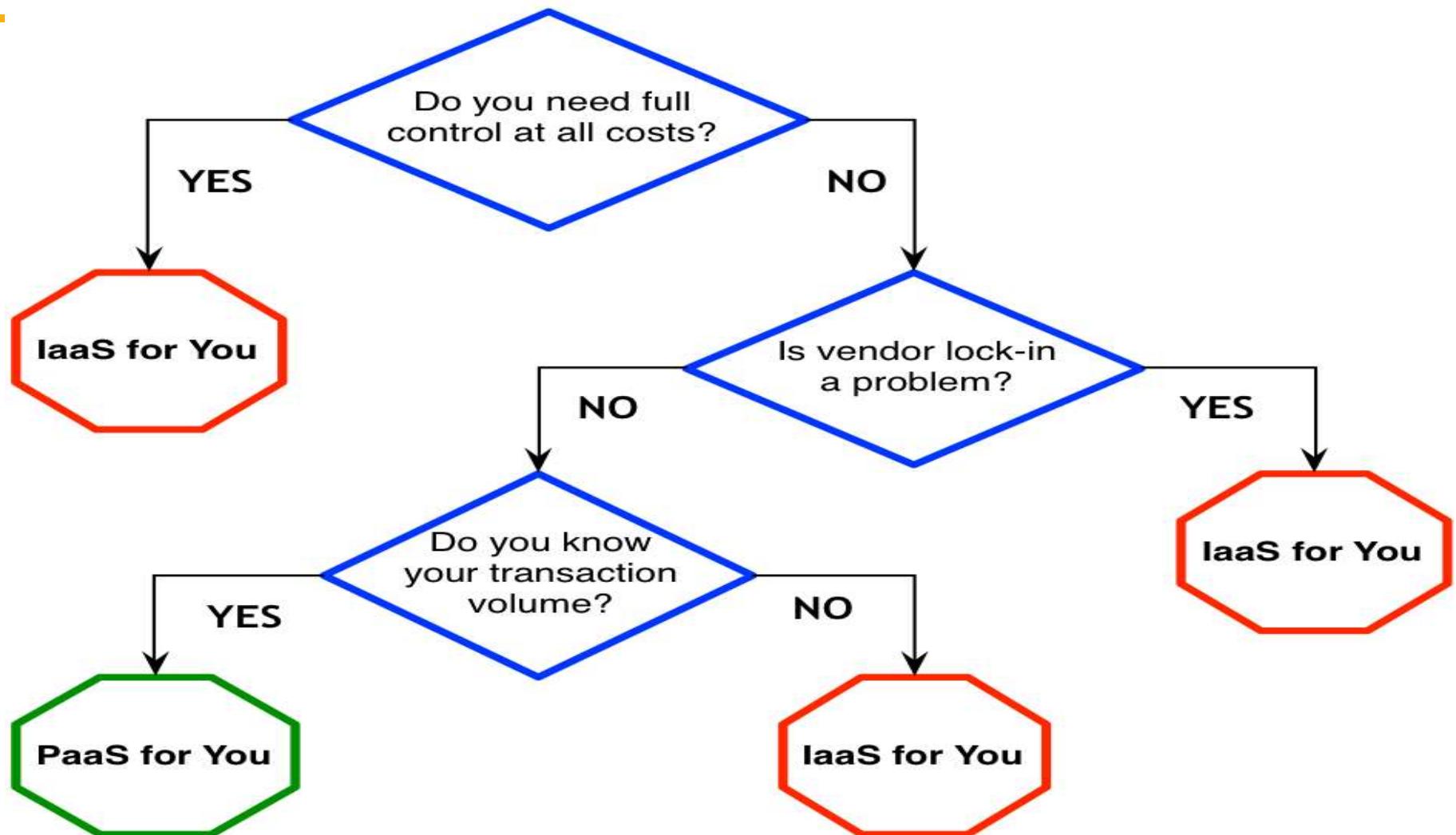
Multi-AZ Deployment

- **Availability Zones:** are distinct locations within a Region that are engineered to be isolated from failures in other Availability Zones.
- Run a DB Instance as a **Multi-AZ deployment**, the “primary” serves database writes and reads. Amazon RDS provisions and maintains a “standby” behind the scenes, which is an up-to-date replica of the primary. The standby is “promoted” in failover scenarios. After failover, the standby becomes the primary and accepts your database operations.

Scalability- Scale Up vs. Scale Out

- Scale Up/Vertical Scaling
 - Select next available configuration (EC2, RDS)
 - Relatively simple but limited scalability
- Scale Out/Horizontal Scaling
 - Add additional resources
 - Complicated but high scalability
 - RDS Read replicas

IaaS or PaaS Decision Tree

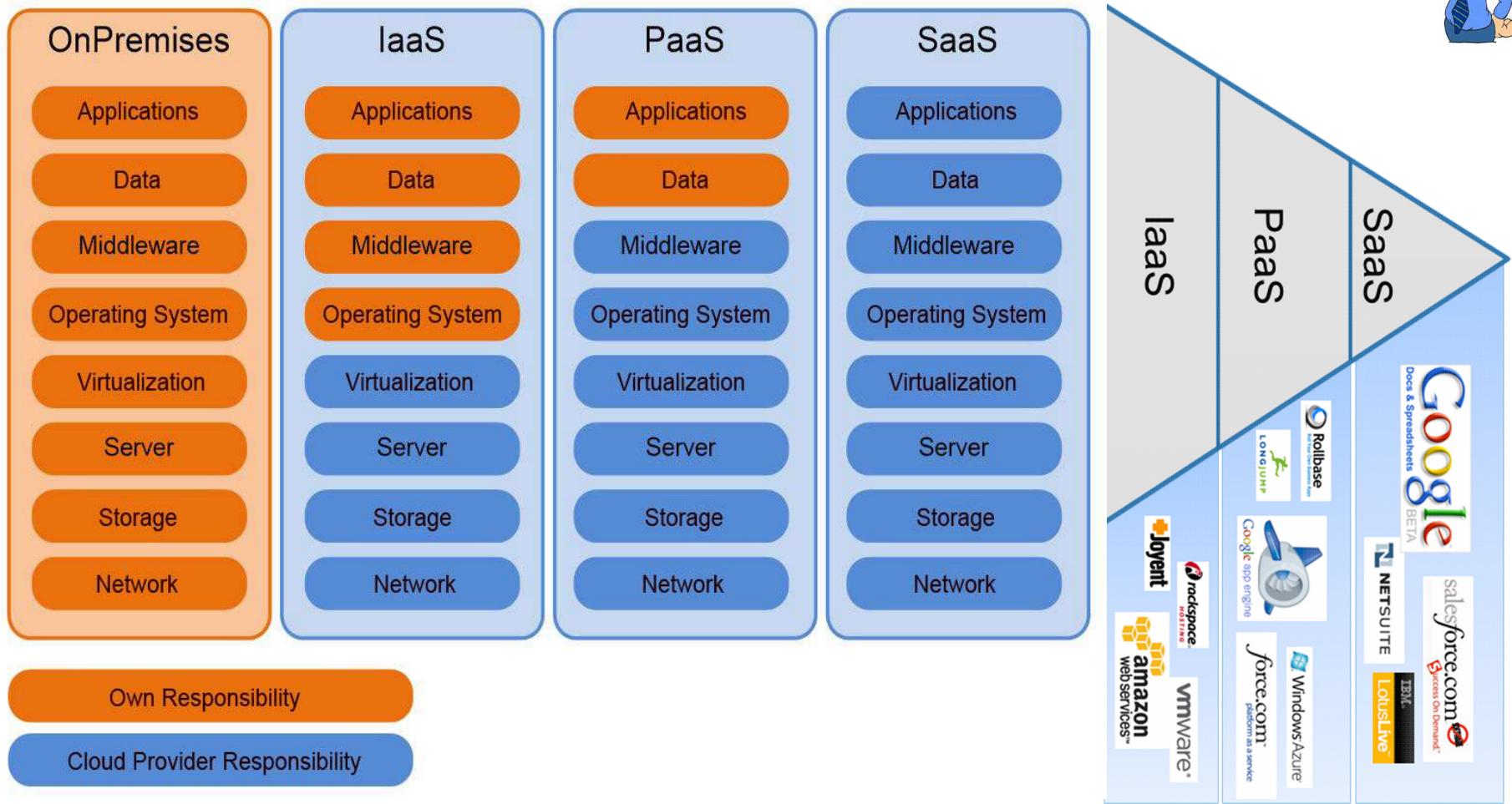




heard of 3 models of Cloud Computing?



Yes, Yes, IaaS, PaaS and SaaS

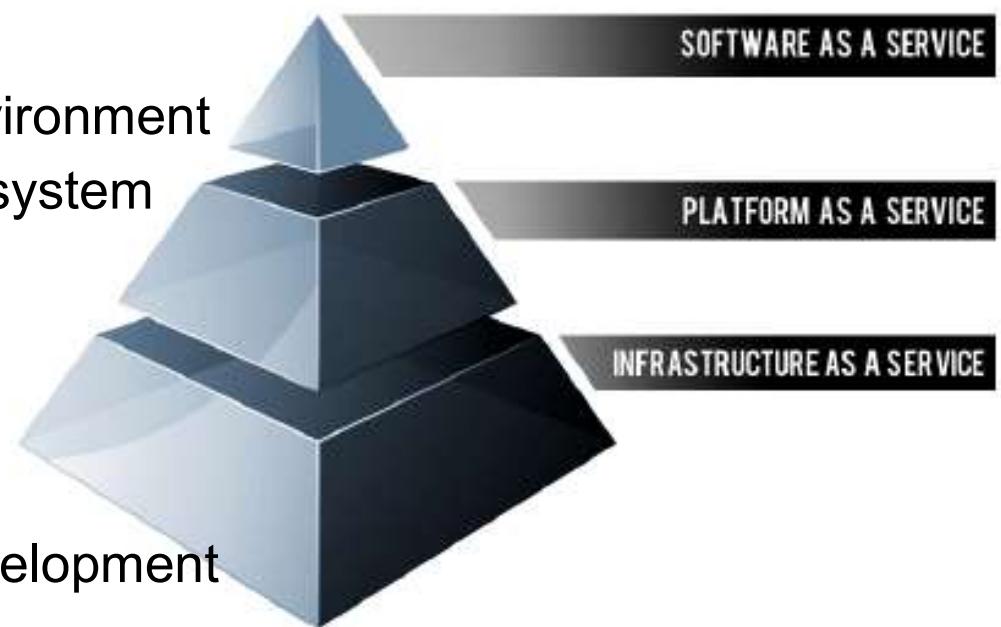


Introduction to PaaS

- Platform as a Service, referred to as PaaS, is a category of cloud computing that provides a platform and environment to allow developers to build applications and services over the internet.
- Platform as a Service allows users to create software applications using tools supplied by the provider.
- PaaS services are hosted in the cloud and accessed by users simply via their web browser.
- PaaS services can consist of preconfigured features that customers can subscribe to; they can choose to include the features that meet their requirements while discarding those that do not.

Building blocks of PaaS

- PaaS providers can assist developers from the conception of their original ideas to the creation of applications, and through to testing and deployment.
- Below are some of the features that can be included with a PaaS offering:
 - Operating system
 - Server-side scripting environment
 - Database management system
 - Server Software
 - Support
 - Storage
 - Network access
 - Tools for design and development
 - Hosting



Characteristics of PaaS

- Services to develop, test, deploy, host and maintain applications in the same integrated development environment. All the varying services needed to fulfill the application development process
- Web based user interface creation tools help to create, modify, test and deploy different UI scenarios
- Multi-tenant architecture where multiple concurrent users utilize the same development application
- Built in scalability of deployed software including load balancing and failover
- Integration with web services and databases via common standards
- Support for development team collaboration – some PaaS solutions include project planning and communication tools
- Tools to handle billing and subscription management

Characteristics of PaaS

PaaS, which is similar in many ways to Infrastructure as a Service, is differentiated from IaaS by the addition of value added services and comes in two distinct flavours;

1. A collaborative platform for software development, focused on workflow management regardless of the data source being used for the application. An example of this approach would be Heroku, a PaaS that utilizes the Ruby on Rails development language.
2. A platform that allows for the creation of software utilizing proprietary data from an application. This sort of PaaS can be seen as a method to create applications with a common data form or type. An example of this sort of platform would be the Force.com PaaS from Salesforce.com which is used almost exclusively to develop applications that work with the Salesforce.com CRM

Advantages and Risks

Advantages

- Users don't have to invest in physical infrastructure
- PaaS allows developers to frequently change or upgrade operating system features. It also helps development teams collaborate on projects.
- Makes development possible for 'non-experts'
- Teams in various locations can work together
- Security is provided, including data security and backup and recovery.
- Adaptability; Features can be changed if circumstances dictate that they should.
- Flexibility; customers can have control over the tools that are installed within their platforms and can create a platform that suits their specific requirements. They can 'pick and choose' the features they feel are necessary.

Advantages and Risks

Risks

- Since users rely on a provider's infrastructure and software, vendor lock-in can be an issue in PaaS environments.
- Other risks associated with PaaS are provider downtime or a provider changing its development roadmap.
- If a provider stops supporting a certain programming language, users may be forced to change their programming language, or the provider itself. Both are difficult and disruptive steps.

Amazon Relational Database Service (RDS)

- **Amazon Relational Database Service** a web service that provides the capabilities of MySQL, Oracle, or Microsoft SQL Server relational database as a managed, cloud-based service
- **On-demand provisioning**
- **No operating system for you to access**
- **Platforms:**
 - MySQL
 - Oracle
 - SQL Server
 - PostgreSQL
- “Mostly” pre-configured
- Basic monitoring / metrics provided
- Automated backups
- Automated recovery
- User initiated “snapshots”
- Automated replication
- Software upgrades provided

RDS Features and Functionality

- Pre-configured Parameters
- Monitoring and Metrics
- Automatic Software Patching
- Automated Backups
- DB Snapshots
- Push-Button Scaling
- Automatic Host Replacement
- Replication: two features
- Multi-AZ Deployment, Read Replica

Oracle Deployment options

RDS

- Quick provisioning
- “Easy” management
- Simple environment setup
- Simplified replication strategy
- No OS-level control
- Limited granular fine tuning
- Limited platform / versions

EC2+Database

- You manage it yourself
- OS & storage overhead
- Software / version management
- Configuration
- Usually more costly
- More control over config / performance
- Allows more complex setups

RDS Limitation

- Failovers are NOT instant.
 - Can take up to 6 minutes. Maybe more, depending on database size
- Only limited platforms / versions are supported.
- Upgrades / patching requires downtime.
 - You set an allowable maintenance window. Amazon will upgrade sometime during that window.
- Can suffer from “noisy neighbor” syndrome.
- Performance issues are sometimes hard to pinpoint
- Master-Master replication is NOT supported



BITS Pilani
Pilani Campus

BITS Pilani presentation

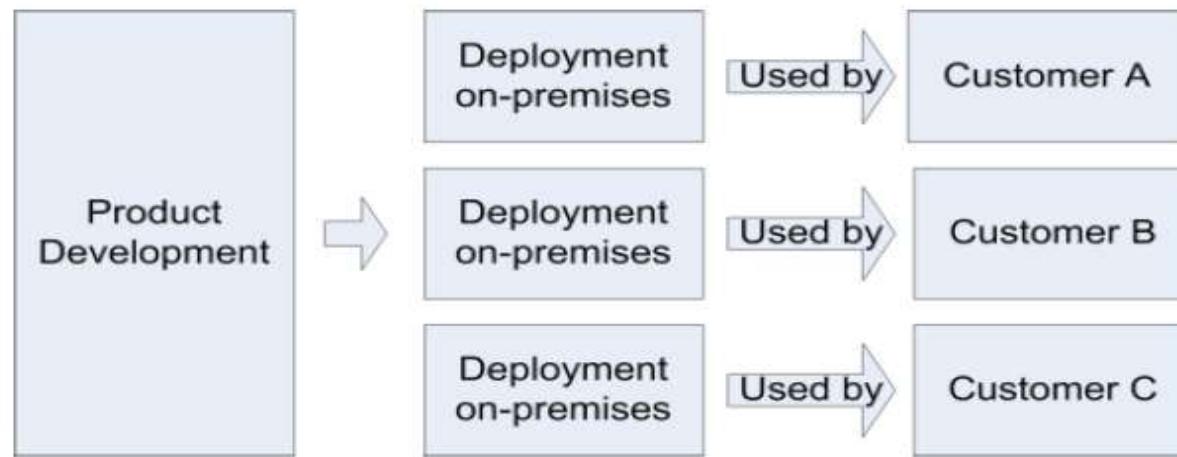
Mridul Moitra
Cloud Computing



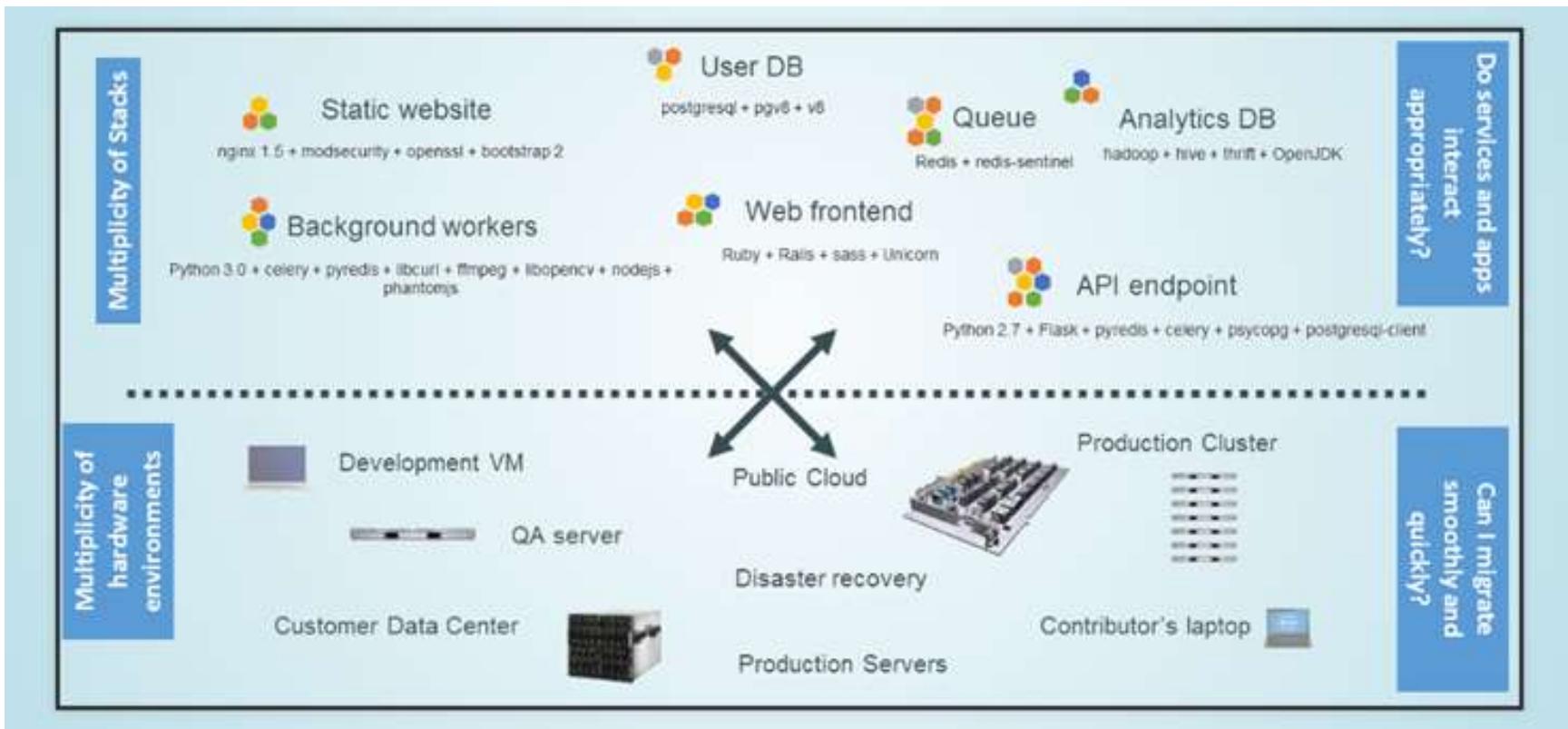
BITS Pilani

<CSI ZG527 / SS ZG527 / SE ZG527
Cloud Computing- Docker Technology
Lecture No. 5

Traditional Deployment Model



Challenges



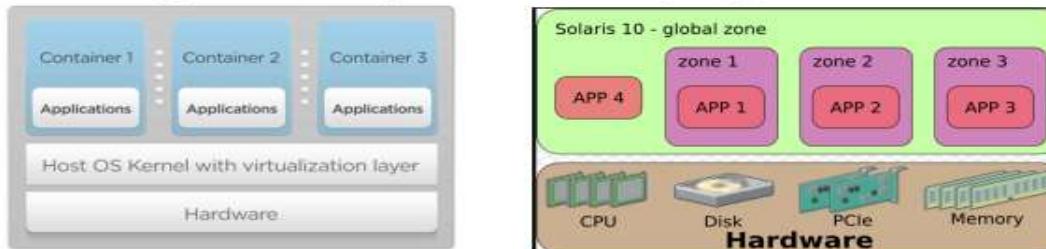
Introduction

- Linux containers (LXC) are “lightweight” VMs
- Docker is a commoditized LXC technique that dramatically simplifies the use of LXC

OS Virtualization

OS Virtualization

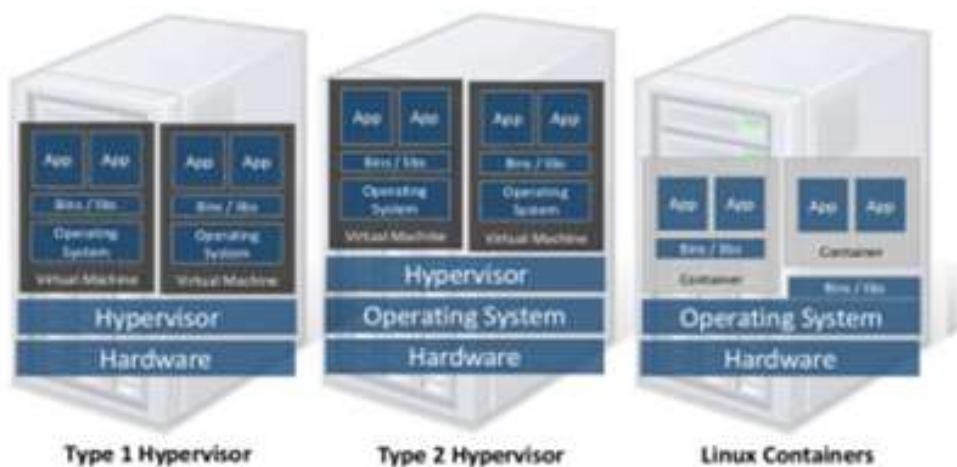
- Emulate OS-level interface with native interface
- “Lightweight” virtual machines
 - No hypervisor, OS provides necessary support



- Referred to as *containers*
 - Solaris containers, BSD jails, Linux containers

Linux Container

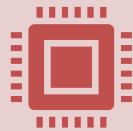
- Containers share OS kernel of the host
 - OS provides resource isolation
- Benefits
 - Fast provisioning, bare-metal like performance, lightweight



OS Mechanisms for LXC

- OS mechanisms for resource isolation and management
- namespaces: process-based resource isolation
- Cgroups: limits, prioritization, accounting, control
- chroot: apparent root directory
- Linux security module, access control
- Tools (e.g., docker) for easy management

Linux Namespaces



Linux kernel provides
the “control groups”
(cgroups) functionality

allows limitation and prioritization of
resources (CPU, memory, block I/O,
network, etc.) without the need for
starting any VM



“namespace
isolation” functionality

allows complete isolation of an
applications' view of the operating
environment,
including process trees, networking, user
IDs and mounted file systems

Container Features

- Containers running in the user space
- Each container has
 - Own process space
 - Own network interface
 - Own /sbin/init (coordinates the rest of the boot process and configures the environment for the user)
 - Run stuff as root
- Share kernel with the host
- No device emulation

Isolation with cgroups

- Memory
- Cpu
- Blkio
- devices

Memory cgroup

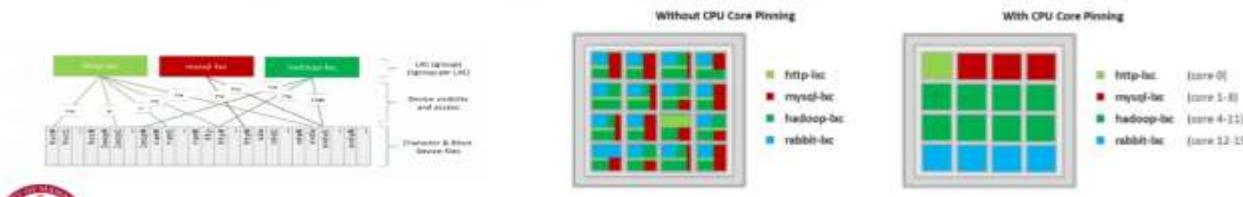
- keeps track pages used by each group:
 - file (read/write/mmap from block devices; swap)
 - anonymous (stack, heap, anonymous mmap)
 - active (recently accessed)
 - inactive (candidate for eviction)
- each page is charged to a group
- pages can be shared
- Individual (per-cgroup) limits and out-of-memory killer

CPU cgroup

- keep track of user/system CPU time
- set relative weight per group
- pin groups to specific CPU(s)
 - Can be used to reserve CPUs for some apps

Linux CGROUPS

- Resource isolation
 - what and how much can a container use?
 - Set upper bounds (limits) on resources that can be used
 - Fair sharing of certain resources
- Examples:
 - cpu: weighted proportional share of CPU for a group
 - cpuset: cores that a group can access
 - block io: weighted proportional block IO access
 - memory: max memory limit for a group



Blkio cgroup

- keep track IOs for each block device
 - read vs write; sync vs async
- set relative weights
- set throttle (limits) for each block device
 - read vs write; bytes/sec vs operations/sec

Devices cgroup

- controls read/write/mknod permissions
- typically:
 - allow: /dev/{tty,zero,random,null}...
 - deny: everything else
 - maybe: /dev/net/tun, /dev/fuse, /dev/kvm, /dev/dri...
- fine-grained control for GPU, virtualization, etc

Almost no overhead

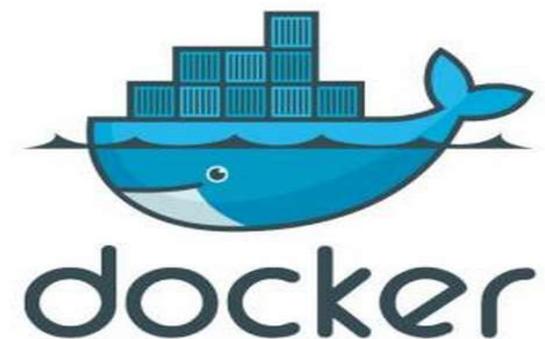
- Processes are isolated, but run straight on the host
- CPU performance = native performance
- Memory performance = a few % shaved off for (optional) accounting
- Network performance = small overhead; can be reduced to zero

Proportional Share Scheduling

- Uses a variant of *proportional-share scheduling*
- *Share-based* scheduling:
 - Assign each process a weight w_i (a “share”)
 - Allocation is proportional to share
 - fairness: reused unused cycles to others in proportion to weight
 - Examples: fair queuing, start time fair queuing
- *Hard limits*: assign upper bounds (e.g., 30%), no reallocation
- Credit-based: allocate credits every time T, can accumulate credits, and can burst up-to credit limit
 - can a process starve other processes?

Docker

Introduction and Demo



Agenda

01

What is Docker

02

Why use Docker

03

How to setup Docker in Linux

04

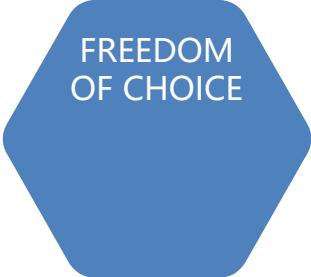
Commands & References

01

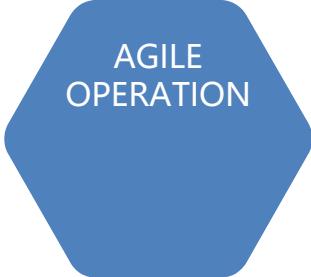
What is Docker

What is Docker - Overview

Docker is the company driving the container movement and the only container platform provider to address every application across the hybrid cloud. Today's businesses are under pressure to digitally transform but are constrained by existing applications and infrastructure while rationalizing an increasingly diverse portfolio of clouds, datacenters and application architectures. Docker enables true independence between applications and infrastructure and developers and IT ops to unlock their potential and creates a model for better collaboration and innovation.



FREEDOM
OF CHOICE



AGILE
OPERATION



INTEGRATED
SECURITY

Docker history

- 2013-03: Releases as Open Source
- 2013-09: Red Hat collaboration (Fedora, RHEL, OpenShift)
- 2014-03: 34th most starred GitHub project
- 2014-05: JAX Innovation Award (most innovative open technology)

What is Docker?

Docker is a software platform that allows you to build, test, and deploy applications quickly, packaging software into standardized units called containers.

- Open Source engine to commoditize LXC
- using copy-on-write for quick provisioning
- allowing to **create and share images**
- **standard format** for containers
- standard, *reproducible* way to *easily* build *trusted* images
(Dockerfile, Stackbrew...)

What is Docker – Basic Concepts

LXC

LXC(Linux Containers) is an operating-system-level virtualization method for running multiple isolated Linux systems(containers) on a control host using a single Linux kernel. [Wikipedia](#)

CGroups & Namespaces

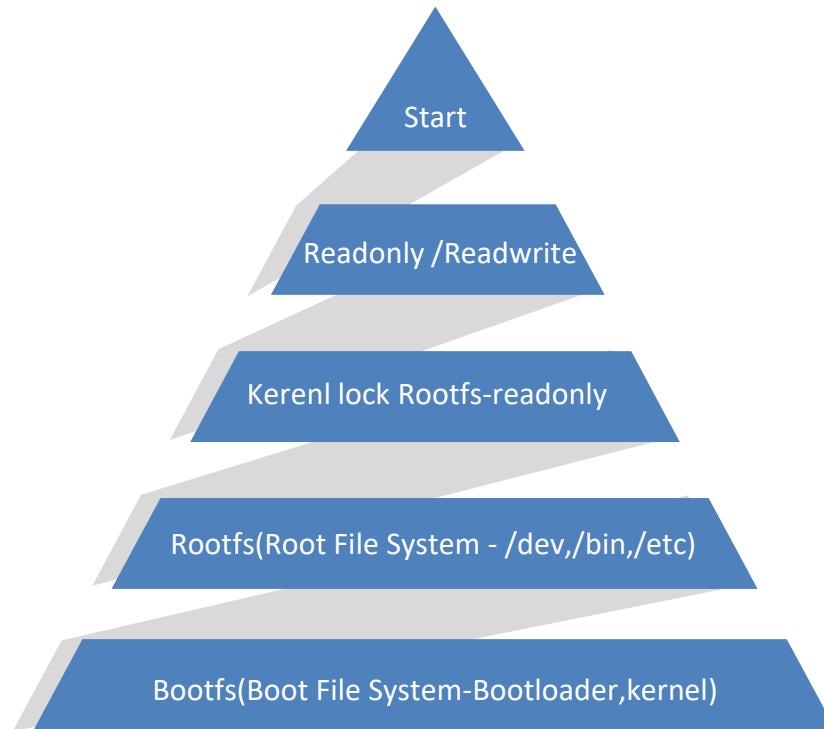
The [Linux kernel](#) provides the [cgroups](#) functionality that allows limitation and prioritization of resources (CPU, memory, block I/O, network, etc.) without the need for starting any [virtual machines](#), and also [namespace isolation](#) functionality that allows complete isolation of an applications' view of the operating environment, including [process trees](#),[networking](#), [user IDs](#) and [mounted file systems](#)

AUFS

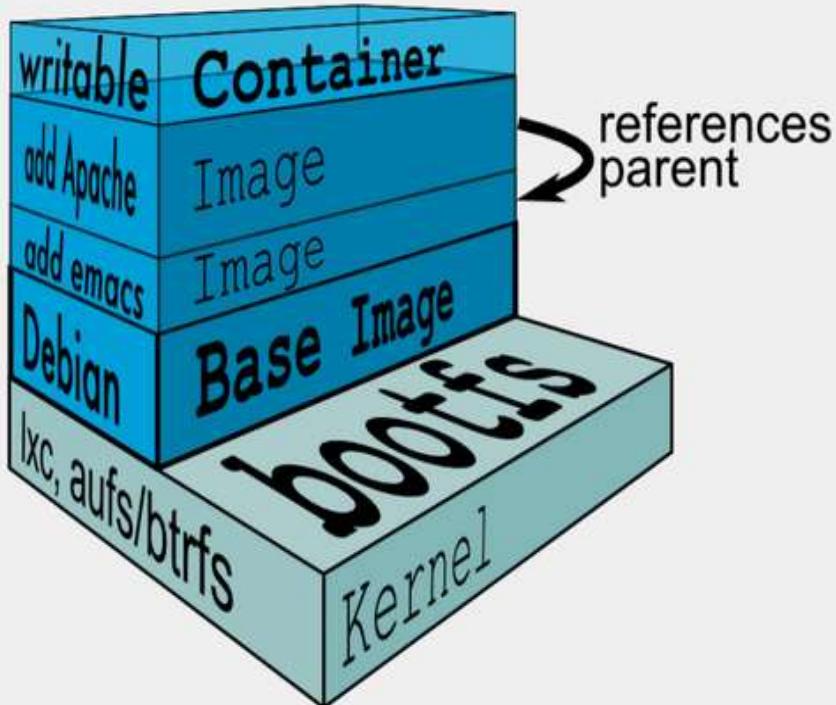
AUFS (short for advanced multi-layered unification filesystem) implements a [union mount](#) for [Linux file systems](#)

What is Docker – Basic Concepts

Linux Boot

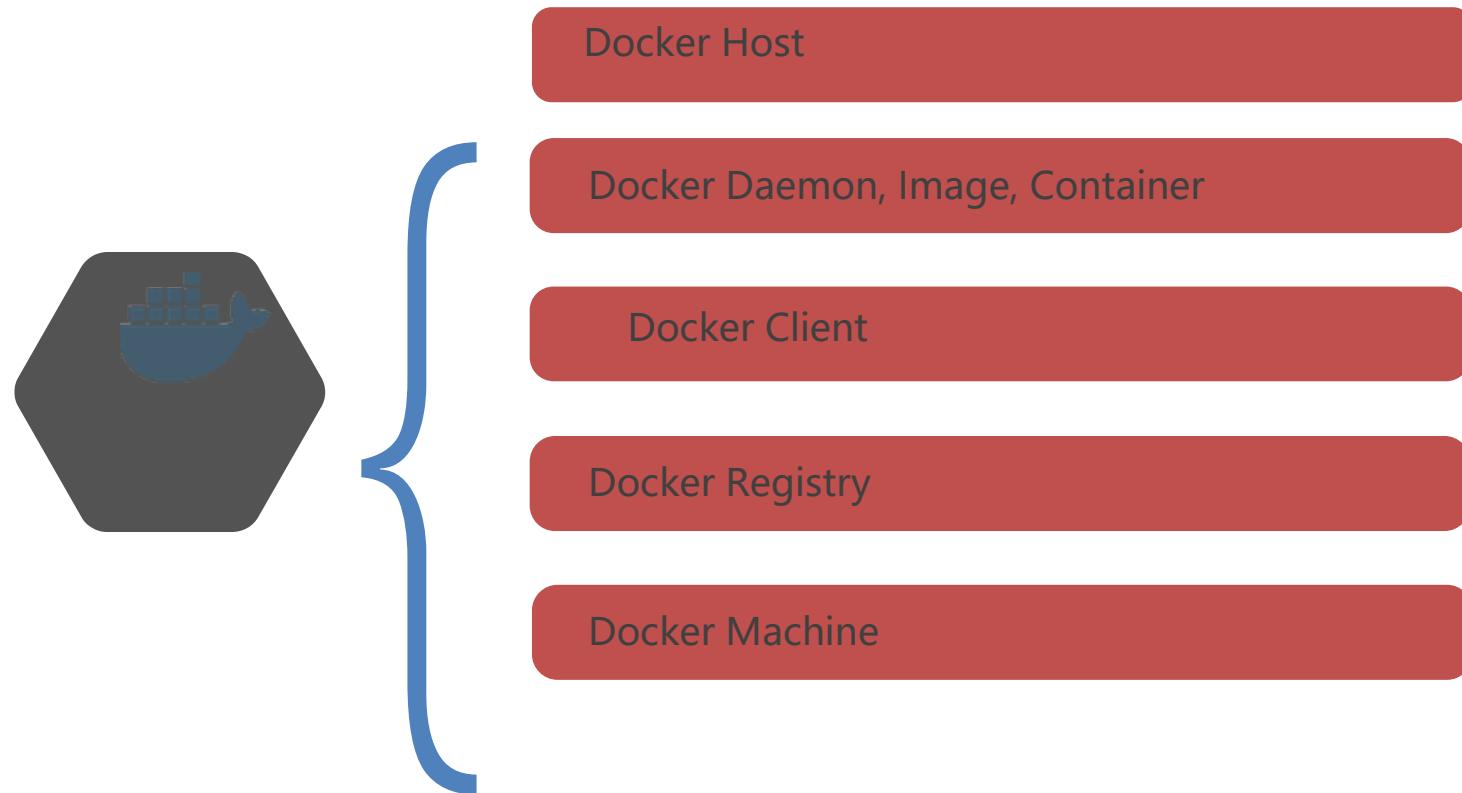


Docker Images and Uses

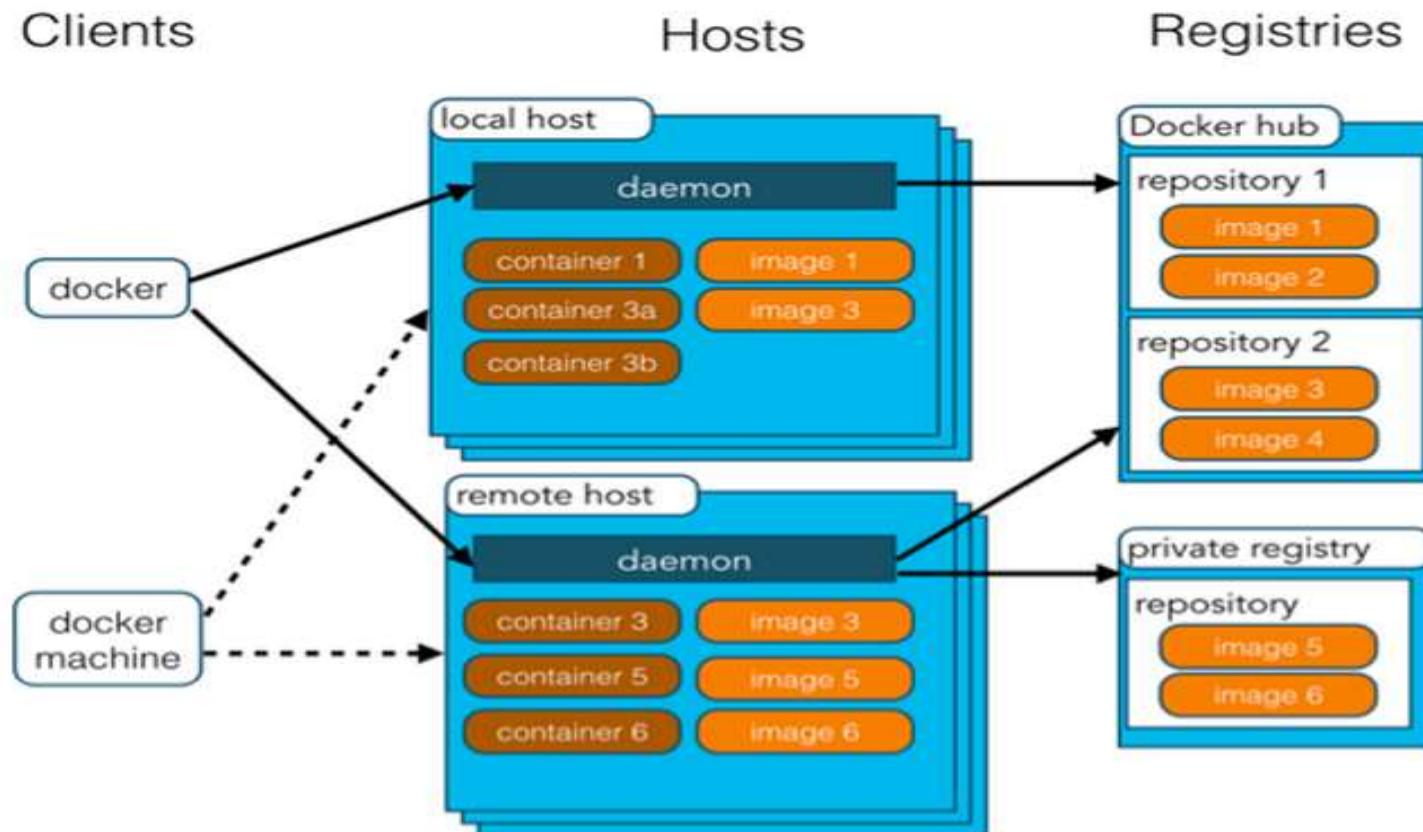


- Docker uses a union file system (AuFS)
 - allows containers to use host FS safely
- Essentially a copy-on-write file system
 - read-only files shared (e.g., share glibc)
 - make a copy upon write
- Allows for small efficient container images
- Docker Use Cases
 - “Run once, deploy anywhere”
 - Images can be pulled/pushed to repository
 - Containers can be a single process (useful for microservices) or a full OS

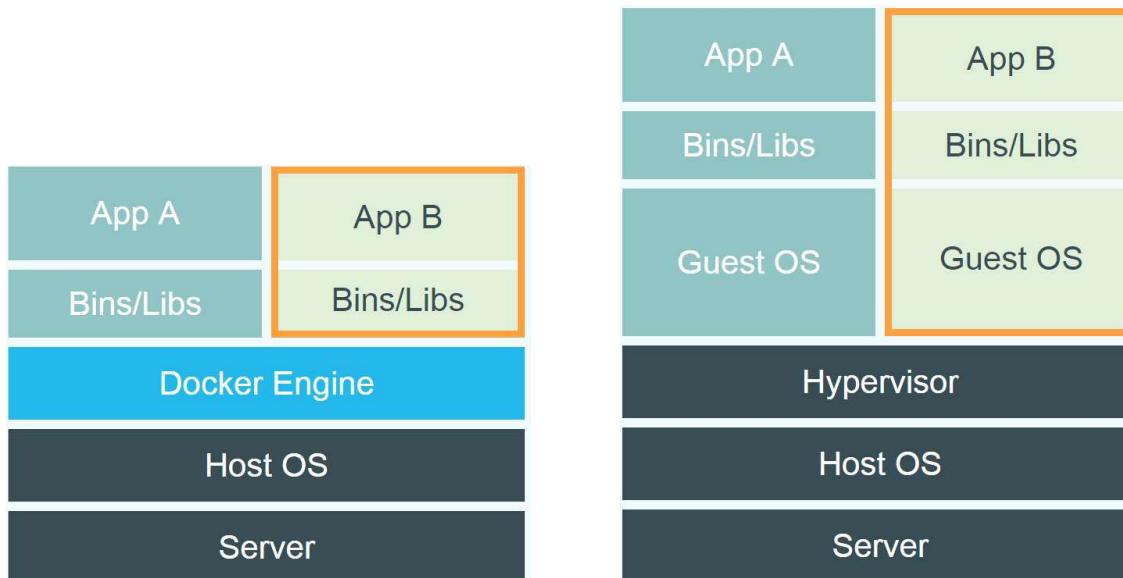
What is Docker - Contains



What is Docker - Contains



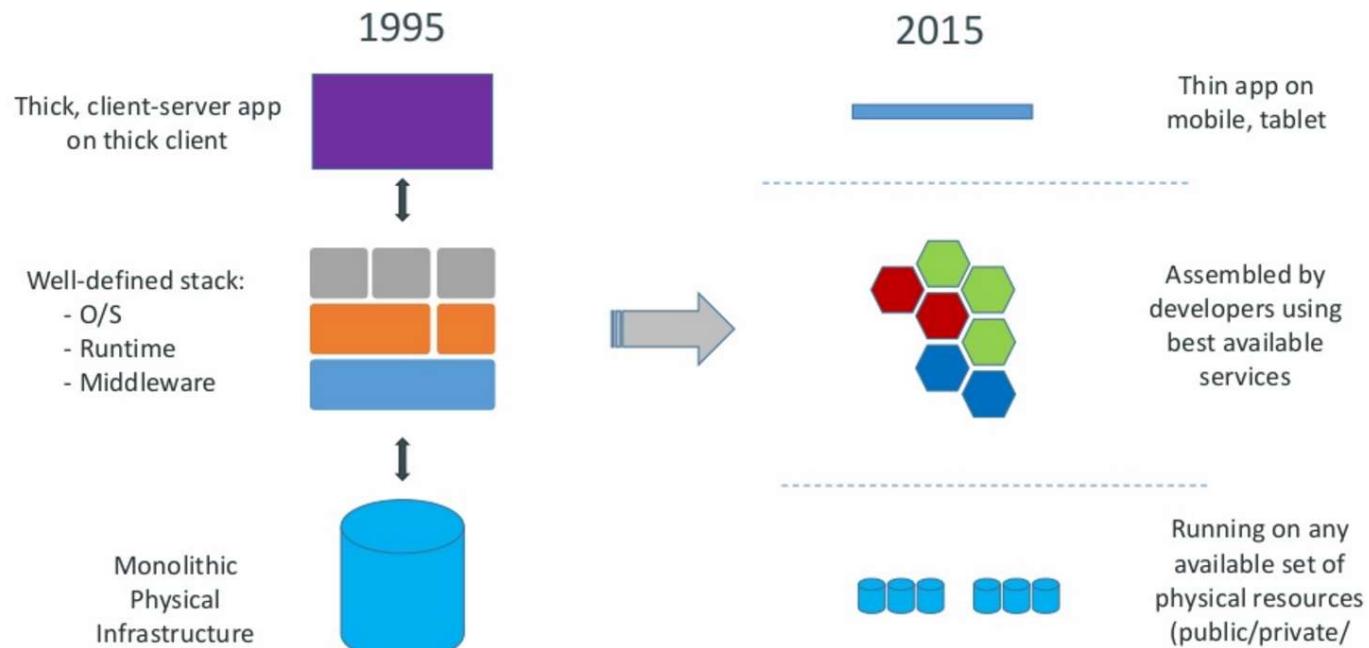
Comparison between LXC/docker and VM



02

Why use Docker

Motivation

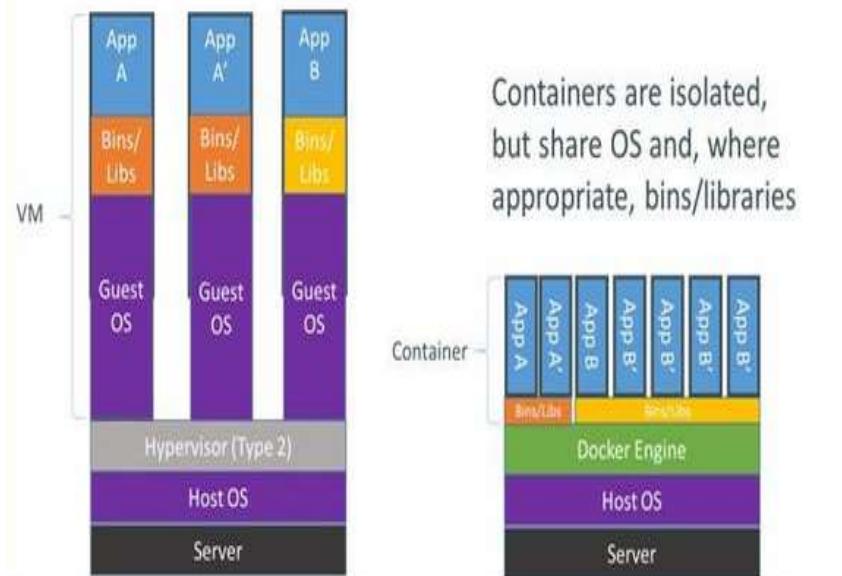


Docker

- Minimal learning curve
- Rebuilds are easy
- Caching system makes rebuilds faster
- Single file to define the whole environment!

Why use Docker – Compare with VM

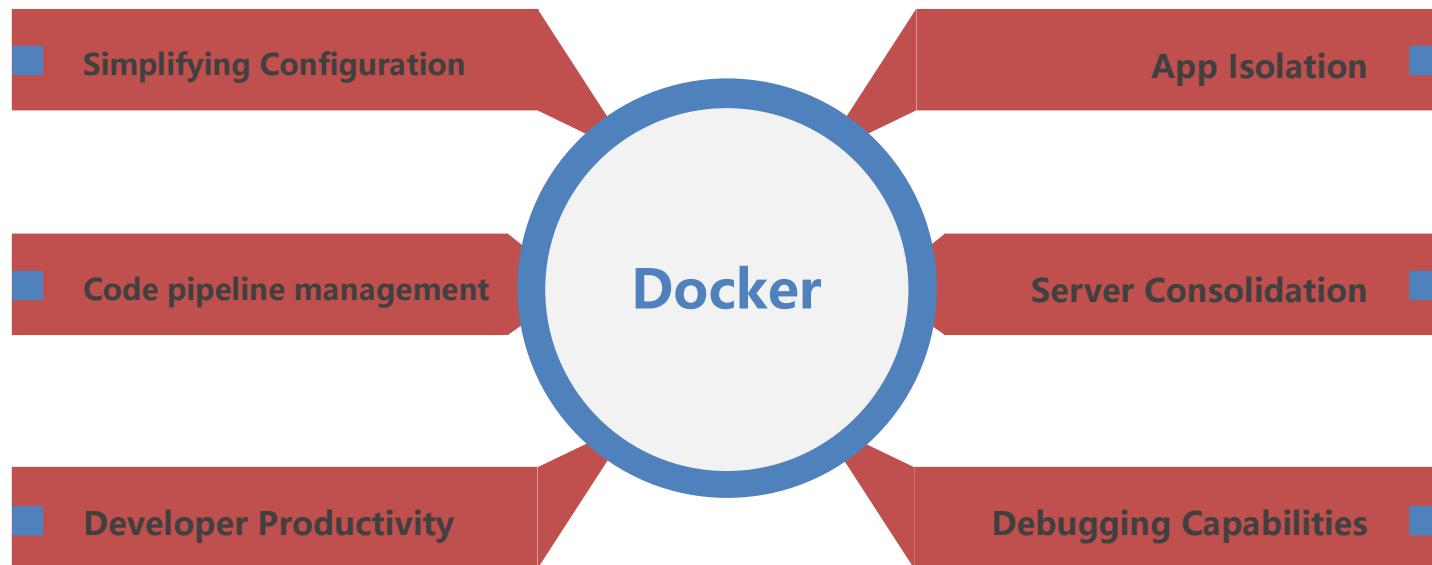
Containers vs. VMs



Hyper-V (OS at least 5G)
VMWare, AWS EC2

Docker

Why use Docker – Advantages



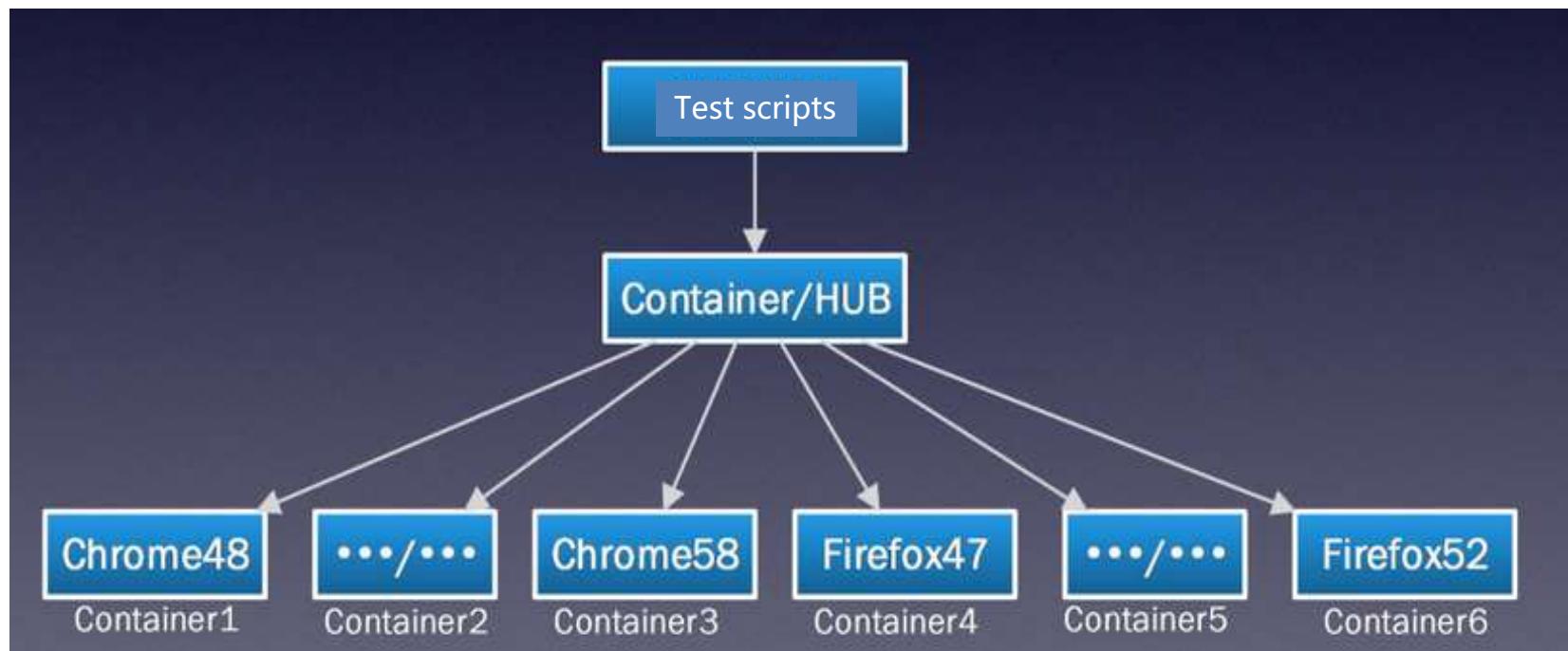
Deploy Reliability and Consistently

- If it works locally, it will work on the server
- *With exactly the same behavior*
- Regardless of versions
- Regardless of distros
- Regardless of dependencies

Deploy Efficiently

- Containers are lightweight
 - Typical laptop runs 10-100 containers easily
 - Typical server can run 100-1000 containers
- Containers can run at native speeds
 - Lies, damn lies, and other benchmarks:

Why use Docker – Docker in Test



Docker container—developer viewpoint

Build once...run anywhere

- A clean, safe, hygienic and portable runtime environment for your app.
- No worries about missing dependencies, packages and other pain points during subsequent deployments.
- Run each app in its own isolated container, so you can run various versions of libraries and other dependencies for each app without worrying
- Automate testing, integration, packaging...anything you can script
- Reduce/eliminate concerns about compatibility on different platforms, either your own or your customers.
- Cheap, zero-penalty containers to deploy services? A VM without the overhead of a VM? Instant replay and reset of image snapshots? That's the power of Docker

Administrative Benefits

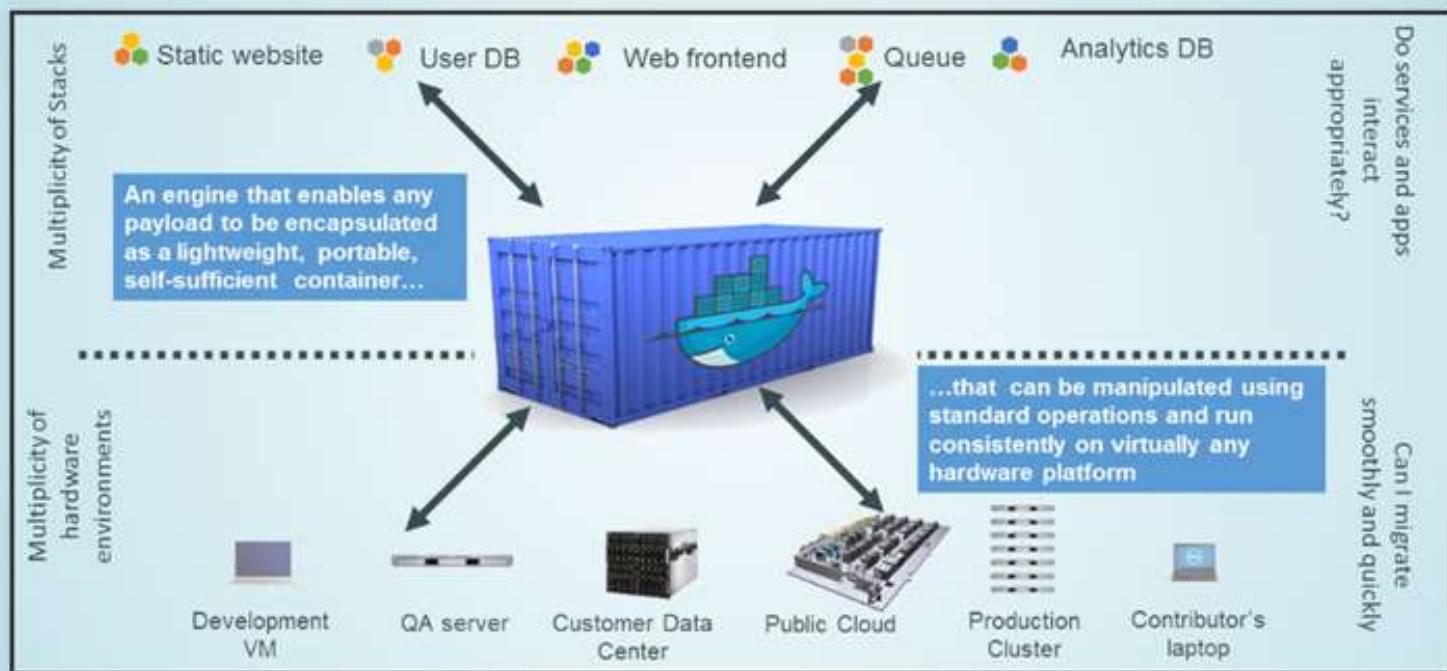
Why Administrators Care

Configure once... run anything

- Make the entire lifecycle more efficient, consistent, and repeatable
- Increase the quality of code produced by developers.
- Eliminate inconsistencies between development, test, production, and customer environments.
- Support segregation of duties.
- Significantly improves the speed and reliability of continuous deployment and continuous integration systems.
- Because the containers are so lightweight, address significant performance, costs, deployment, and portability issues normally associated with VMs.

Docker Code Deployment

Docker is a Container System for Code



Docker Technical Details

More Technical Details

Why

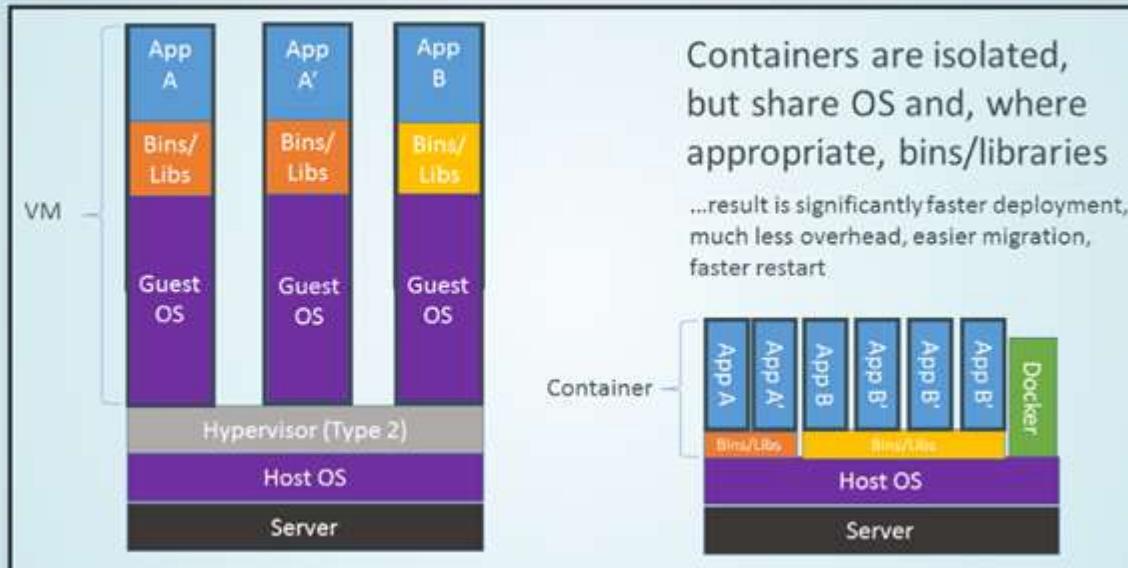
- Run everywhere
 - Regardless of kernel version
 - Regardless of host distro
 - Physical or virtual, cloud or not
 - Container and host architecture must match...
- Run anything
 - If it can run on the host, it can run in the container
 - If it can on a Linux kernel, it can run

What

- High level: a lightweight VM
 - Own process space
 - Own network interface
 - Can run stuff as root
 - Can have its own /sbin/init (different from host)
 - <>machine container<>
- Low level: chroot on steroids
 - Can also *not* have its own /sbin/init
 - Container = isolated processes
 - Share kernel with host
 - <>application container<>

Comparison between VMS vs Containers

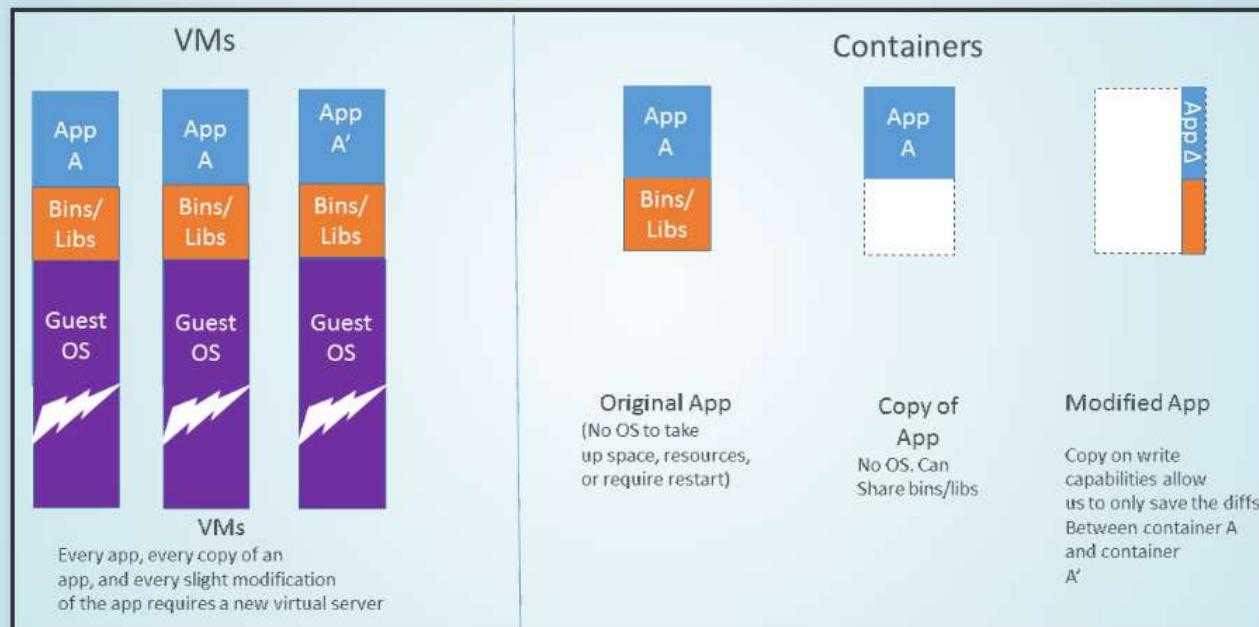
VMs vs Containers



Inbox (11,141) - mridulmoitra@googlemail.com | docker ppt - Google Search | Intro to Docker

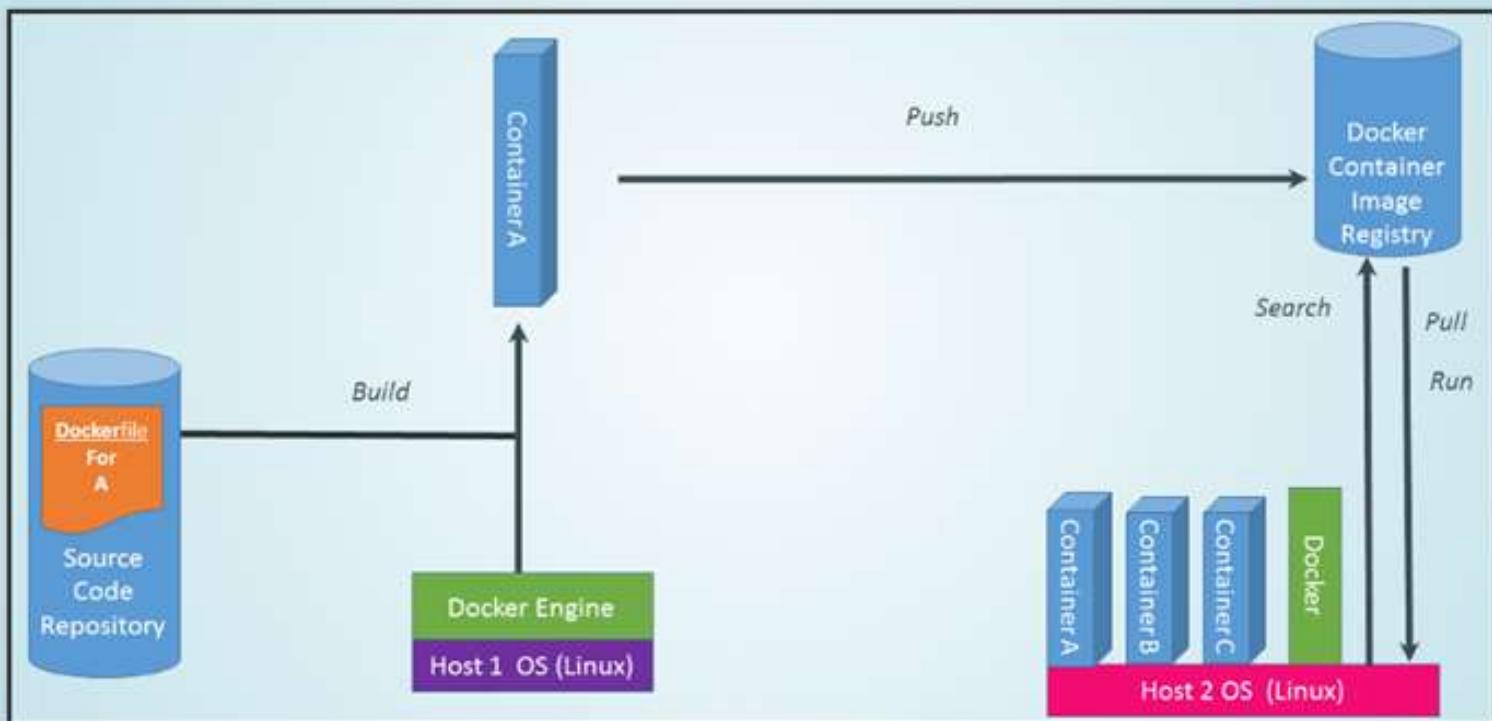
pointful.github.io/docker-intro/#/14

Why are Docker Containers Lightweight?



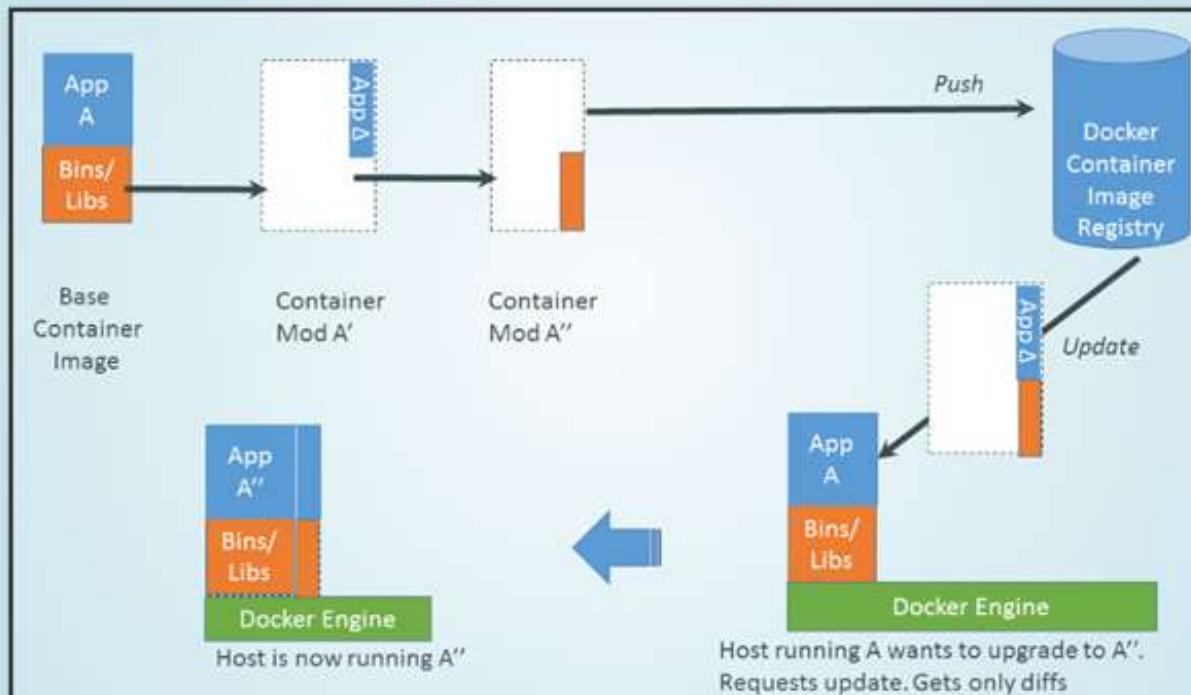
Docker Deployment

What are the Basics of a Docker System?



Docker Changes

Changes and Updates

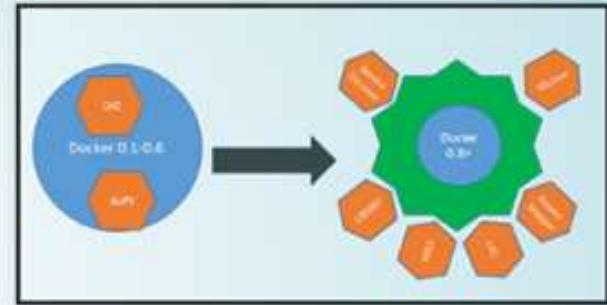


Ecosystem Supports

- Operating systems
 - Virtually any distribution with a 2.6.32+ kernel
 - Red Hat/Docker collaboration to make work across RHEL 6.4+, Fedora, and other members of the family (2.6.32 +)
 - CoreOS—Small core OS purpose built with Docker
- OpenStack
 - Docker integration into NOVA (& compatibility with Glance, Horizon, etc.) accepted for Havana release
- Private PaaS
 - OpenShift, Solum (Rackspace, OpenStack), Other TBA
- Public PaaS
 - Deis, Voxoz, Cocaine (Yandex), Baidu PaaS
- Public IaaS
 - Native support in Rackspace, Digital Ocean,+++
 - AMI (or equivalent) available for AWS & other
- DevOps Tools
 - Integrations with Chef, Puppet, Jenkins, Travis, Salt, Ansible +++
- Orchestration tools
 - Mesos, Heat, ++
 - Shipyard & others purpose built for Docker
- Applications
 - 1000's of Dockerized applications available at index.docker.io

Docker Futures

- Docker 0.7 (current release)
 - Fedora compatibility
 - Reduce kernel dependencies
 - Device mapper
 - Container linking
- Docker 0.8 (Dec)
 - Shrink and stabilize Core
 - Provide stable, pluggable API
 - RHEL compatibility
 - Nested containers
 - Beam: Introspection API based on Redis
 - Expand snapshot management features for data volumes
 - Will consider this "production ready"
- Docker 0.9 (Jan)
- Docker 1.0 (Feb)
 - Will offer support for this product



Dockerfile

It is possible to build your own images reading instructions from a
Dockerfile

```
FROM centos:7

RUN yum install -y python-devel python-virtualenv
RUN virtualenv /opt/indico/venv
RUN pip install indico
COPY entrypoint.sh /opt/indico/entrypoint.sh
EXPOSE 8000
ENTRYPOINT /opt/indico/entrypoint.sh
```

docker-compose

Allows to run multi-container Docker applications reading instructions from a `docker-compose.yml` file

```
version: "2"
services:
  my-application:
    build: .
    ports:
      - "8000:8000"
    environment:
      - CONFIG_FILE
  db:
    image: postgres
  redis:
    image: redis
    command: redis-server --save "" --appendonly no
    ports:
      - "6379"
```

Docker use cases

- Development Environment
- Environments for Integration Tests
- Quick evaluation of software
- Microservices
- Multi-Tenancy
- Unified execution environment (dev ↗ test ↗ prod (local, VM, cloud, ...))

03

How to setup Docker in Linux

How to setup Docker in Linux – Oracle linux 7.x

1. Touch file /etc/yum.repos.d/docker.repo

name=Docker Repository

baseurl=https://yum.dockerproject.org/repo/main/oraclelinux/7

enabled=1

gpgcheck=1

gpgkey=https://yum.dockerproject.org/gpg

2. Yum install -y docker-engine

3. Configure Firewall :

systemctl disable firewalld

yum install -y iptables-services

systemctl enable iptables

systemctl start iptables

4. Automatic start: systemctl enable docker.service

5. Manually start: systemctl start docker.service

6. Check status: systemctl status docker.service



How to setup Docker in Linux – RHEL 7.4 build

```
[root@sgli-ddncaut03a ~]# uname -r  
3.10.0-693.5.2.el7.x86_64  
[root@sgli-ddncaut03a ~]# yum install docker  
Loaded plugins: product-id, search-disabled-repos
```

```
Installed:  
  docker.x86_64 2:1.12.6-61.git85d7426.el7  
  
Dependency Installed:  
  atomic-registries.x86_64 1:1.19.1-5.git48c224b.el7  
  audit-libs-python.x86_64 0:2.7.6-3.el7  
  checkpolicy.x86_64 0:2.5-4.el7  
  container-selinux.noarch 2:2.28-1.git85ce147.el7  
  container-storage-setup.noarch 0:0.7.0-1.git4ca59c5.el7  
  docker-client.x86_64 2:1.12.6-61.git85d7426.el7  
  docker-common.x86_64 2:1.12.6-61.git85d7426.el7  
  docker-rhel-push-plugin.x86_64 2:1.12.6-61.git85d7426.el7  
  json-glib.x86_64 0:1.2.6-1.el7  
  libcgroup.x86_64 0:0.41-13.el7  
  libsemanage-python.x86_64 0:2.5-8.el7  
  oci-register-machine.x86_64 1:0-3.13.gitcd1e331.el7  
  oci-systemd-hook.x86_64 1:0.1.14-1.git1ba44c6.el7  
  oci-umount.x86_64 2:2.0.0-1.git299e781.el7  
  policycoreutils-python.x86_64 0:2.5-17.1.el7  
  python-IPy.noarch 0:0.75-6.el7  
  setools-libs.x86_64 0:3.3.8-1.1.el7  
  skopeo-containers.x86_64 1:0.1.24-1.dev.git28d4e08.el7  
  subscription-manager-plugin-container.x86_64 0:1.19.23-1.el7_4  
  yajl.x86_64 0:2.0.4-4.el7  
  
Complete!  
[root@sgli-ddncaut03a ~]# █
```

How to setup Docker in Linux – RHEL 7.4 build



```
[root@sg1i-ddncaut03a ~]# docker version
Client:
Version:          1.12.6
API version:      1.24
Package version:  docker-1.12.6-61.git85d7426.el7.x86_64
Go version:        go1.8.3
Git commit:       85d7426/1.12.6
Built:            Tue Sep 26 15:30:51 2017
OS/Arch:          linux/amd64

Server:
Version:          1.12.6
API version:      1.24
Package version:  docker-1.12.6-61.git85d7426.el7.x86_64
Go version:        go1.8.3
Git commit:       85d7426/1.12.6
Built:            Tue Sep 26 15:30:51 2017
OS/Arch:          linux/amd64
[root@sg1i-ddncaut03a ~]# █
```

How to setup Docker in Linux – RHEL 7.4 build

```
[root@sgli-ddncaut03a ~]# docker search oracle
INDEX          NAME
docker.io       docker.io/oraclelinux
docker.io       docker.io/frolvlad/alpine-oraclejdk8
docker.io       docker.io/sath89/oracle-12c
docker.io       docker.io/alexeiled/docker-oracle-xe-11g
docker.io       docker.io/sath89/oracle-xe-11g
docker.io       docker.io/jaspeen/oracle-11g
docker.io       docker.io/isuper/java-oracle
docker.io       docker.io/wnameless/oracle-xe-11g
docker.io       docker.io/oracle/glassfish
docker.io       docker.io/oracle/openjdk
docker.io       docker.io/airdock/oracle-jdk
docker.io       docker.io/ingensi/oracle-jdk
docker.io       docker.io/cogniteev/oracle-java
docker.io       docker.io/n3ziniukas/ubuntu-oracle-jdk
docker.io       docker.io/oracle/nosql
docker.io       docker.io/srgio/java-oracle
docker.io       docker.io/andrepbt/oracle-java
docker.io       docker.io/openweb/oracle-tomcat
docker.io       docker.io/oracle/java7
docker.io       docker.io/martinseeler/oracle-server-jre
docker.io       docker.io/davidcaste/debian-oracle-java
docker.io       docker.io/teradatalabs/centos6-java8-oracle
docker.io       docker.io/spansari/nodejs-oracledb
docker.io       docker.io/publicisworldwide/oracle-core
docker.io       docker.io/softwareplant/oracle
[...]
[root@sgli-ddncaut03a ~]# docker search oracle
INDEX          NAME
docker.io       docker.io/oraclelinux
docker.io       docker.io/frolvlad/alpine-oraclejdk8
docker.io       docker.io/sath89/oracle-12c
docker.io       docker.io/alexeiled/docker-oracle-xe-11g
docker.io       docker.io/sath89/oracle-xe-11g
docker.io       docker.io/jaspeen/oracle-11g
docker.io       docker.io/isuper/java-oracle
docker.io       docker.io/wnameless/oracle-xe-11g
docker.io       docker.io/oracle/glassfish
docker.io       docker.io/oracle/openjdk
docker.io       docker.io/airdock/oracle-jdk
docker.io       docker.io/ingensi/oracle-jdk
docker.io       docker.io/cogniteev/oracle-java
docker.io       docker.io/n3ziniukas/ubuntu-oracle-jdk
docker.io       docker.io/oracle/nosql
docker.io       docker.io/srgio/java-oracle
docker.io       docker.io/andrepbt/oracle-java
docker.io       docker.io/openweb/oracle-tomcat
docker.io       docker.io/oracle/java7
docker.io       docker.io/martinseeler/oracle-server-jre
docker.io       docker.io/davidcaste/debian-oracle-java
docker.io       docker.io/teradatalabs/centos6-java8-oracle
docker.io       docker.io/spansari/nodejs-oracledb
docker.io       docker.io/publicisworldwide/oracle-core
docker.io       docker.io/softwareplant/oracle
[...]
[root@sgli-ddncaut03a ~]# docker pull docker.io/oraclelinux
Using default tag: latest
Trying to pull repository docker.io/library/oraclelinux ...
latest: Pulling from docker.io/library/oraclelinux
1b19d6599a70: Downloading [=====>] 82.89 MB/86.06 MB
[...]
[root@sgli-ddncaut03a ~]# docker pull docker.io/oraclelinux
Using default tag: latest
Trying to pull repository docker.io/library/oraclelinux ...
latest: Pulling from docker.io/library/oraclelinux
1b19d6599a70: Pull complete
Digest: sha256:6067eb9ac8edc2042508e2adfd00b9fb1cc1d38e708d0f5f98058a8740ba0661
[root@sgli-ddncaut03a ~]#
```

How to setup Docker in Linux – RHEL 7.4 build

```
[root@sgli-ddncaut03a ~]# docker images
REPOSITORY          TAG      IMAGE ID      CREATED        SIZE
docker.io/oraclelinux latest  af8cf7fc5b7e  2 weeks ago   233.7 MB
[root@sgli-ddncaut03a ~]#
```

```
[root@sgli-ddncaut03a ~]# docker inspect af8cf7fc5b7e
[
  {
    "Id": "sha256:af8cf7fc5b7e9e4dcee6db022f23118d98ff54bfea1458bfb2d2ad7fad61770f",
    "RepoTags": [
      "docker.io/oraclelinux:latest"
    ],
    "RepoDigests": [
      "docker.io/oraclelinux@sha256:6067eb9ac8edc2042508e2adfd00b9fb1cc1d38e708d0f5f98058a8740ba0661"
    ],
    "Parent": "",
    "Comment": "",
    "Created": "2018-04-18T18:40:12.935217168Z",
    "Container": "4b74c7f189be3a3f2b318729e06b963df935520e5544520c2b26c17fddf63f55",
    "ContainerConfig": {
      "Hostname": "4b74c7f189be",
      "Domainname": "",
      "User": "",
      "AttachStdin": false,
      "AttachStdout": false,
      "AttachStderr": false,
      "Tty": false,
      "OpenStdin": false,
      "StdinOnce": false,
      "Env": [
        "PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"
      ],
      "Cmd": [
        "/bin/sh",
        "-c",
        "#(nop) ",
        "CMD [\"/bin/bash\"]"
      ]
    }
  }
]
```



How to setup Docker in Linux – RHEL 7.4 build

```
[root@sgli-ddncaut03a ~]# docker run --help
Usage: docker run [OPTIONS] IMAGE [COMMAND] [ARG...]
Run a command in a new container
```

```
[root@sgli-ddncaut03a ~]# docker run -i -t docker.io/oraclelinux /bin/bash
[root@9524b56297f5 /]# echo "hello docker"
hello docker
[root@9524b56297f5 /]# ls
bin boot dev etc home lib lib64 media mnt opt proc root run sbin srv sys tmp usr var
```

```
[root@sgli-ddncaut03a ~]# docker ps
CONTAINER ID        IMAGE               COMMAND       CREATED          STATUS          PORTS          NAMES
9524b56297f5        docker.io/oraclelinux   "/bin/bash"   About a minute ago   Up About a minute   zen_chandrasekhar
[root@sgli-ddncaut03a ~]# docker ps -a
CONTAINER ID        IMAGE               COMMAND       CREATED          STATUS          PORTS          NAMES
9524b56297f5        docker.io/oraclelinux   "/bin/bash"   About a minute ago   Up About a minute   zen_chandrasekhar
[root@sgli-ddncaut03a ~]#
```

Commands & Reference – Docker Commands

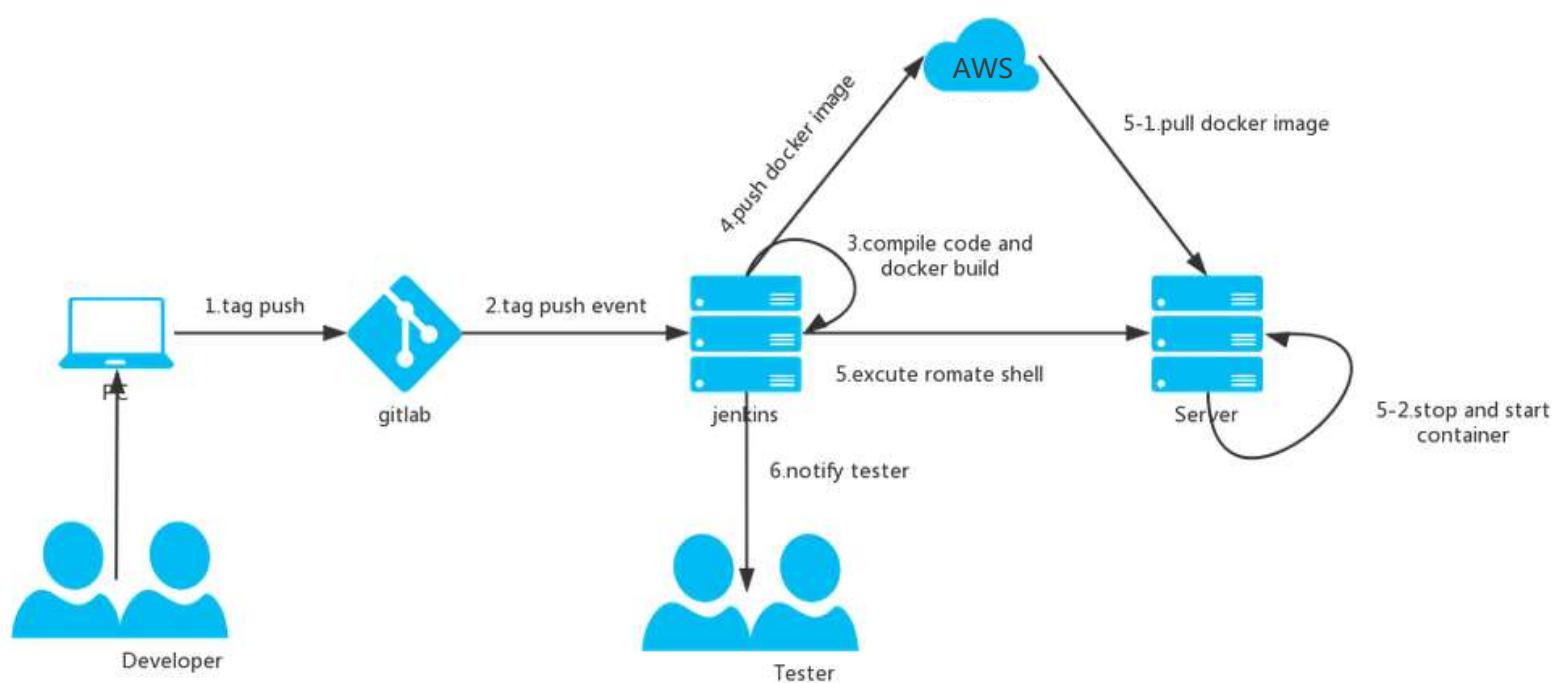
docker run [OPTIONS] IMAGE [COMMAND] [ARG...]	runoob@runoob:~\$ docker run -it nginx:latest /bin/bash
docker start [OPTIONS] CONTAINER [CONTAINER...]	docker start myrunoob
docker stop [OPTIONS] CONTAINER [CONTAINER...]	docker stop myrunoob
docker restart [OPTIONS] CONTAINER [CONTAINER...]	docker restart myrunoob
docker kill [OPTIONS] CONTAINER [CONTAINER...]	runoob@runoob:~\$ docker kill -s KILL mynginx
docker rm [OPTIONS] CONTAINER [CONTAINER...]	docker rm -f db01, db02
docker create [OPTIONS] IMAGE [COMMAND] [ARG...]	runoob@runoob:~\$ docker create --name myrunoob nginx:latest
docker ps [OPTIONS]	runoob@runoob:~\$ docker ps -a -q
docker inspect [OPTIONS] NAME ID [NAME ID...]	runoob@runoob:~\$ docker inspect mysql:5.6
docker top [OPTIONS] CONTAINER [ps OPTIONS]	runoob@runoob:~/mysql\$ docker top mymysql
docker logs [OPTIONS] CONTAINER	runoob@runoob:~\$ docker logs -f mynginx
docker commit [OPTIONS] CONTAINER [REPOSITORY[:TAG]]	runoob@runoob:~\$ docker commit -a "runoob.com" -m "my apache" a404c6c174a2 mymysql:v1
docker diff [OPTIONS] CONTAINER	runoob@runoob:~\$ docker diff mymysql
docker login [OPTIONS] [SERVER]	docker login -u username -p password
docker pull [OPTIONS] NAME[:TAG @DIGEST]	docker pull java
docker push [OPTIONS] NAME[:TAG]	docker push myapache:v1
docker search [OPTIONS] TERM	runoob@runoob:~\$ docker search -s 10 java
docker images [OPTIONS] [REPOSITORY[:TAG]]	runoob@runoob:~\$ docker images
docker build [OPTIONS] PATH URL -	docker build -t runoob/ubuntu:v1 .

Commands & Reference – Dockerfile Commands



FROM	FROM <image>:<tag>
MAINTAINER	MAINTAINER <name>
RUN	RUN <command> (the command is run in a shell - '/bin/sh -c')
CMD	CMD ["executable", "param1", "param2"] (like an exec, this is the preferred form)
ENTRYPOINT	ENTRYPOINT ["executable", "param1", "param2"] (like an exec, the preferred form)
USER	ENTRYPOINT ["memcached", "-u", "daemon"]
EXPOSE	EXPOSE <port> [<port>...]
ENV	ENV <key> <value>
ADD	ADD <src> <dest>
VOLUME	VOLUME ["<mountpoint>"]

Why use Docker – Docker in Devops





Commands & References – References

<http://www.docker.com>

<https://docs.docker.com/linux/>

<https://docs.docker.com/engine/userguide/>

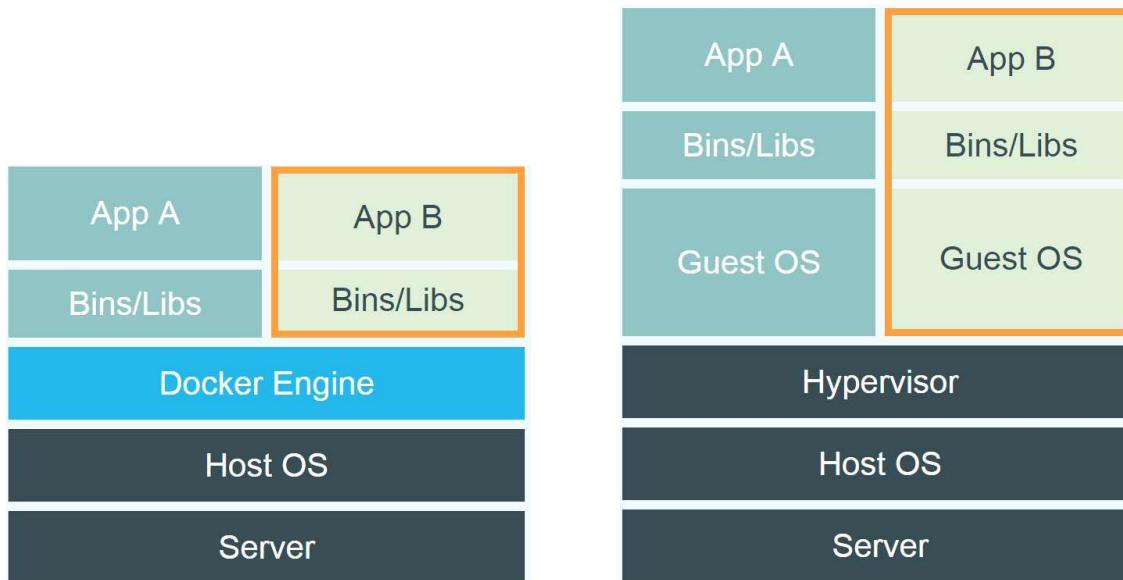
<https://www.docker.com/open-source>

<https://hub.docker.com/>

<https://www.docker-cn.com/>

<https://thehub.thomsonreuters.com/docs/DOC-2572951>

Comparison between LXC/docker and VM



Docker Engine

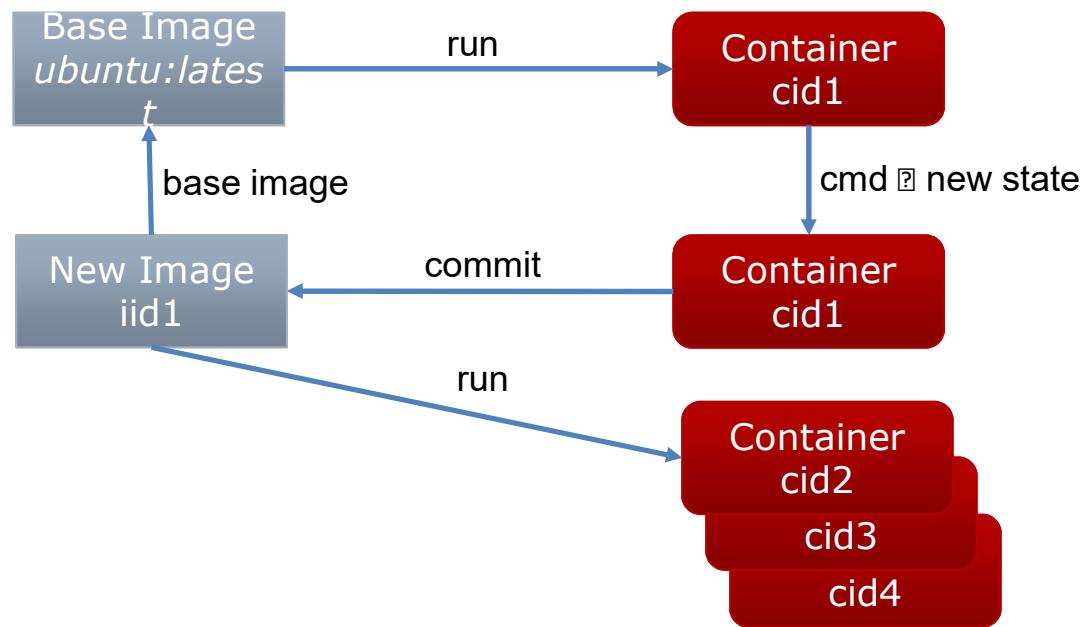
- the Docker engine runs in the background
 - manages containers, images, and builds
 - HTTP API (over UNIX or TCP socket)
 - embedded CLI talking to the API

Building docker image

- With run/commit commands

- 1) docker run ubuntu bash
- 2) apt-get install this and that
- 3) docker commit <containerid> <imagename>
- 4) docker run <imagename> bash
- 5) git clone git://.../mycode
- 6) pip install -r requirements.txt
- 7) docker commit <containerid> <imagename>
- 8) repeat steps 4-7 as necessary
- 9) docker tag <imagename> <user/image>
- 10) docker push <user/image>

Docker



Authoring image with a dockerfile

□ A sample dockerfile

FROM ubuntu

```
RUN apt-get -y update
RUN apt-get install -y g++
RUN apt-get install -y erlang-dev erlang-manpages erlang-base-
hipe ...
RUN apt-get install -y libmozjs185-dev libicu-dev libtool ...
RUN apt-get install -y make wget
RUN wget http://.../apache-couchdb-1.3.1.tar.gz | tar -C /tmp -
zxf-
RUN cd /tmp/apache-couchdb-* && ./configure && make install
RUN printf "[httpd]\nport = 8101\nbind_address = 0.0.0.0" >
/usr/local/etc/couchdb/local.d/docker.ini
```

EXPOSE 8101

CMD ["/usr/local/bin/couchdb"]

Run the command to build:

```
docker build -t your_account/couchdb .
```

Docker Hub

- Public repository of Docker images
 - <https://hub.docker.com/>
 - docker search [term]
- Automated: Has been automatically built from Dockerfile
 - Source for build is available on GitHub

Dev-> test->production

- code in local environment
(« dockerized » or not)
- each push to the git repo triggers a hook
- the hook tells a build server to clone the code and run « docker build » (using the Dockerfile)
- the containers are tested (nosetests, Jenkins...),
and if the tests pass, pushed to the registry
- production servers pull the containers and run them
- for network services, load balancers are updated

Docker has a repository like github

you can push and pull container images to/from
the Docker registry

which is something like a “GitHub” for Docker
container images.

Docker has a repository like github

you can push and pull container images to/from
the Docker registry

which is something like a “GitHub” for Docker
container images.

Multitenancy – Introduction

- Multi-tenancy is an architecture in which a single instance of a software application serves multiple customers. Each customer is called a tenant. Tenants may be given the ability to customize some parts of the application, such as color of the user interface (UI) or business rules, but they cannot customize the application's code.
- A software-as-a-service ([SaaS](#)) provider, for example, can run one instance of its application on one instance of a database and provide web access to multiple customers. In such a scenario, each tenant's data is isolated and remains invisible to other tenants.

Multitenancy – Introduction

- Multi-tenancy is an architectural pattern
- A single instance of the software is run on the service provider's infrastructure
- Multiple tenants access the same instance.
- In contrast to the multi-user model, multi-tenancy requires customizing the single instance according to the multi-faceted requirements of many tenants.

Multitenancy – key aspects

A Multi-tenants application lets customers (tenants) share the same hardware resources, by offering them one shared application and database instance ,while allowing them to configure the application to fit there needs as if it runs on dedicated environment.

These definition focus on what we believe to be the key aspects of multi tenancy:

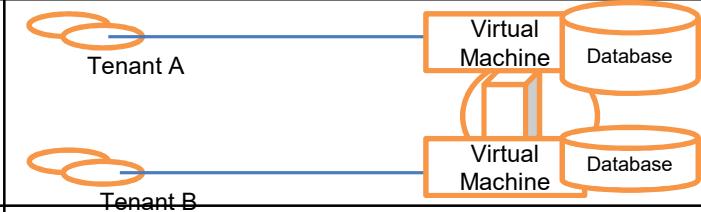
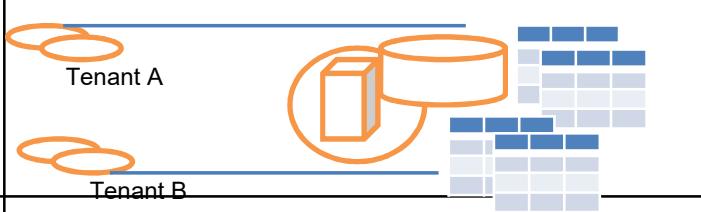
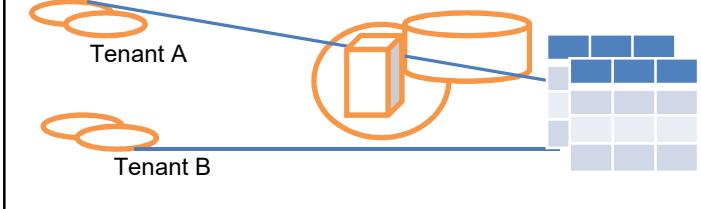
- 1.The ability of the application to share hardware resources.
- 2.The offering of a high degree of configurability of the software.
- 3.The architectural approach in which the tenants make use of a single application and database instance.

Multi-tenants Deployment Modes for Application Server

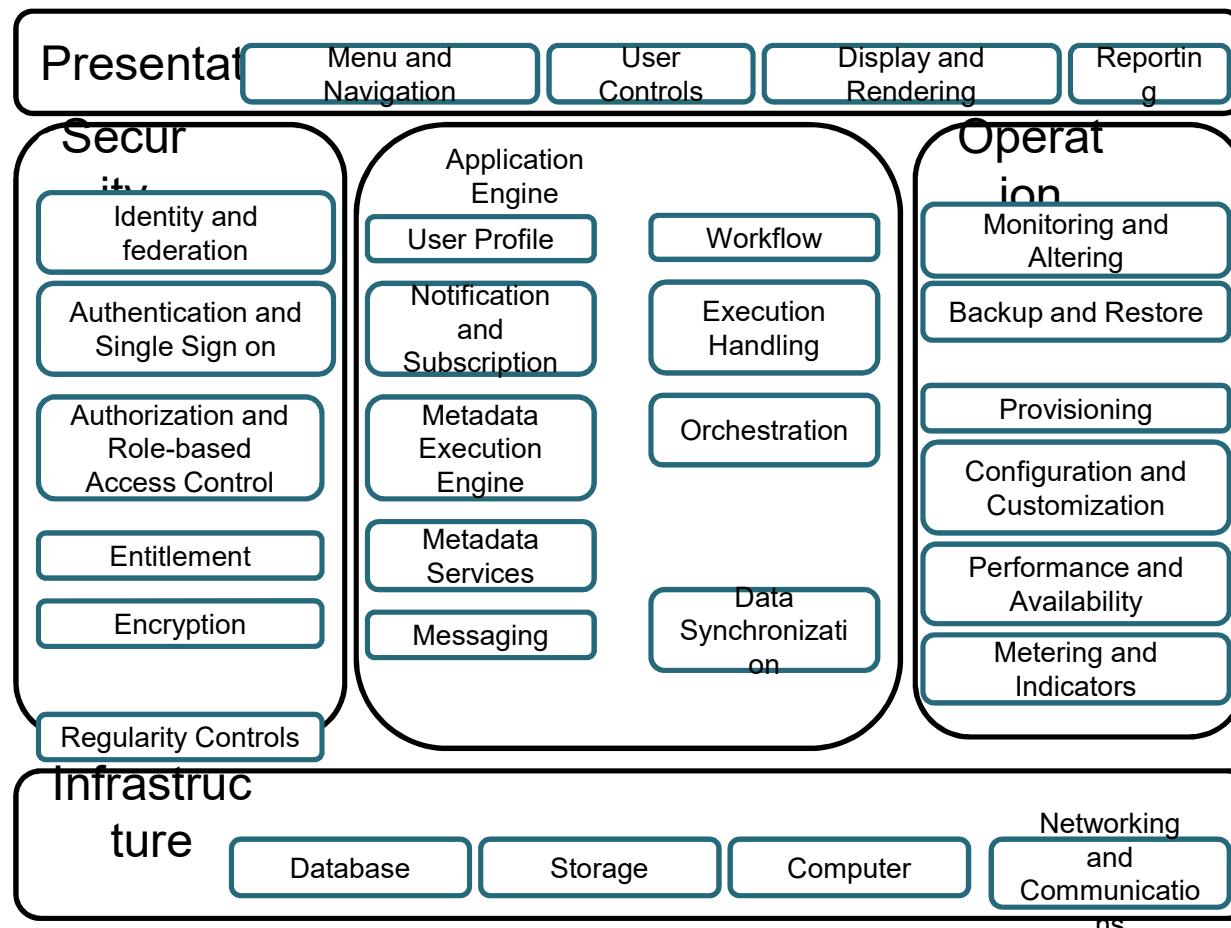
Fully isolated Application server Each tenant accesses an application server running on a dedicated servers.	<p>The diagram shows two separate application servers. The top server is labeled 'Application Server' and has two blue circles below it labeled 'Tenant A'. The bottom server is also labeled 'Application Server' and has two blue circles below it labeled 'Tenant B'. Lines connect each tenant to its respective application server.</p>
Virtualized Application Server Each tenant accesses a dedicated application running on a separate virtual machine.	<p>The diagram shows two separate virtual machines. Each virtual machine contains an 'Application server' box. Tenant A is connected to the top application server in the first VM, and Tenant B is connected to the top application server in the second VM. Both VMs also have 'Virtual machine' boxes at the bottom.</p>
Shared Virtual Server Each tenant accesses a dedicated application server running on a shared virtual machine.	<p>The diagram shows a single virtual machine containing multiple application servers. Tenant A is connected to the top application server in the VM, and Tenant B is connected to the middle application server in the VM. The VM is labeled 'Virtual machine'.</p>
Shared Application Server The tenant shared the application server and access application resources through separate session or threads.	<p>The diagram shows a single application server box. Tenant A is connected to the top session thread, and Tenant B is connected to the middle session thread. The application server box is labeled 'Session Thread'.</p>

75

Multi-tenants Deployment Modes in Data Centers

Fully isolated data center The tenants do not share any data center resources	
Virtualized servers The tenants share the same host but access different databases running on separate virtual machines	
Shared Server The tenants share the same server (Hostname or IP) but access different databases	
Shared Database The tenants share the same server and database (shared or different ports) but access different schema (tables)	
Shared Schema The tenants share the same server, database and schema (tables). The irrespective data is segregated by key and rows.	

Conceptual framework of Software as a Service



77

Docker needs containers

