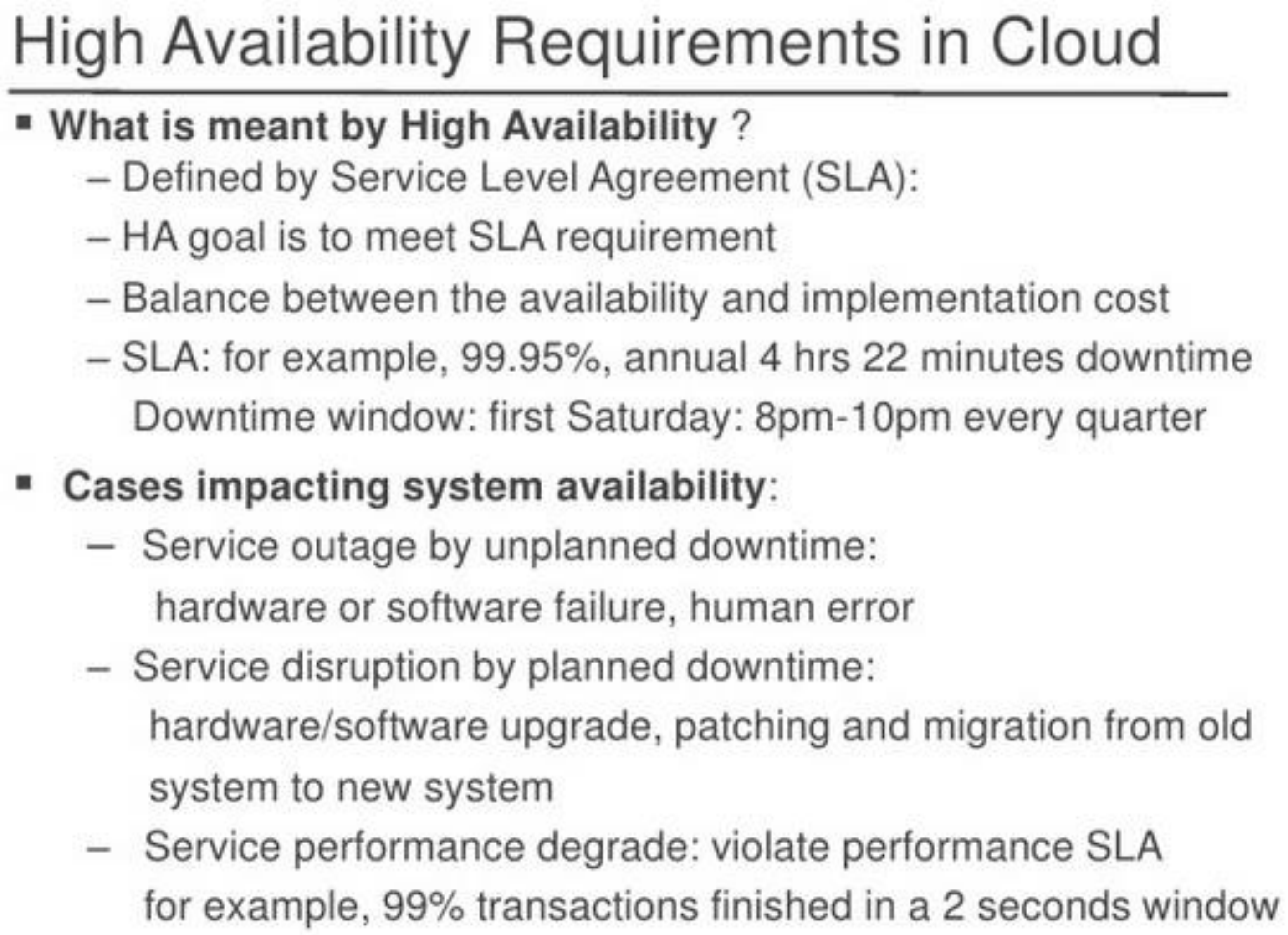
BITS Pilani

Cloud Computing

SEWP ZG527

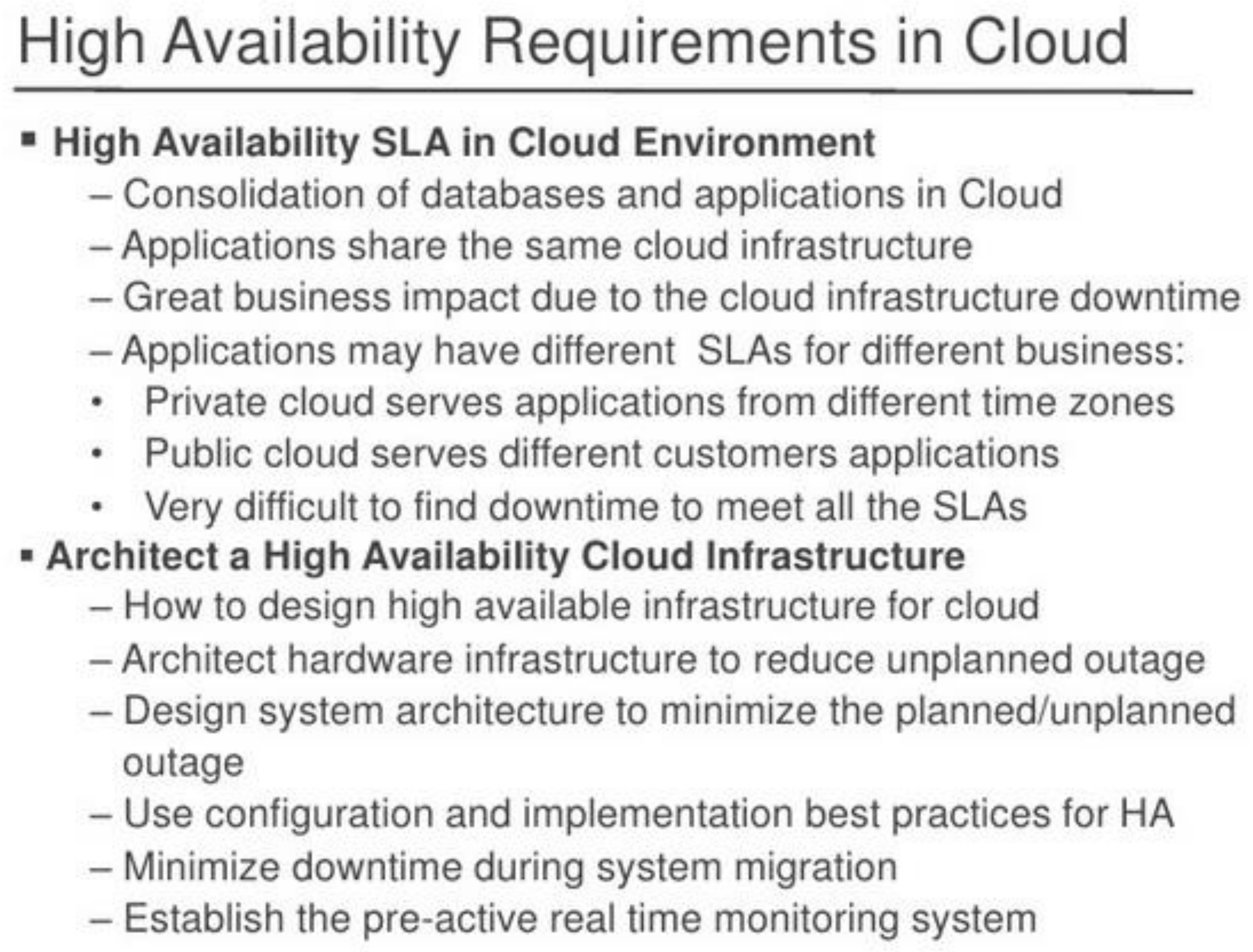
High Availability



High Availability Requirements in Cloud

- **What is meant by High Availability** ?
  - Defined by Service Level Agreement (SLA):
  - HA goal is to meet SLA requirement
  - Balance between the availability and implementation cost
  - SLA: for example, 99.95%, annual 4 hrs 22 minutes downtime
    Downtime window: first Saturday: 8pm-10pm every quarter
- **Cases impacting system availability**:
  - Service outage by unplanned downtime:
    hardware or software failure, human error
  - Service disruption by planned downtime:
    hardware/software upgrade, patching and migration from old system to new system
  - Service performance degrade: violate performance SLA
    for example, 99% transactions finished in a 2 seconds window

High Availability



High Availability Requirements in Cloud

- **High Availability SLA in Cloud Environment**
  - Consolidation of databases and applications in Cloud
  - Applications share the same cloud infrastructure
  - Great business impact due to the cloud infrastructure downtime
  - Applications may have different SLAs for different business:
    - Private cloud serves applications from different time zones
    - Public cloud serves different customers applications
    - Very difficult to find downtime to meet all the SLAs
- **Architect a High Availability Cloud Infrastructure**
  - How to design high available infrastructure for cloud
  - Architect hardware infrastructure to reduce unplanned outage
  - Design system architecture to minimize the planned/unplanned outage
  - Use configuration and implementation best practices for HA
  - Minimize downtime during system migration
  - Establish the pre-active real time monitoring system

High Availability

Steps to achieve high availabilty

Build for server failure

Build for zone failure

Build for Cloud failure

Automating and testing

Key aspects of SLA

In the early days of web-application deployment, **performance of the application** at peak load was a single important criterion for provisioning server resources.

Provisioning in those days involved deciding hardware configuration, determining the number of physical machines, and acquiring them upfront so that the overall business objectives could be achieved.

The web applications were **hosted** on these dedicated individual servers within enterprises' own server rooms. These web applications were used to provide different kinds of e-services to various clients.

Key aspects of SLA

Typically, the **service-level objectives** (SLOs) for these applications were response time and throughput of the application end-user requests.

The capacity buildup was to cater to the estimated peak load experienced by the application. The activity of determining the number of servers and their capacity that could satisfactorily serve the application end-user requests at peak loads is called capacity planning

Key aspects of SLA

Due to the increasing **complexity of managing the hugh Data centres**, enterprises started outsourcing the application hosting to the infrastructure providers. They would procure the hardware and make it available for application hosting.

It necessitated the enterprises to enter into a **legal agreement with the infrastructure service providers** to guarantee a minimum quality of service (QoS).

Typically, the **QoS parameters** are related to the availability of the system CPU, data storage, and network for efficient execution of the application at peak loads.

This legal agreement is known as the service-level agreement (SLA)

Key aspects of SLA

For example, one SLA may state that the application's server machine will be available for 99.9% of the key business hours of the application's end users, also called core time, and 85% of the non-core time.

Another SLA may state that the service provider would respond to a reported issue in less than 10 minutes during the core time, but would respond in one hour during non-core time.

These SLAs are known as the infrastructure SLAs, and the infrastructure service providers are known as **Application Service Providers (ASPs)**

Key aspects of SLA

The dedicated hosting practice resulted in **massive redundancies** within the ASP’s data centers **due to the underutilization of many of their servers**. This is because the applications were not fully utilizing their servers’ capacity at nonpeak loads.

To **reduce the redundancies** and increase the server utilization in data centers, ASPs started co-hosting applications with complementary workload patterns.

Co-hosting of applications means **deploying more than one application on a single server**. This led to further cost advantage for both the ASPs and enterprises.

Key aspects of SLA

However, newer challenges such as application performance isolation and **security guarantees** emerged and needed to be addressed. Performance isolation implies that one application should not steal the resources being utilized by other co-located applications.

Hence, appropriate measures are needed to **guarantee security and performance isolation**. These challenges prevented ASPs from fully realizing the benefits of co-hosting.

Virtualization technologies have been proposed to overcome the above challenges. The ASPs could exploit the containerization features of virtualization technologies to provide performance isolation and guarantee data security to different co-hosted applications

Key aspects of SLA

Adoption of virtualization technologies required ASPs to get more detailed insight into the application runtime characteristics with high accuracy. Based on these characteristics, **ASPs can allocate system resources more efficiently to these applications on-demand**, so that application-level metrics can be monitored and met effectively.

These metrics are request rates and response times. Therefore, different SLAs than the infrastructure SLAs are required. These SLAs are called application SLAs. These service providers are known as **Managed Service Providers (MSP)** because the service providers were responsible for managing the application availability too.

Key aspects of SLA

To **fulfill the SLOs** mentioned in the application SLA and also make their IT infrastructure elastic, an in-depth understanding of the application’s behavior is required for the MSPs. Elasticity implies progressively scaling up the IT infrastructure to take the increasing load of an application. The customer is billed based on their application usage of infrastructure resources for a given period only. The infrastructure can be augmented by procuring resources dynamically from multiple sources, including other MSPs, if resources are scarce at their data centers. This kind of new hosting infrastructure is called cloud platform.

Key aspects of SLA

Traditionally, load balancing techniques and admission control mechanisms have been used to **provide guaranteed quality of service (QoS)** for hosted web applications. These mechanisms can be viewed as the first attempt towards managing the SLOs.

Now it is also possible for a **customer and the service provider to mutually agree upon a set of SLAs** with different performance and cost structure rather than a single SLA. The customer has the flexibility to choose any of the agreed SLAs from the available offerings. At runtime, the customer can switch between the different SLAs.

Key aspects of SLA

Key Components of a Service-Level Agreement

Service Level Parameter

Describes an observable property of a service whose value is measurable. **Metrics** These are definitions of values of service properties that are measured from a service providing system or computed from other metrics and constants.

Metrics are the key instrument to describe exactly what SLA parameters mean by specifying how to measure or compute the parameter values.

Function A function specifies how to compute a metric’s value from the values of other metrics and constants. Functions are central to describing exactly how SLA parameters are computed from resource metrics.

Measurement directives These specify how to measure a metric.

Key aspects of SLA

Key Contractual Elements of an Infrastructural SLA Hardware availability 99% uptime in a calendar month Power availability 99.99% of the time in a calendar month

Data center network availability 99.99% of the time in a calendar month

Backbone network availability 99.999% of the time in a calendar month

Service credit for unavailability Refund of service credit prorated on downtime period

Outage notification guarantee Notification of customer within 1 hr of complete downtime

Internet latency guarantee When latency is measured at 5 min intervals to an upstream provider, the average doesn't exceed 60 msec

Packet loss guarantee Shall not exceed 1% in a calendar month

Key aspects of SLA

Key contractual components of an application SLA

Service level parameter metric Web site response time (e.g., max of 3.5 sec per user request)

Latency of web server (WS) (e.g., max of 0.2 sec per request)

Latency of DB (e.g., max of 0.5 sec per query)Function

Average latency of WS (latency of web server 1+latency of web server 2) /2

Web site response time Average latency of web server+ latency of database Measurement directive

DB latency available via http://mgmtserver/em/latency WS latency available via http://mgmtserver/ws/instanceno/ latency

Service level objective Service assurance

web site latency , 1 sec when concurrent connection , 1000

Penalty 1000 USD for every minute while the SLO was breached

Key aspects of SLA

Each SLA goes through a sequence of steps starting from identification of terms and conditions, activation and monitoring of the stated terms and conditions, and eventual termination of contract once the hosting relationship ceases to exist.

Such a sequence of steps is called **SLA life cycle** and consists of the following five phases:

1. Contract definition
2. Publishing and discovery
3. Negotiation
4. Operationalization
5. De-commissioning

Key aspects of SLA

.Some of the parameters

The SLA class (Platinum, Gold, Silver, etc.) to which the application belongs to.

The amount of penalty associated with SLA breach. Whether the application is at the threshold of breaching the SLA.

Whether the application has already breached the SLA.

The number of applications belonging to the same customer that has breached SLA.

The number of applications belonging to the same customer about to breach SLA

. The type of action to be performed to rectify the situation.

BITS Pilani

Cloud Computing

SEWP ZG527

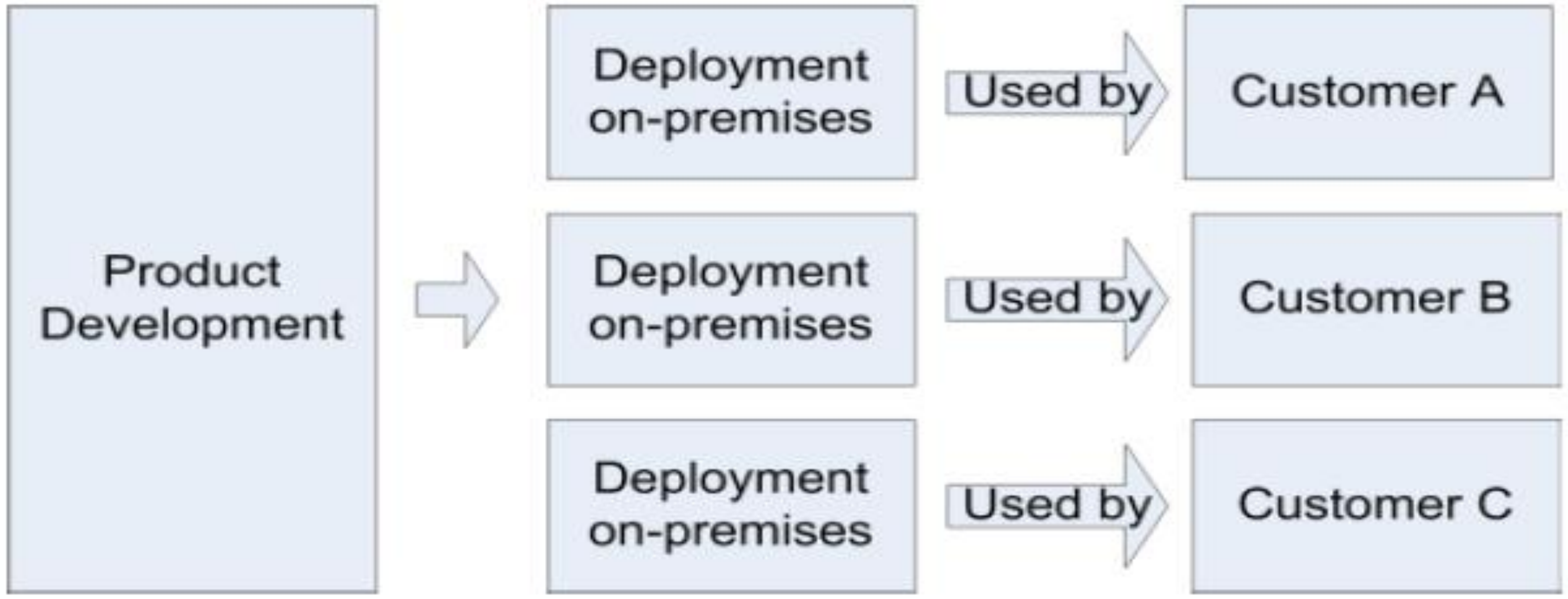
Multitenancy – What is it?

Pros and Cons

	House	Apartment
Effective use of land	**-**	**+**
Privacy	**+**	**-**
Infrastructure sharing	**-**	**+**
Maintenance cost sharing	**-**	**+**
Freedom	**+**	**-**

House:	Privacy and freedom
Apartment:	Cost efficiency

Traditional Deployment Model



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.

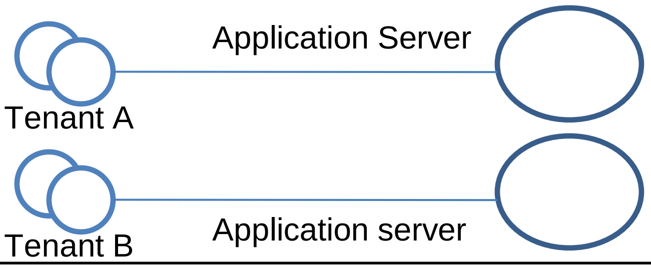
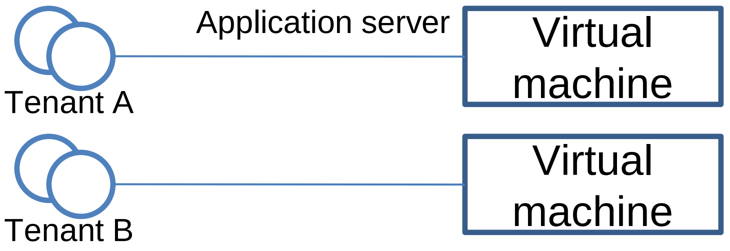
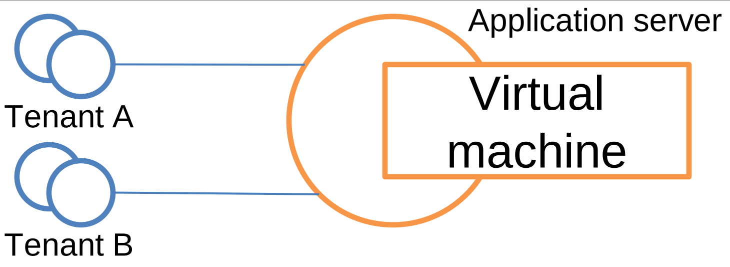
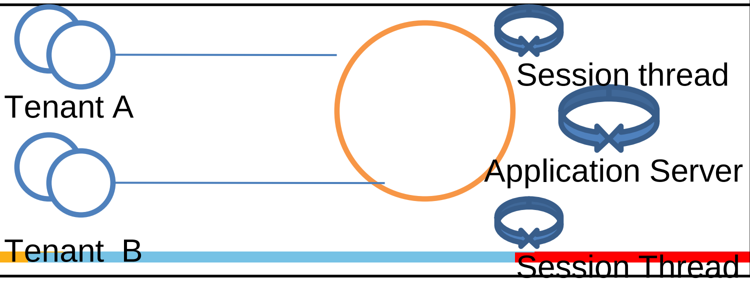
Multitenancy – key aspects

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.

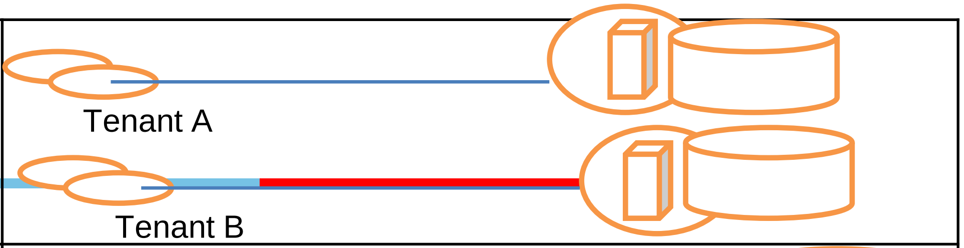
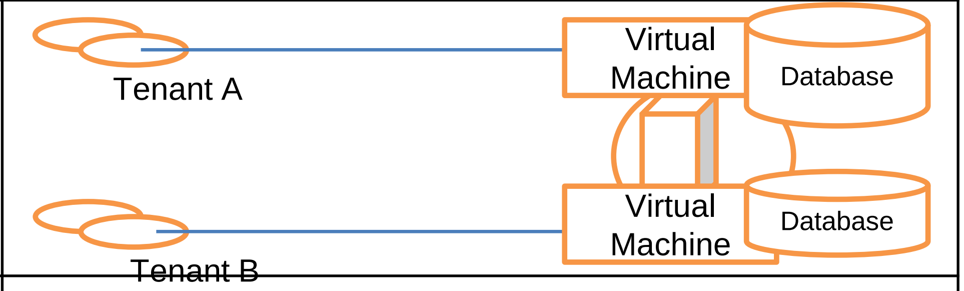
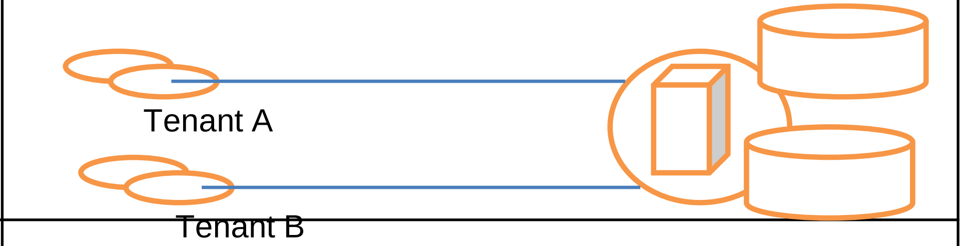
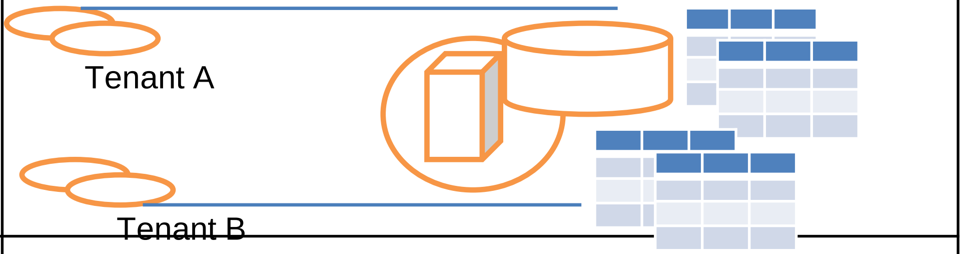
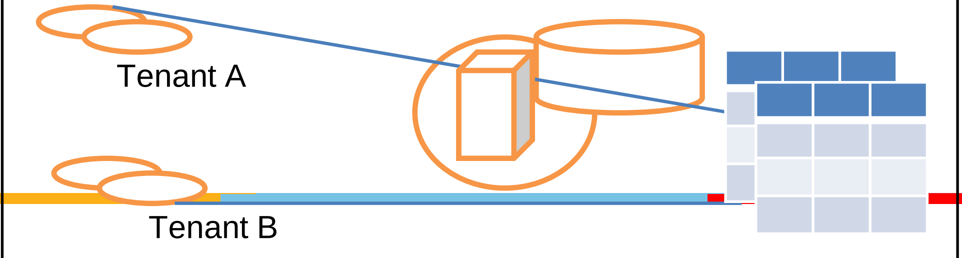
These definition focus on what we believe to be the key aspects of multi tenancy:

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.

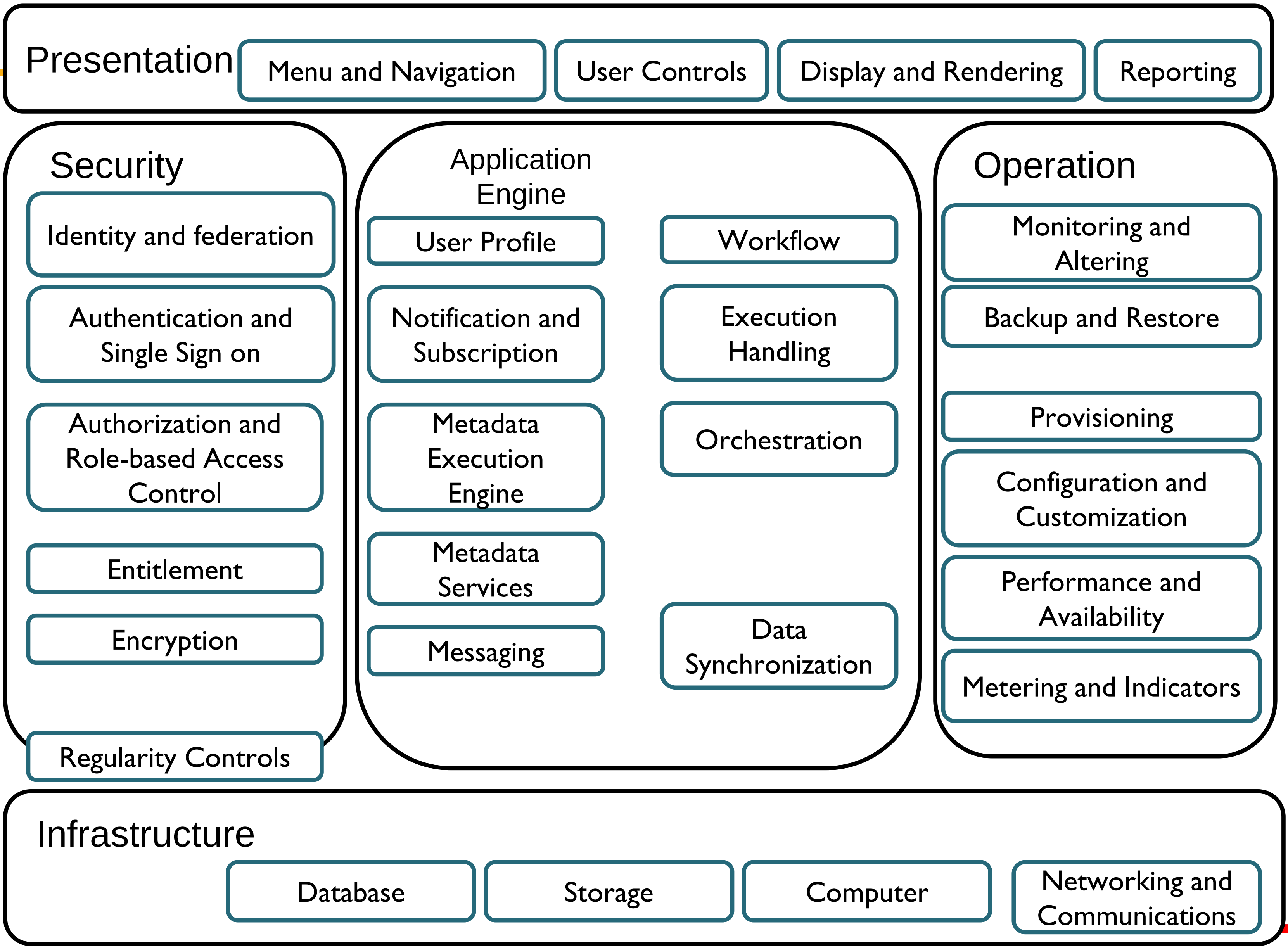
Multi-tenants Deployment Modes for Application Server

Fully isolated Application server Each tenant accesses an application server running on a dedicated servers.	 Application Server Tenant A Tenant B Application server
Virtualized Application Server Each tenant accesses a dedicated application running on a separate virtual machine.	 Application server Virtual machine Tenant A Virtual machine Tenant B
Shared Virtual Server Each tenant accesses a dedicated application server running on a shared virtual machine.	 Application server Virtual machine Tenant A Tenant B
Shared Application Server The tenant shared the application server and access application resources through separate session or threads.	 Session thread Tenant A Application Server Tenant B Session Thread

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**	 Tenant A Virtual Machine Database Virtual Machine Database Tenant B
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



Thank you

BITS Pilani

Cloud Computing

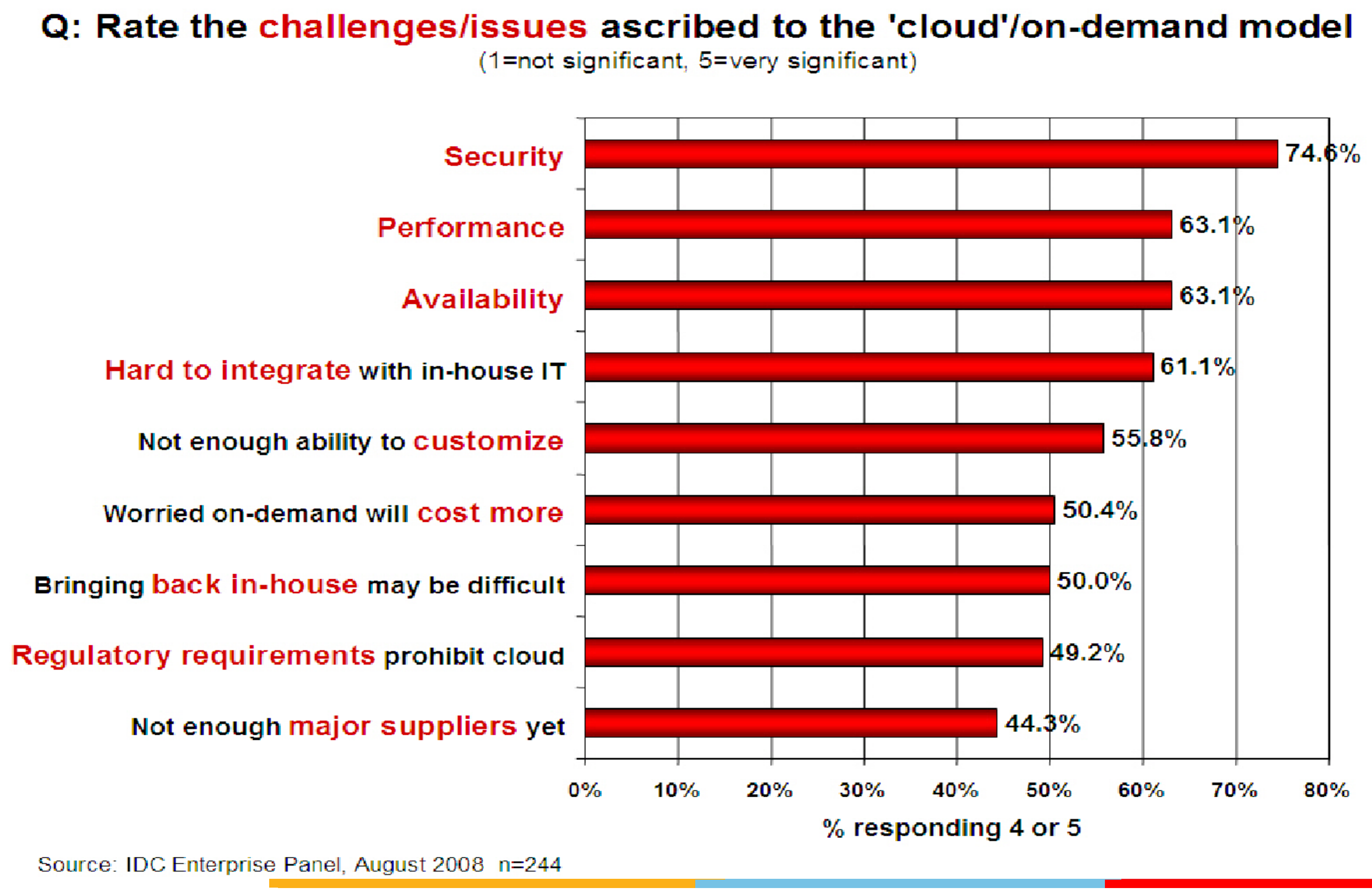
SEWP ZG527

Introduction to cloud security

If cloud computing is so great, why isn't everyone doing it?

- The cloud acts as a big black box, nothing inside the cloud is visible to the clients
- Clients have no idea or control over what happens inside a cloud
- Even if the cloud provider is honest, it can have malicious system admins who can tamper with the VMs and violate confidentiality and integrity
- Clouds are still subject to traditional data confidentiality, integrity, availability, and privacy issues, plus some additional attacks

Companies are still afraid to use clouds



Cloud Security Issues

- Most security problems stem from:
  - Loss of Control
    - Take back control
      - Data and apps may still need to be on the cloud
      - But can they be managed in some way by the consumer?
  - Lack of trust
    - Increase trust (mechanisms)
      - Technology
      - Policy, regulation
      - Contracts (incentives): topic of a future talk
  - Multi-tenancy
    - Private cloud
      - Takes away the reasons to use a cloud in the first place
    - VPC: its still not a separate system
    - Strong separation
- These problems exist mainly in 3rd party management models
  - Self-managed clouds still have security issues, but not related to above

Loss of Control in the Cloud

Consumer's loss of control

- Data, applications, resources are located with provider
- User identity management is handled by the cloud
- User access control rules, security policies and enforcement are managed by the cloud provider
- Consumer relies on provider to ensure
  - Data security and privacy
  - Resource availability
  - Monitoring and repairing of services/resources

Multi-tenancy Issues in the Cloud

- Conflict between tenants' opposing goals
  - Tenants share a pool of resources and have opposing goals
- How does multi-tenancy deal with conflict of interest?
  - Can tenants get along together and 'play nicely' ?
  - If they can't, can we isolate them?
- How to provide separation between tenants?
- Cloud Computing brings new threats

Multiple independent users share the same physical infrastructure

Thus an attacker can legitimately be in the same physical machine as the target

Taxonomy of Fear

- Confidentiality
  - Fear of loss of control over data
    - Will the sensitive data stored on a cloud remain confidential?
    - Will cloud compromises leak confidential client data
  - Will the cloud provider itself be honest and won’t peek into the data?
- Integrity
  - How do I know that the cloud provider is doing the computations correctly?
  - How do I ensure that the cloud provider really stored my data without tampering with it?

Taxonomy of Fear

Availability

- Will critical systems go down at the client, if the provider is attacked in a Denial of Service attack?
- What happens if cloud provider goes out of business?
- Would cloud scale well-enough?
- Often-voiced concern
  – Although cloud providers argue their downtime compares well with cloud user's own data centers

Taxonomy of Fear

- Privacy issues raised via massive data mining
  - Cloud now stores data from a lot of clients, and can run data mining algorithms to get large amounts of information on clients
- Increased attack surface
  - Entity outside the organization now stores and computes data, and so
  - Attackers can now target the communication link between cloud provider and client
  - Cloud provider employees can be phished

Taxonomy of Fear

- Audit-ability and forensics (out of control of data)
  - Difficult to audit data held outside organisation in a cloud
  - Forensics also made difficult since now clients don’t maintain data locally
- Legal quagmire and transitive trust issues
  - Who is responsible for complying with regulations?
  - e.g., SOX, HIPAA, GLBA ?
  - If cloud provider subcontracts to third party clouds, will the data still be secure?

Threat Model

- A threat model helps in analysing a security problem, design mitigation strategies, and evaluate solutions
- Steps:
  - Identify attackers, assets, threats and other components
  - Rank the threats
  - Choose mitigation strategies
  - Build solutions based on the strategies

Threat Model

- Basic components
  - Attacker modelling
    - Choose what attacker to consider
      - insider vs. outsider?
      - single vs. collaborator?
    - Attacker motivation and capabilities
  - Attacker goals
  - Vulnerabilities / threats

Thank you