Virtualization and Multi-tenancy

Week 6

School of Software

Faculty of Engineering and Information Technology

University of Technology, Sydney

SCHOOL OF SOFTWARE

Learning Objectives

- Understand Virtualization
- Types of Virtual Machines
- Cloud Computing and Virtualization
- Applications of Virtualization in Enterprises
- Multi-tenancy

Why virtualization?

(Case Study) Web Server A belonging to Company B has the following utilization profile:

- It is being used by the employees of Company B around 50% of the time (on an average), per day (over a 24 hour time period).
- The rest of the time it is lying idle (and is not being utilized)

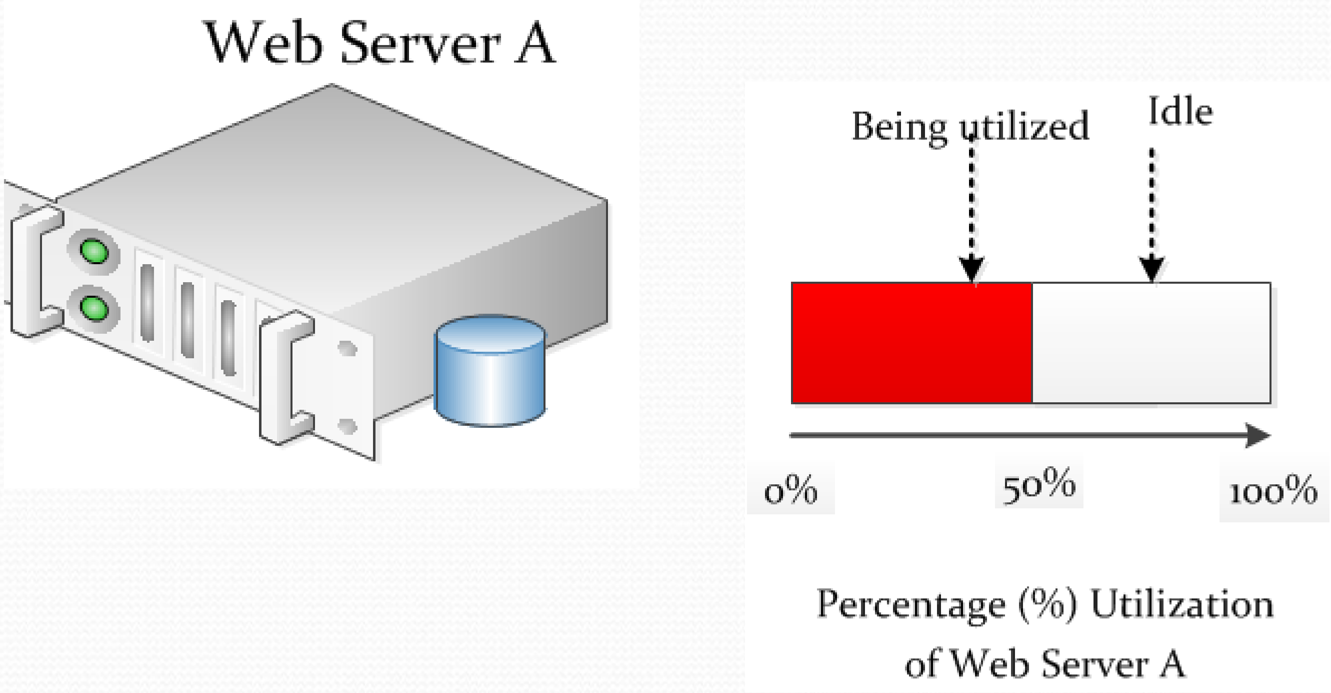


Figure : Utilization Profile of Web Server A

Virtualization Explained

(Case Study Cont….) Company B strategizes to make available the unused capacity of Web Server A (50%) to other consumers.

- The technology which enables Company B to divide the Web Server A, into two logical units; and provide each part to different consumers is "*Virtualization*".
  - The Web Server A is divided "*virtually*" into two web servers. Each of these logically divided web server is termed as "*virtual server*" or "*virtual machine (VM)*".

- A VM is slice of the physical machine implemented by software
- The user of each virtual machine is able to use it and carry out the same activities, as they would, if they were working on the physical web server (Web Server A).

Virtualization Explained

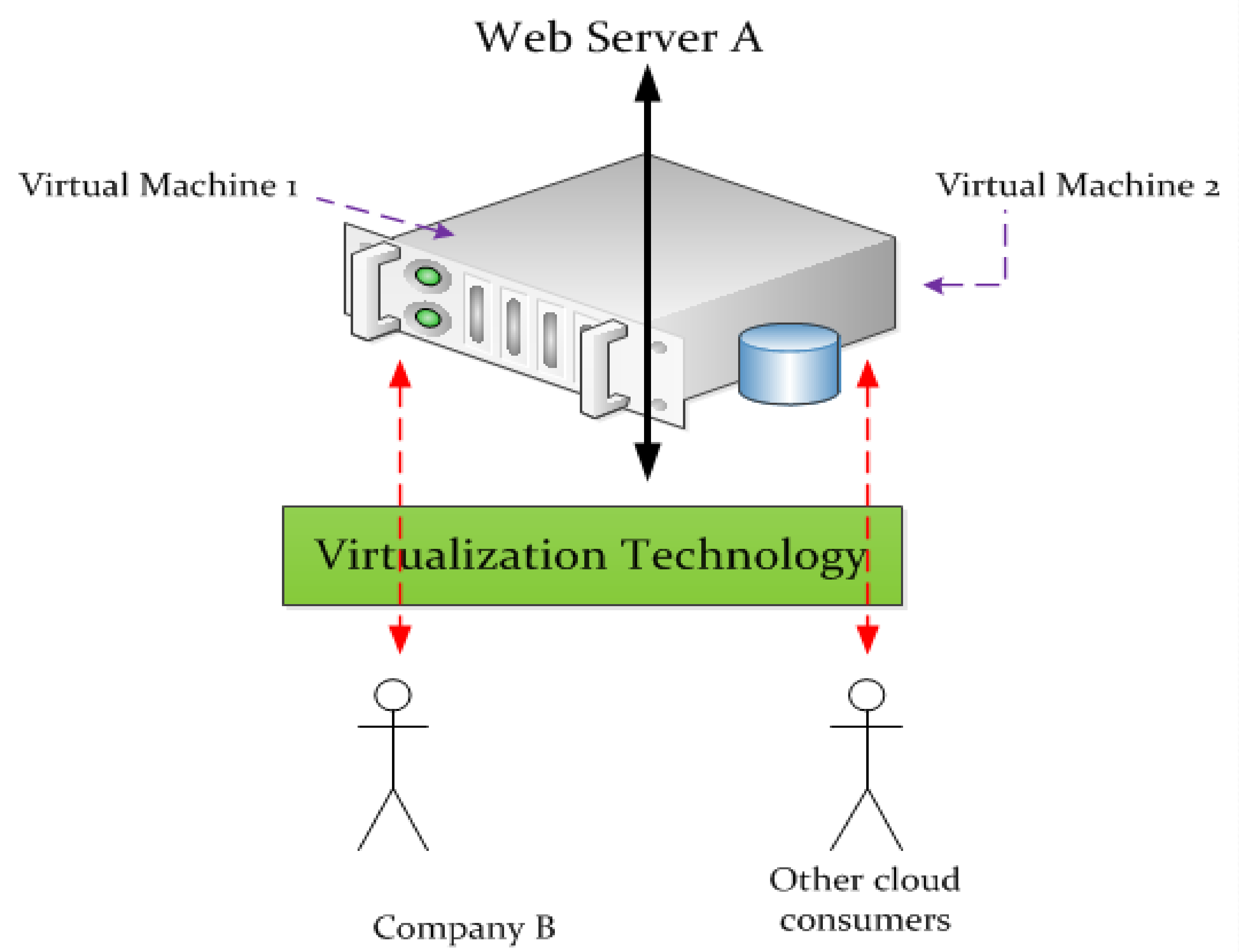


Figure : Pictorial representation of the “logical” division of Web Server A

Background on virtualization

- Virtualization dates back to the era of early “mainframes” (during 1960’s).

- Very widely used in Grid Computing to enable higher utilization of the computing resources (during late 1990’s)

- Virtualization is applicable to both hardware and software

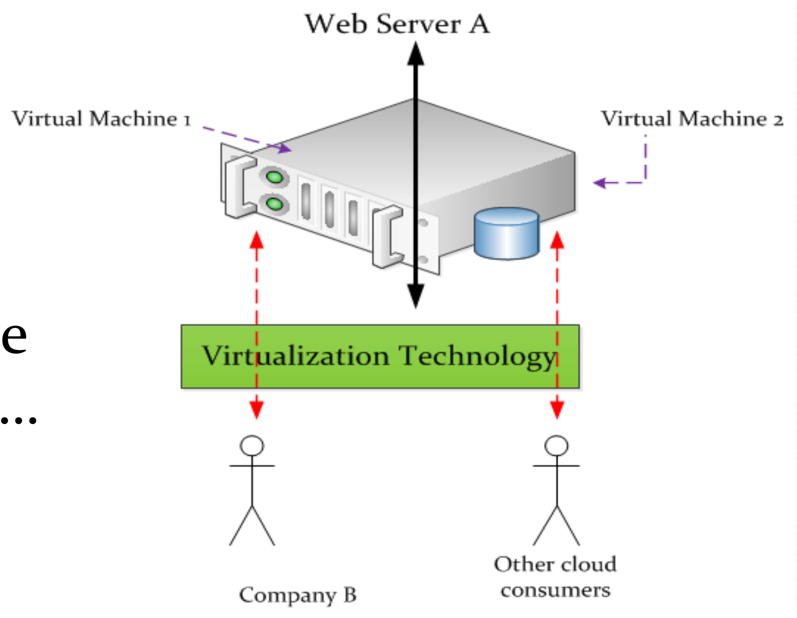
- In the context of computing hardware *virtualization* means creating a virtual version of ….
  - Web Server → (“Virtual” Servers) or (“Virtual” machines)
  - Memory → (“Virtual” Memory)

What is Virtualization?

- Definition of Virtualization:

  “Any means by which *many different users are able to simultaneously interact with, a single computing system*, while *each perceiving that they have an entire virtual instance such as “virtual machine” to themselves*, is a form of virtualization.” (Shroff 2010)

- Technology that enables sharing of computing resources such as:
  - Hardware resources such as physical servers, memory etc…
  - Software resources such as software frameworks, database engines etc.…



Virtual Machines

- One of the most common forms of virtualization in computing hardware

- Broadly, virtual machines are of three types:
  - Process Virtual Machine [Abbreviation: *PVM*]
  - System Virtual Machine (Host) [Abbreviation: *SVM- Host*]
  - System Virtual Machine (Native) [Abbreviation: *SVM- Native*]

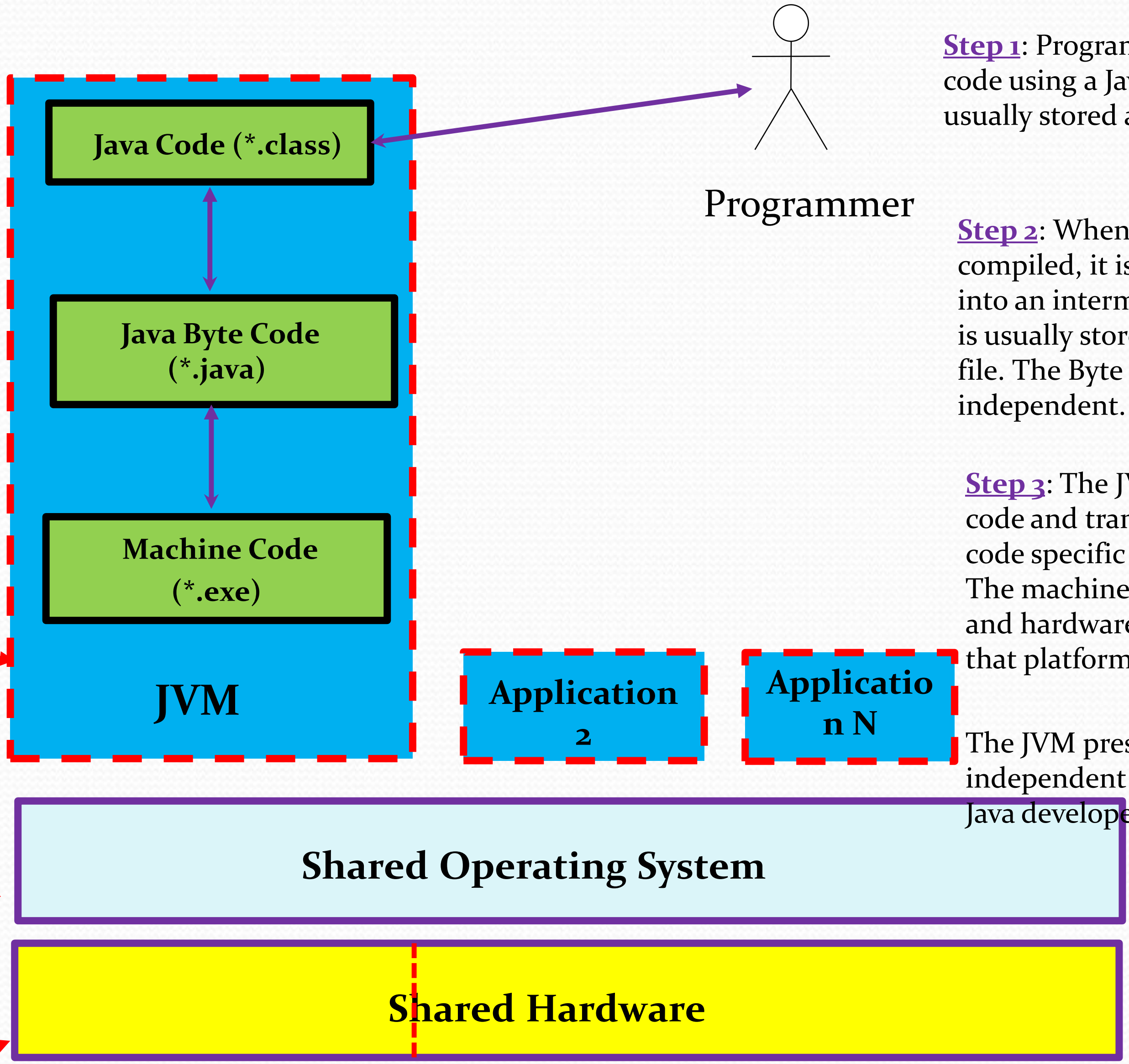
Process Virtual Machine (PVM)

- PVM provides a platform for the execution of a single program (Process)

- Java Virtual machine (JVM):
  - One of the most common forms of PVM;
  - Runs as a process on underlying shared hardware and OS;
  - Creates and uses a logically separate slice of resources such as RAM memory for itself.

- Working of Java Virtual Machine (JVM):
  - Java programs are compiled to byte code (unlike that of other programming languages such as C or C++ which are directly compiled to machine code);
  - Byte code can be executed on top of any JVM;
  - The JVM has access to a set of OS calls and hardware instructions to translate the byte code to machine language instructions.

Illustration of working of JVM



Step 1: Programmer write Java code using a Java compiler . It is usually stored as a *.class file

Step 2: When a Java program is compiled, it is first converted into an intermediate byte code. It is usually stored as a *.java or *.jar file. The Byte code is platform independent.

Step 3: The JVM takes the byte code and translates it to machine code specific for that platform. The machine code includes OS and hardware calls specific for that platform (as an *.exe file).

The JVM presents a platform independent environment for Java developers.

Java Virtual Machine runs on shared OS and hardware as an application or a process

Shared OS amongst all the applications

Shared hardware amongst all the applications

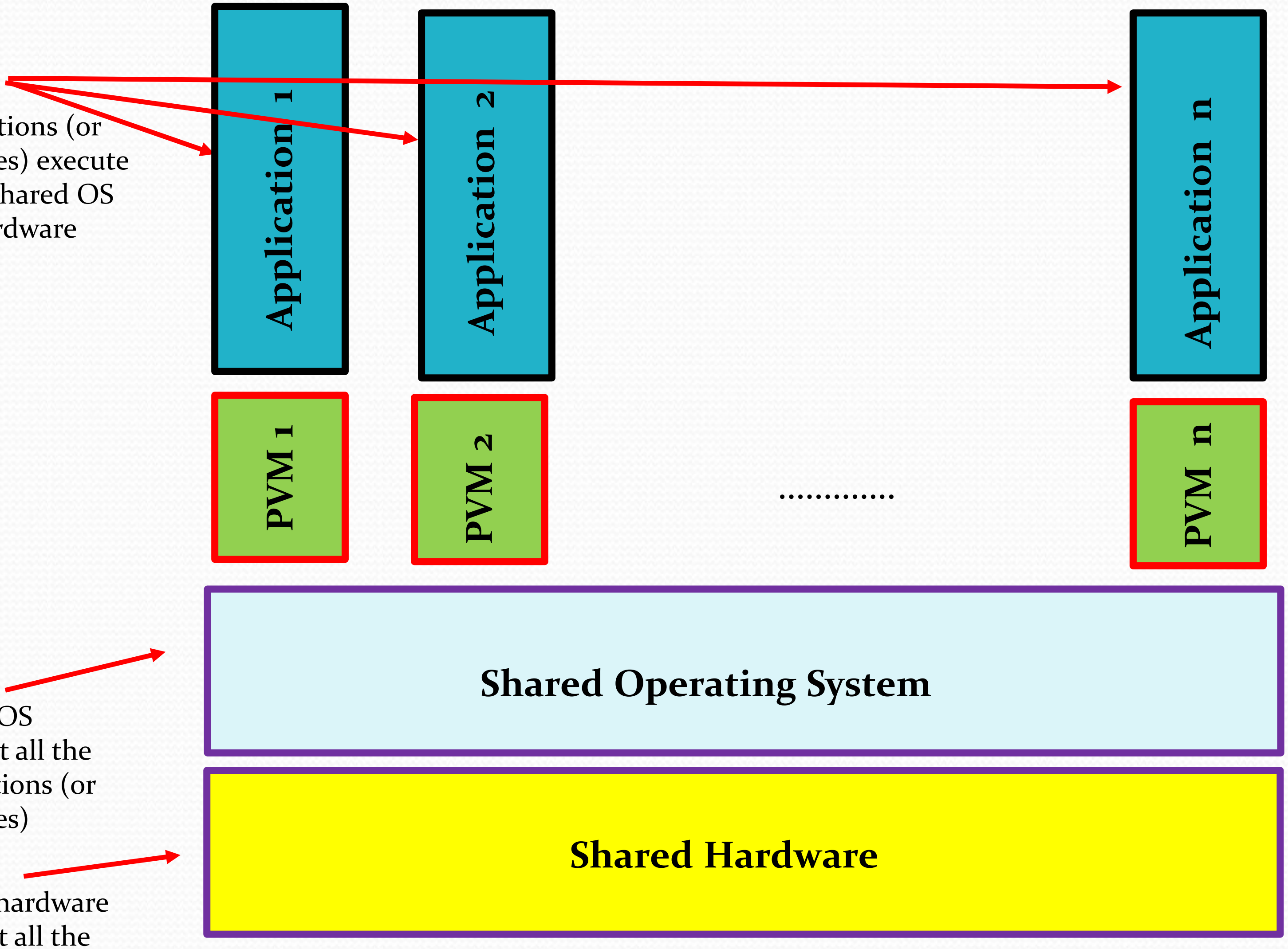
Figure: Illustration of the working of JVM

Java Virtual Machine is a type of PVM

- Similar to JVM, a PVM creates logical partitions of computing resources, within a given physical machine, and allocate each logical partition, to a single process (or application).

- Enables multiple processes to run on the same hardware and operating system environment, independent of each other

Process Virtual Machine



Application 1

Application 2

Application n

PVM 1

PVM 2

....

PVM n

Shared Operating System

Shared Hardware

Applications (or processes) execute on the shared OS and Hardware

Shared OS amongst all the applications (or processes)

Shared hardware amongst all the applications (or processes)

Figure: Process Virtual Machine

Process Virtual Machine (PVM)

- The various processes in a PVM, cannot operate or run their own OS. They share the same underlying OS.

- Abstraction of resources provided by Google App Engine is similar to that of PVM.

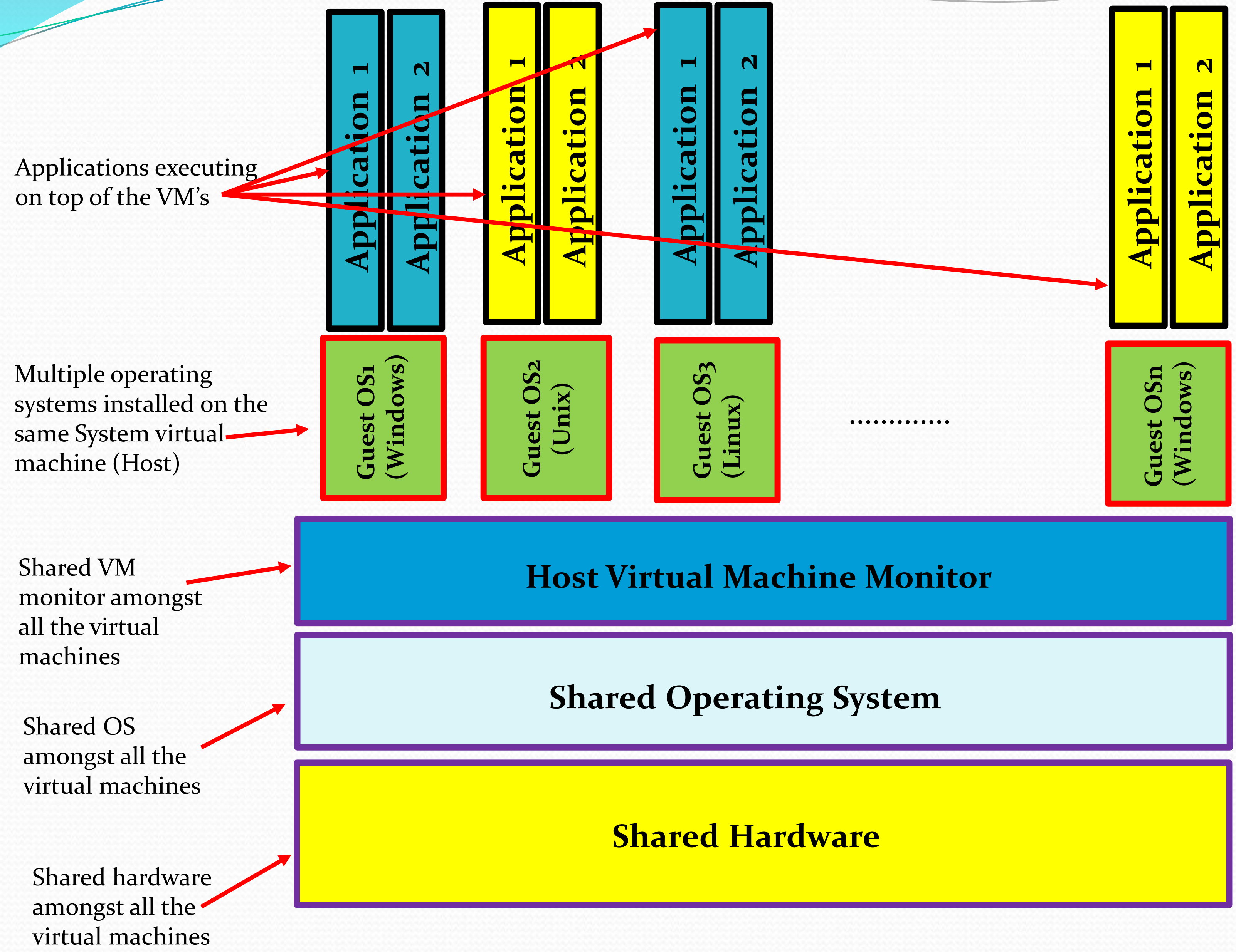
System Virtual Machine (Host)

- In contrast to PVM, each virtual machine in SVM-Host, can run an independent operating system instance, independent of the other VM instances
  - Such virtual machines are termed as "*system virtual machines*"

- In System Virtual Machine (Host) [*SVM-Host*], the virtual machines instances share the same hardware, *<u>and</u>* the same operating system ("*host operating system*")

- A SVM-Host can have many "instances" of the same operating system (say Linux or Windows) being run on it ("*Guest OS*")

System Virtual Machine (Host)


Applications executing on top of the VM's

Multiple operating systems installed on the same System virtual machine (Host)

Shared VM monitor amongst all the virtual machines
Host Virtual Machine Monitor

Shared OS amongst all the virtual machines

Shared hardware amongst all the virtual machines

Figure: System Virtual Machine (Host)

System Virtual Machine (Host)

- Virtual machine monitor (VMM) – or Hypervisors enable the system virtual machine instances to share the same hardware and the “host” operating system

- Virtual machine monitor software is installed and runs on the host operating system. Its functions include:
  - To create and manage the instances of system virtual machines;
  - Passing hardware access calls from Guest OS to Host OS, and converting them (if required).

System Virtual Machine (Native)

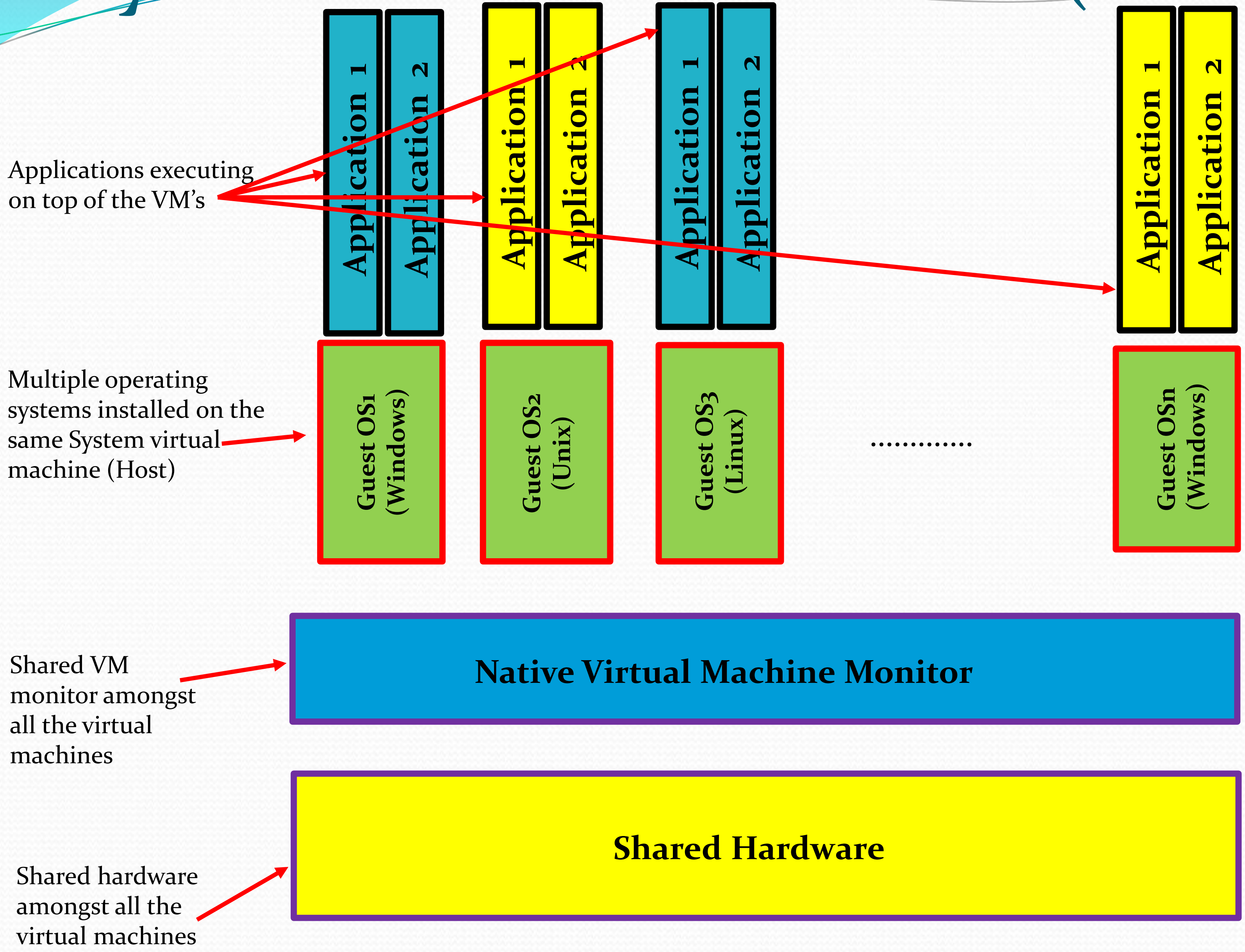
- System Virtual Machine (Native) [*SVM-Native*] does not have a host operating system

- The virtual machine monitor is installed and executes directly on the shared underlying hardware.

- VMM is responsible for creating and running the virtual machine instances on the underlying shared hardware (this process is known as *"native"*; in contrast to the shared host OS)

- The VMM maintains the state of each virtual machine

System Virtual Machine (Native)



Applications executing on top of the VM's

Multiple operating systems installed on the same System virtual machine (Host)

Shared VM monitor amongst all the virtual machines

Shared hardware amongst all the virtual machines

Figure: System Virtual Machine (Native)

System Virtual Machine vs. Process Virtual Machine

Factor	System Virtual Machine	Process Virtual Machine
Operating System	Each system virtual machine needs to be able to run a full operating system (of its own).	The various processes share the same operating system and cannot execute or run their own operating system

System Virtual Machine (Host) vs. System Virtual Machine (Native)

Factor	System Virtual Machine (Host)	System Virtual Machine (Native)
Host Operating Systems	A host (or base) operating system is shared across all the instances of the virtual machines.	There is no host operating system
Virtual Machine Monitor	Executes on top of the host operating system	Executes on top of underlying (shared) hardware

- Virtualization technology used by Amazon EC2 is based on SVM

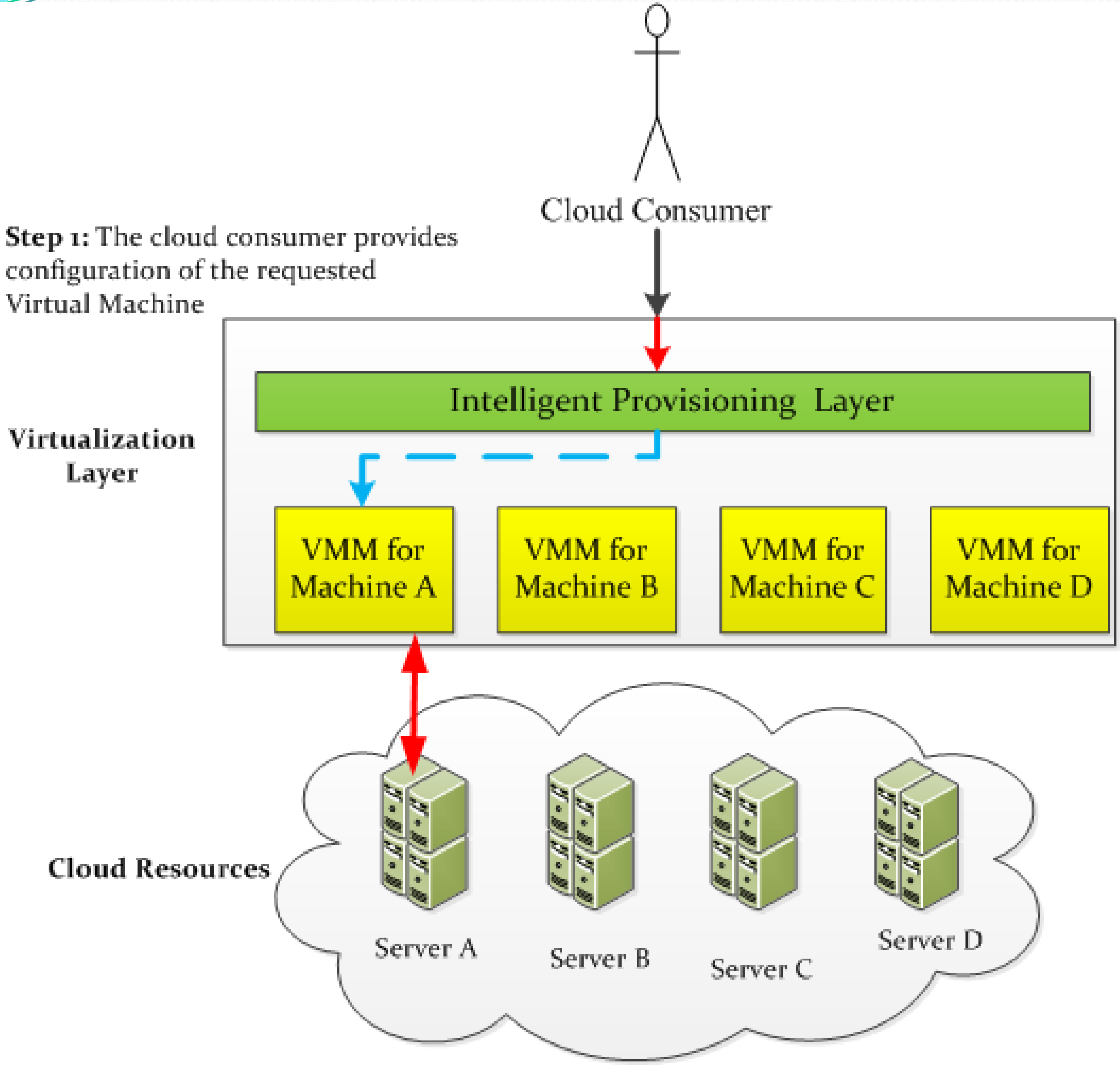
Cloud Computing and Virtualization

- Applications of virtualization in Cloud Computing is not confined to just creating virtual machines
- Major Cloud Providers make use of the virtualization layer to achieve *“elastic computing”*
- Elastic Computing:
  - refers to the ability of the cloud provider to allocate resources on demand to the users (in response to any reasonable resource request);
  - the resources (or additional capacity) can be made available to the cloud consumer(s) instantly;
  - gives the cloud consumers the illusion of “infinite capacity” by the cloud provider.
- Key in-built features for achieving elastic computing:
  - Automated Provisioning;
  - Elastic Operations

Automated Provisioning

- Complete automation of the process of provisioning virtual machines to the cloud consumers
  - This process is carried out by an “Intelligent Provisioning Layer” (part of virtualization layer)
- “Automatically” and “intelligently” decide (without any human intervention), the physical server on which a virtual machine should be executed, and assign it to the requesting cloud consumer
- The process of deciding which physical machine should execute the VM, is multi-factorial. Some of the factors are:
  - The resource requirements of the cloud consumer;
  - Available capacity and current utilization level of the existing physical machines;
  - Projected demand of applications running on the existing virtual machine(s)
- The Intelligent Provisioning Layer determines which existing physical machine can “best” host the virtual machine and allocates that VM to it

Automated Provisioning



Step 1: The cloud consumer provides configuration of the requested Virtual Machine

Step 2: The request is passed on to the intelligent provisioning layer.

Step 3: This layer takes into account:

- requested VM configuration by the Cloud Consumer
- the current resource utilization of the physical machines
- projected utilization of the machines

Based on the above it determines which physical server should host the virtual machine.

Figure: Automated Provisioning

Automated Provisioning

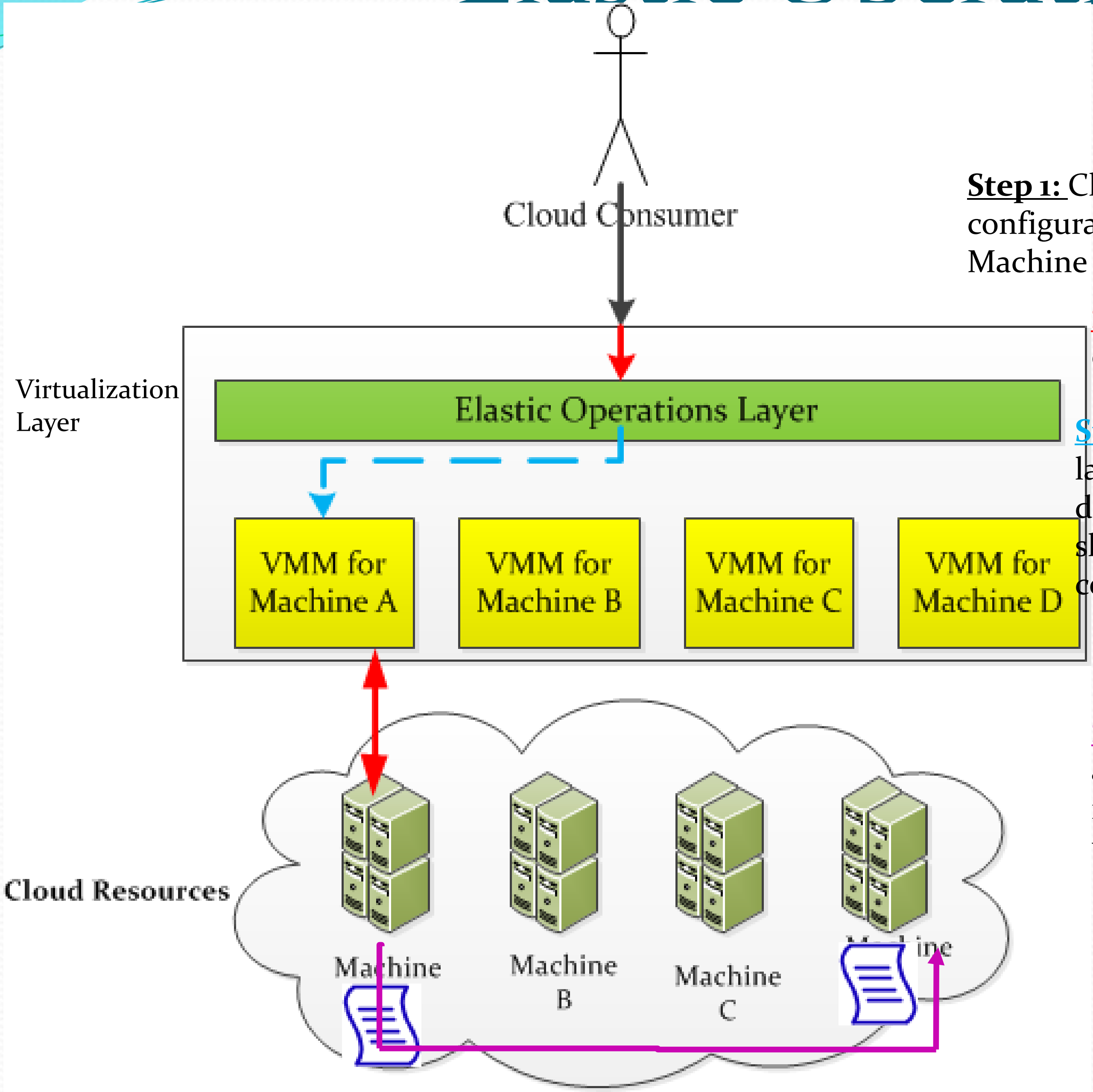
- The process of allocation, configuration and provisioning of the virtual machine is *completely automated*, without any human intervention

- The allocation of a given VM to a physical machine, is static in automated provisioning.

Elastic Operations

- In elastic operations (similar to elastic provisioning), the provisioning of virtual machines is completely automated
- In contrast to elastic provisioning, the allocation of a given VM to a physical machine, is dynamic and may change during the execution of the VM.
- The physical machine on which a given virtual machine runs is *dynamically* controlled, and is determined according to certain criteria, such as:
  - Collocation – move VM frequently communicating closer to each other;
  - Fault tolerance – move VM from physical machines that are likely to fail;
  - Improve utilization of underlying physical machines
- In this mode, virtual machine can be *"moved from"* one physical server to another, at run-time.

Elastic Operations



Step 1: Cloud Consumer provides the configuration of the requested virtual Machine

Step 2: The request is passed to the elastic operations layer

Step 3: The intelligent provisioning layer / elastic operations layer determines the physical machine that should execute the VM, according to certain pre-defined metrics

Step 4: The elastic operations layer additionally monitors the need to migrate VM's; and migrates it at run-time when there is a need.

Elastic Operations

- The migration process is governed by the “Elastic Operations Layer” along with the VMM’s (part of the virtualization layer)

- The process of moving a given virtual machine from one physical server to another is known as *Virtual Machine Migration*

- Depending on the virtual machine migration process employed this process may or may not be opaque to the cloud consumer

- If the number of underlying physical machines are very large; then this coupled with virtual machine migration, can give the cloud consumers an illusion of infinite capacity

Virtual Machine Migration

- The process of moving a running virtual machine from one physical server to another physical server

- Steps involved in the simplest version of VM Migration:
  - **<u>Step 1</u>**: Suspend the running virtual machine (prohibit any further user interaction with that virtual machine till the migration is complete);
  - **<u>Step 2</u>**: Save virtual machine state;
  - **<u>Step 3</u>**: Transport the suspended virtual machine to another physical machine (or server);
  - **<u>Step 4</u>**: Resume execution of the virtual machine from the suspended state (re-initiate interaction with the user)

- VMWare's *VMotion* is used for migration of VM's.

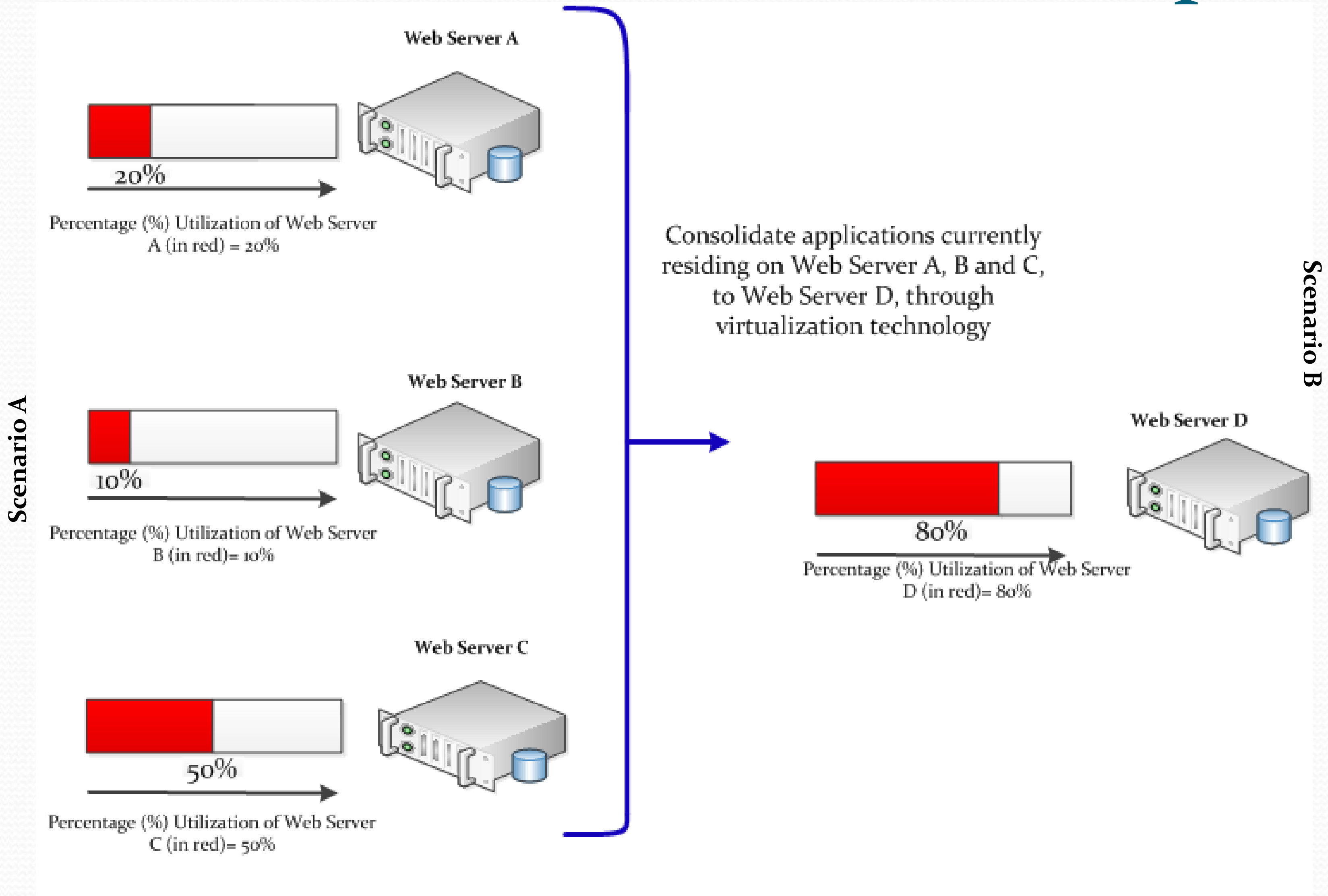
Virtualization Applications

- Some of the common applications of virtualization in enterprises are:
  - Server Consolidation;
  - Addressing enterprise security through the virtualization layer;
  - Desktop Virtualization

Server Consolidation

- Enterprise have multiple server(s) running various applications (leading to “server sprawl”)
- Usually these servers are under-utilized
- Each server requires
  - An upfront capital investment;
  - An ongoing human resource to maintain it;
  - Tedious and time-consuming task (to maintain each server)
- Virtualization offers the opportunity to consolidate applications running on different under-utilized servers across an enterprise
- Results in efficiency in terms of time and the cost of managing numerous multiple physical servers

Server Consolidation Example



Scenario A

Web Server A

20%

Percentage (%) Utilization of Web Server A (in red) = 20%

Web Server B

10%

Percentage (%) Utilization of Web Server B (in red)= 10%

Web Server C

50%

Percentage (%) Utilization of Web Server C (in red)= 50%

Consolidate applications currently residing on Web Server A, B and C, to Web Server D, through virtualization technology

Scenario B

Web Server D

80%

Percentage (%) Utilization of Web Server D (in red)= 80%

Figure: Pictorial representation of server consolidation

Security through the virtualization layer

- Due to the server sprawl, ensuring the security of the individual multiple servers is a challenging, and time consuming task.
- Using virtualization technology, security mechanisms (such as intrusion detection mechanisms, anti virus software etc.), can be made part of the VMM, rather than being executed on the Guest OS
- The security software monitors (such as Intrusion Detection Software) all the VM's rather than monitoring a single VM.
- Easy to detect and stop the virtual machine(s) with suspicious activity (or non-complying activities), without impacting or stopping the legitimate activities of other users

Security through the virtualization layer

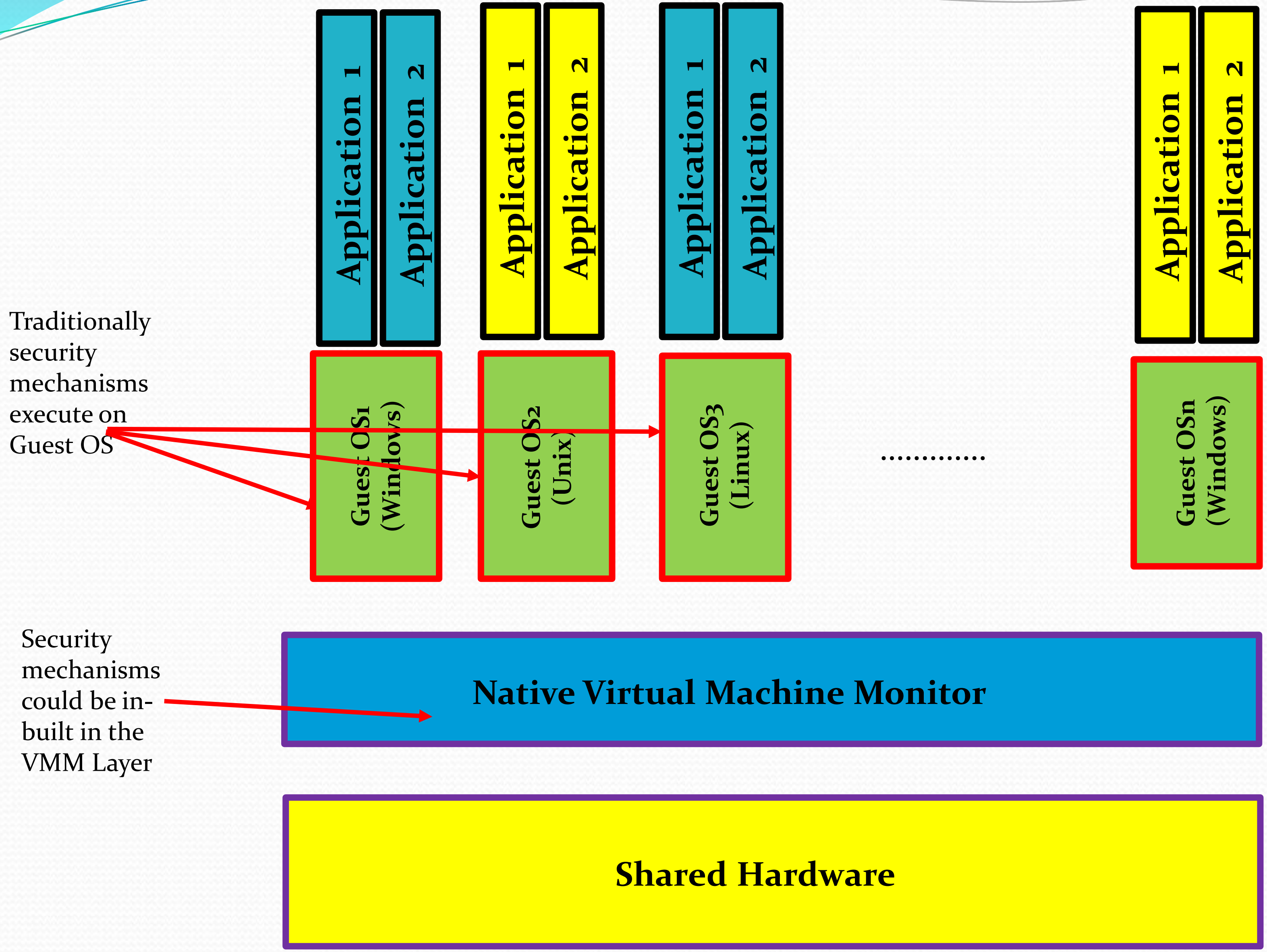


Figure: Achieving system security through VMM

Desktop Virtualization

- (Large) enterprises have a large and fluctuating employee base
- Huge costs involved in procuring, configuring, maintaining and updating the individual desktops of all the employees.
- Managing and ensuring regular software application update(s) on employee's desktops is a major system management task
- *"Desktop virtualization"* can be used to:
  - minimize costs of procuring desktops;
  - minimize effort in managing, updating the applications running on these desktops.
- Deploy and run all end-user desktops as virtual machines from a central enterprise server
- These desktops will be provisioned to the relevant users on demand (Desktop-as-a-Service)
  - In 2013, AWS launched Amazon WorkSpaces

Desktop Virtualization

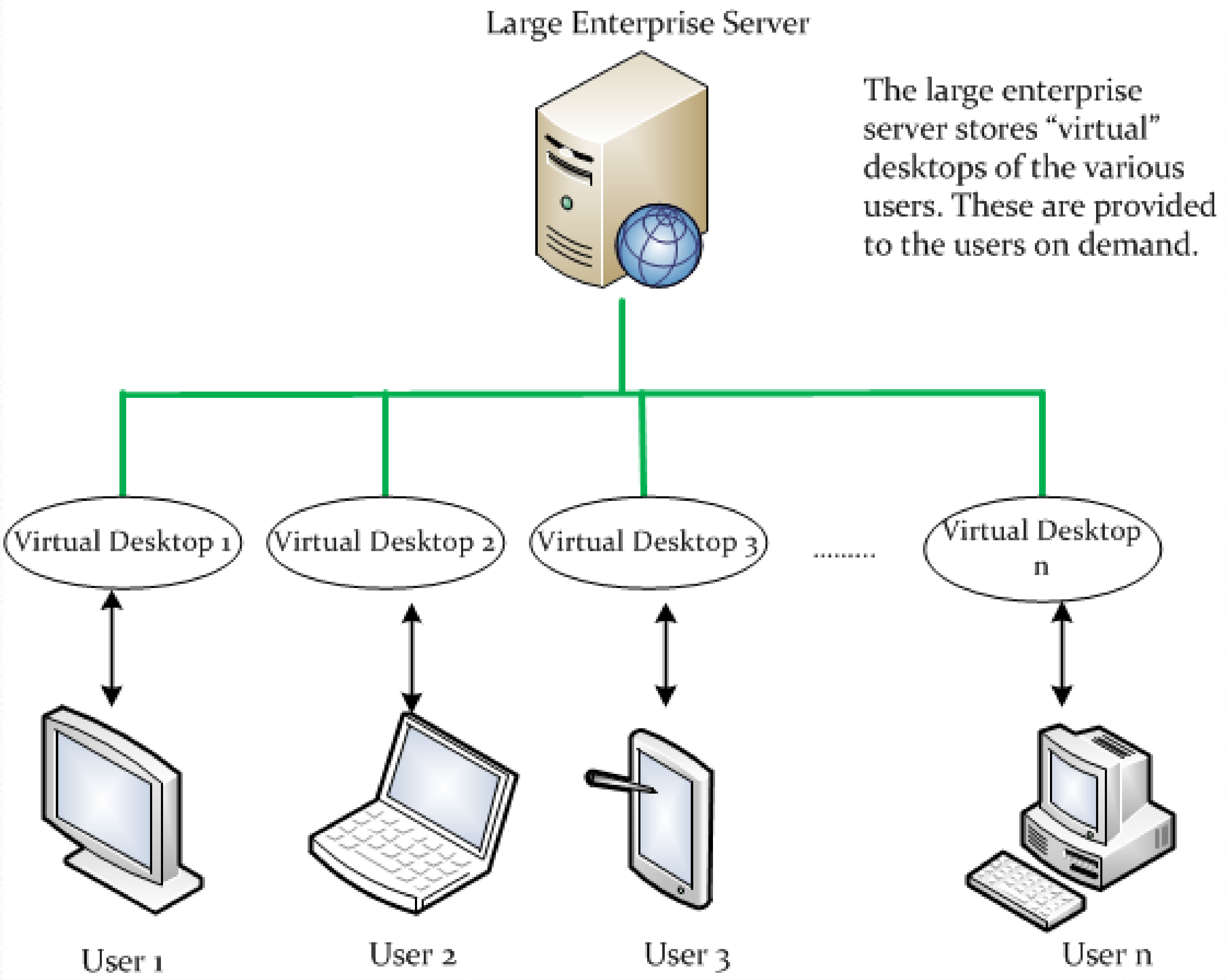


Figure: Desktop virtualization

Advantages of Desktop Virtualization

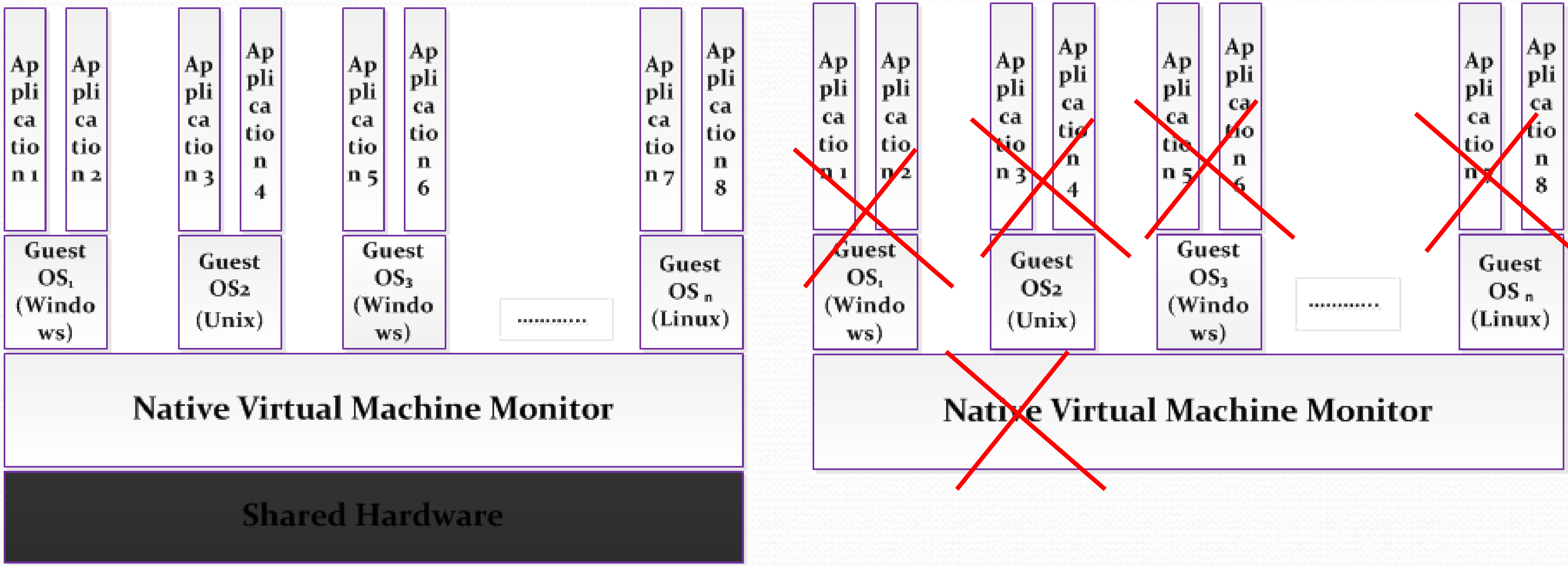
- New desktops (in the form of “virtual desktops”) can be made available very quickly

- Very easy for the enterprise to install new software application, install patches or upgrade existing application.

- Easy for the enterprise to monitor non-complying usage of desktops, as per relevant enterprise policy.

Virtualization Implications

- All the virtual machine instances would fail if the underlying hardware fails (or is compromised)



Application 1 Application 2 Application 3 Application 4 Application 5 Application 6 Application 7 Application 8

Guest OS_1 (Windows) Guest OS2 (Unix) Guest OS_3 (Windows) .... Guest OS_n (Linux)

Native Virtual Machine Monitor

Shared Hardware

Application 1 Application 2 Application 3 Application 4 Application 5 Application 6 Application 7 Application 8

Guest OS_1 (Windows) Guest OS2 (Unix) Guest OS_3 (Windows) .... Guest OS_n (Linux)

Native Virtual Machine Monitor

Figure: Failure of the shared hardware results in the subsequent failure of all the virtual machines running on top of it

Virtualization Implications

- All the virtual machine instances would fail if the underlying VMM fails (or is compromised)

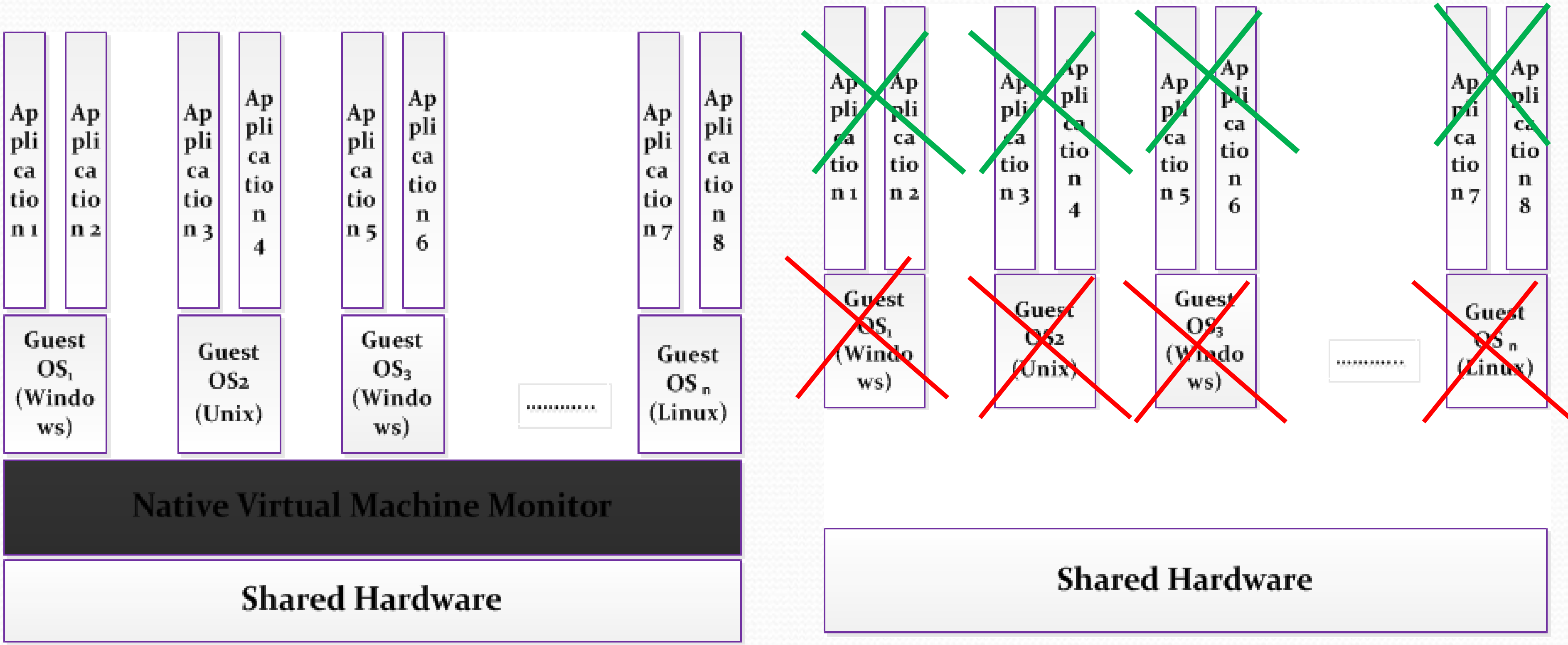


Figure: Failure of the shared VMM layer results in the subsequent failure of all the virtual machines running on top of it

Multi-tenancy

- Multi-tenancy in "*computing*" refers to the ability of *multiple users* being able to use the same hardware or software.

- Virtualization tools (such as VMWare etc..) enable infrastructure or hardware level multi-tenancy:
  - Different tenants can rent (or use) the same physical hardware using virtualization technology

- Multi-tenancy and Platform as a Service (PaaS):
  - Multiple tenants share the same *software resources* and/or hardware resources (a.k.a., computing platform);
  - Virtualization software in PaaS, enables sharing of *both* software framework and hardware resources amongst all the PaaS consumers.

Application-Level Virtualization

- Multi-tenancy and Software as a Service (SaaS)
  - Multiple enterprises (or tenants) use a shared set of *application code*; and *database* for holding the functionality and data of multiple enterprises or tenants;
  - Virtualization software in SaaS, enables sharing of both application code and hardware resources amongst its consumers

- Sharing of resources by PaaS or SaaS consumers is known as *"application level virtualization"*.

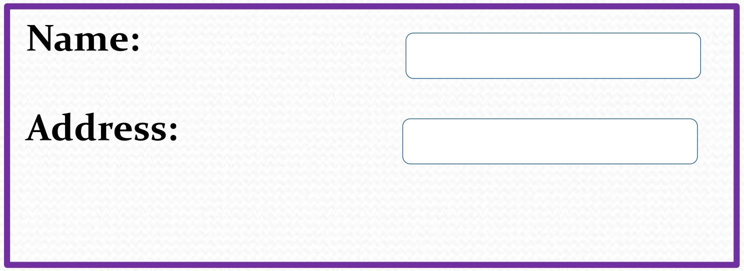
- A major challenge in application level virtualization is to store each tenant's customized data effectively, and present it to the corresponding tenant, on demand.

Implementing Multi-tenancy in SaaS

(Case Study) Consider a "*Customer Details Registration*" software application. This software application takes the customer details, from the end-user, and stores them in a database.

- This "Customer Details Registration Software" can be used by enterprises across the globe dealing with customers
- The "Customer Details Registration Software" can be made available to multiple tenants as a Software-as-a-Service (SaaS) application.
- However, for such as application different enterprises may require minor customizations, such as customized input data fields

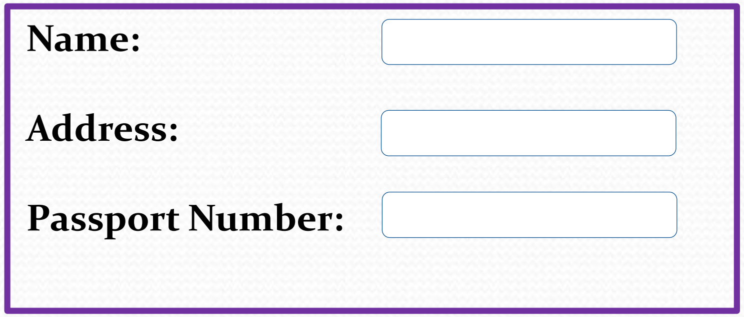
Customization of SaaS application



Name:

Address:

Figure: Example of standard application interface/fields

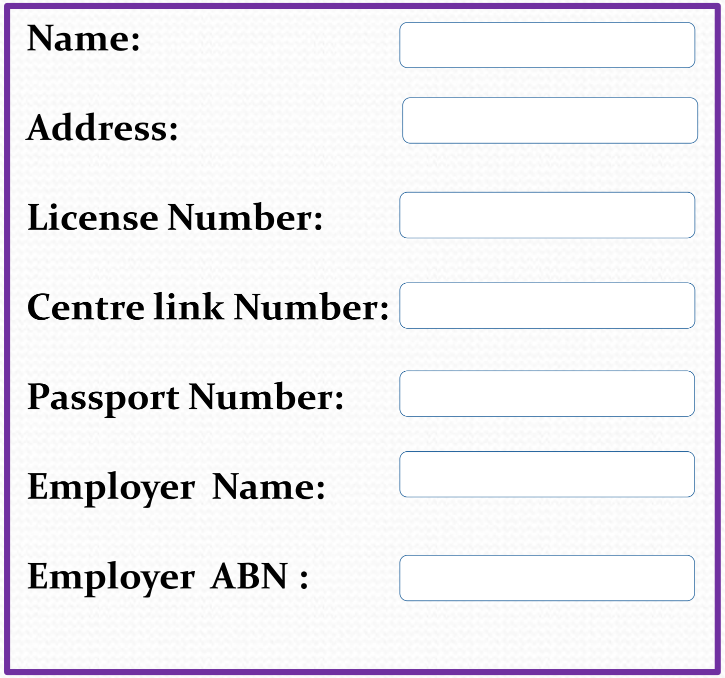


Name:

Address:

Passport Number:

Figure: Example of Customization needed for Customer/User A



Name:

Address:

License Number:

Centre link Number:

Passport Number:

Employer Name:

Employer ABN :

Figure: Example of Customization needed for Customer/User B

Data management approaches in multi-tenant application

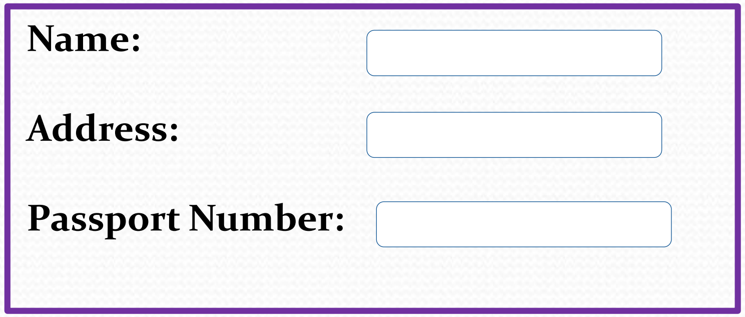
- Three data management approaches (or mechanisms) to store customized enterprise data:
  - **Approach 1:** Separate Database for each tenant.
  - **Approach 2:** Shared Database (amongst multiple tenants) but separate schema (for each tenant)
    - A schema is a collection of all the tables.
  - **Approach 3:** Shared Database (amongst multiple tenants) and shared schema (amongst multiple tenants).

Data management Approaches

- **Approach 1:** Separate database for each tenant
  - Each tenant is allocated and accesses its own separate database
  - The database of each tenant stores all the data (and customizations of the corresponding tenant)
  - This approach is simple to implement
  - Disadvantages of this approach are:
    - High maintenance costs of multiple databases (tens of thousands of databases) for a given application;
    - Additional costs to separately secure each database.

- This approach is not truly multi-tenant

Separate Database for each tenant



Name:

Address:

Passport Number:

Figure: Customization needed for Tenant/Customer/User A

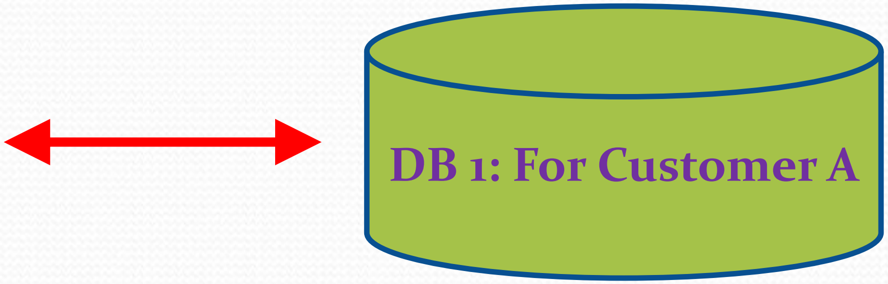
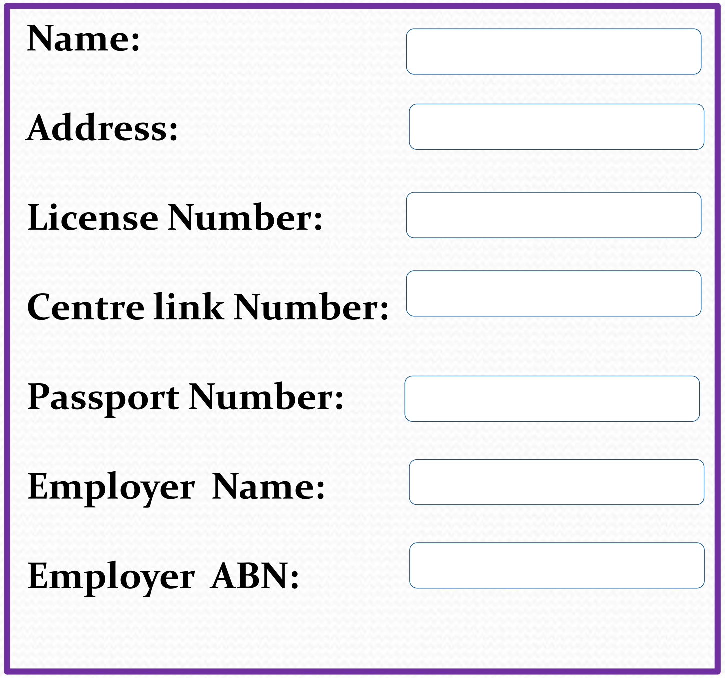


Figure: Database for Tenant/Customer/User A



Name:

Address:

License Number:

Centre link Number:

Passport Number:

Employer Name:

Employer ABN:

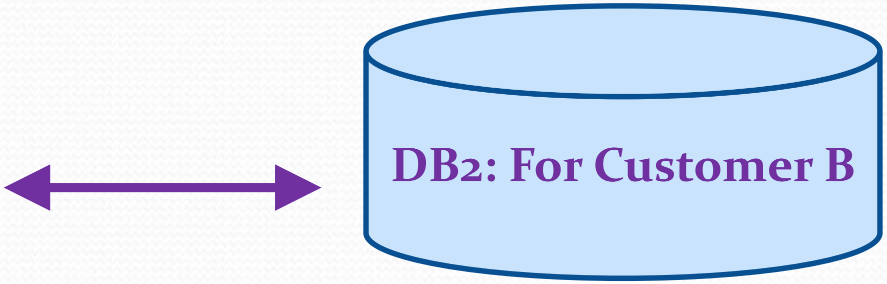


Figure: Database for Tenant/Customer/User B

Figure: Customization needed for Tenant/Customer/User B

Multi-tenancy – Data Management Approaches

- **Approach 2:** Shared database (for multiple tenants) but separate schema (to cater for tenant-specific customizations)
  - In contrast to the separate database approach, in this approach all the tenants share database(s)
  - However, in this approach each tenant has its own separate schema (modelled as its own separate set of tables).
  - The schema of a given customer (along with its tables), is responsible for storing the corresponding tenant's data
  - All the different schemas (corresponding to different tenants) are stored in the shared database
  - Metadata table redirects queries from a given tenant to its corresponding schema
  - Primary disadvantages of this approach is:
    - Becomes very complicated to manage in case of large number of tables of each customer/tenant

Shared Database and Separate Schema

Name:

Address:

Passport Number:

Figure: Customization needed for Customer A

Name:

Address:

License Number:

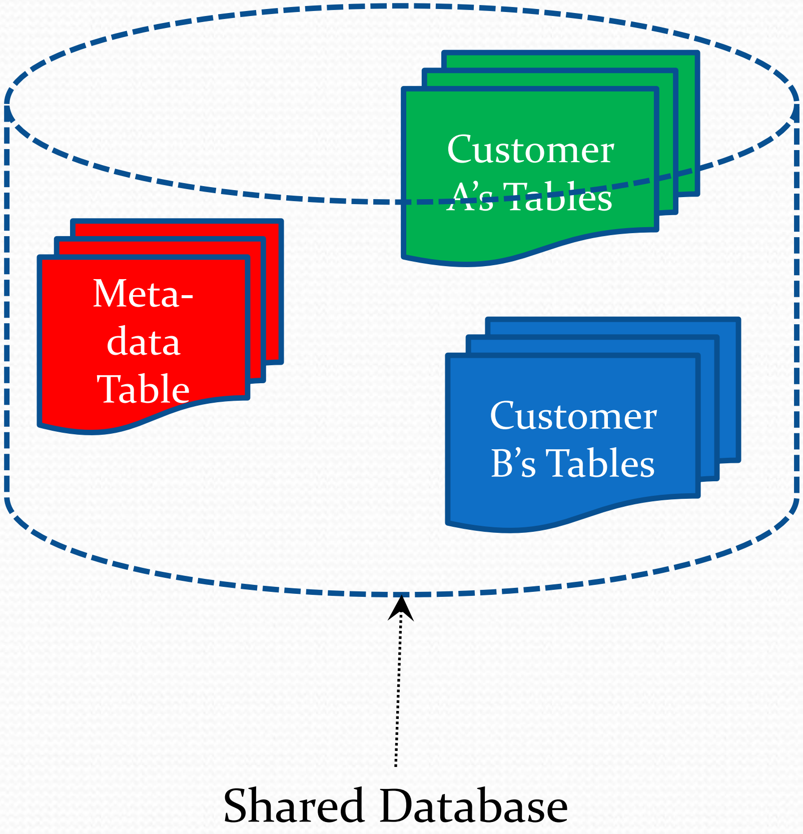
Centre link Number:

Passport Number:

Employer Name:

Employer ABN:

Figure: Customization needed for Customer B



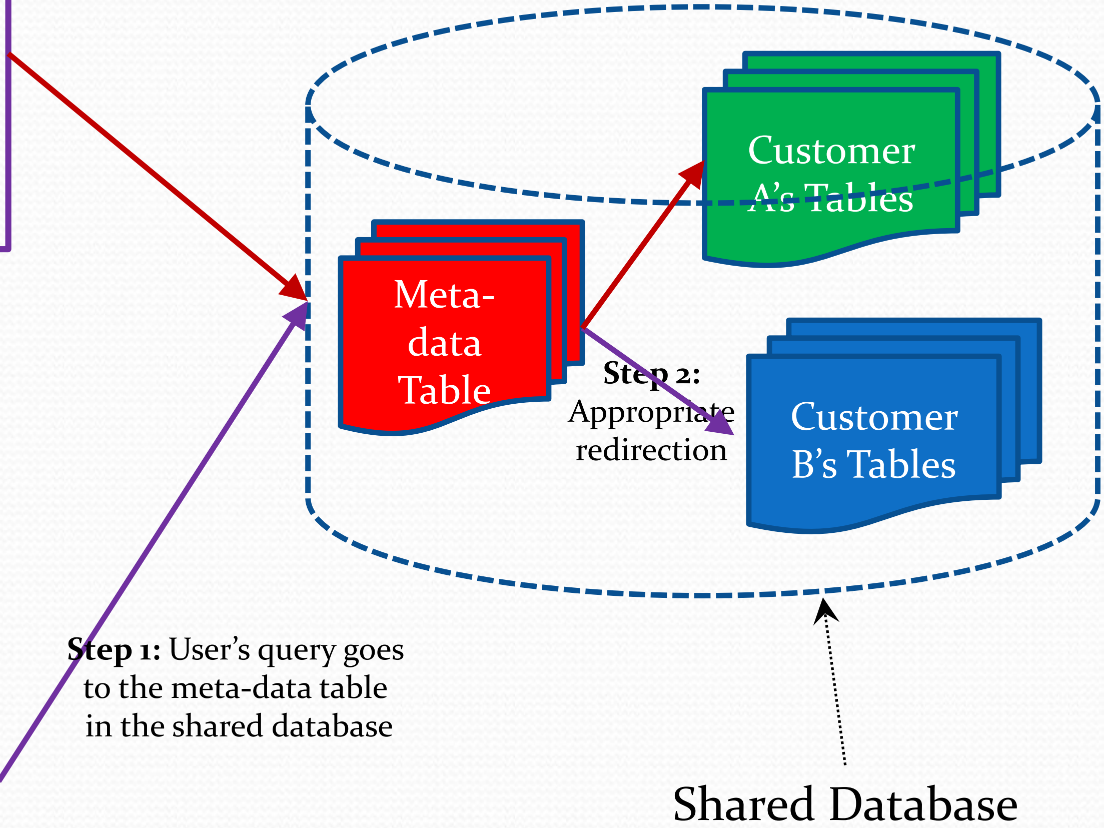
Shared Database and Separate Schema

Name:

Address:

Passport Number:

Figure: Customization needed for Customer A


Meta-data Table
Customer A's Tables
Customer B's Tables
Step 2: Appropriate redirection
Step 1: User's query goes to the meta-data table in the shared database
Shared Database

Name:

Address:

License Number:

Centre link Number:

Passport Number:

Employer Name:

Employer ABN:

Figure: Customization needed for Customer B

Multi-tenancy – Data Management Approaches

- Approach 3: Shared database (for multiple tenants), and Shared Schema
  - In this approach all the tenants share both the database(s) and the schema
  - A schema is a set of tables, that are shared between all the users/tenants
  - A single table can store records for multiple tenants
  - Tenant ID is used to uniquely identify the records of a specific tenant
  - A table created for each new (customized) field in the application
    - Example: Passport Number Table, License Number Table, Centre link Number Table, Employer Name Table, Employer ABN Table etc..
    - These tables are shared amongst all the tenants and do not belong to any tenant in particular
  - Joins between records are carried out at run time to retrieve all the data belonging to a given tenant
  - Common approach to achieving multi-tenancy in PaaS and SaaS applications.

Shared Database and Shared Schema

Name:

Address:

Passport Number:

Figure: Customization needed for Customer A

Name:

Address:

License Number:

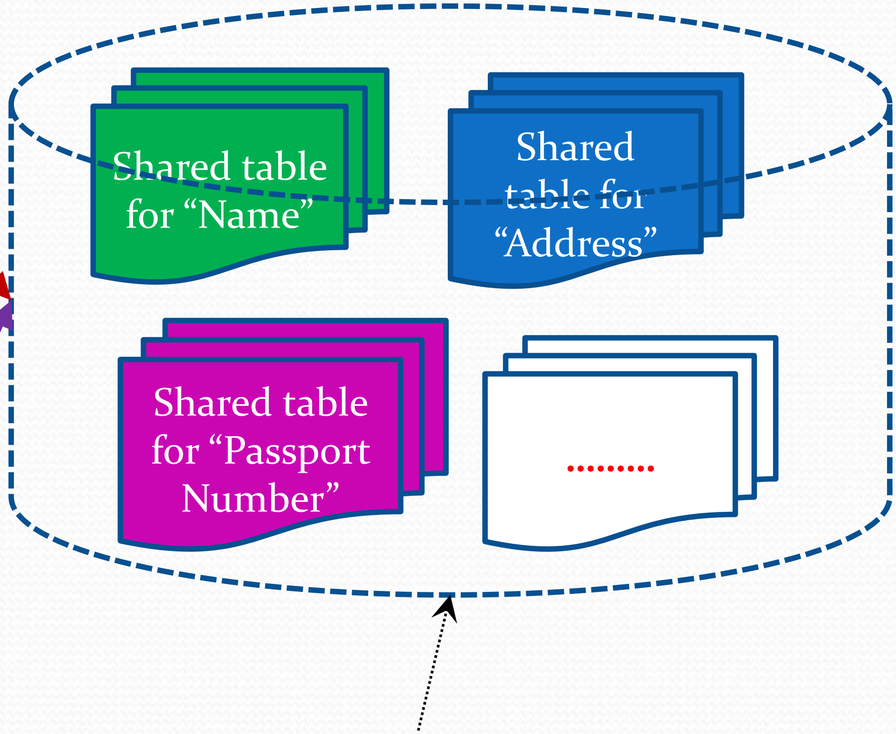
Centre link Number:

Passport Number:

Employer Name:

Employer ABN:

Figure: Customization needed for Customer B


Shared table for "Name"
Shared table for "Address"
Shared table for "Passport Number"
....

Shared Database used to store all the tables shared amongst the tenants

Example structure of “shared table”

- Each shared table comprises of multiple records, with each record belonging to a single tenant
- Each record is annotated with the “Tenant ID” to identify who it belongs to.

Tenant ID	Passport Number
A12345	AUS-00087
A34566	AUS-00067
A99999	AUS-10055
A87562	AUS-02367
……….	……….

Figure: Example of a shared table for “Passport Number”

Reading

Books

1. Shroff, G. (2010), Enterprise cloud computing: technology, architecture, application, Cambridge University Press, UK – Chapters 8 and 9