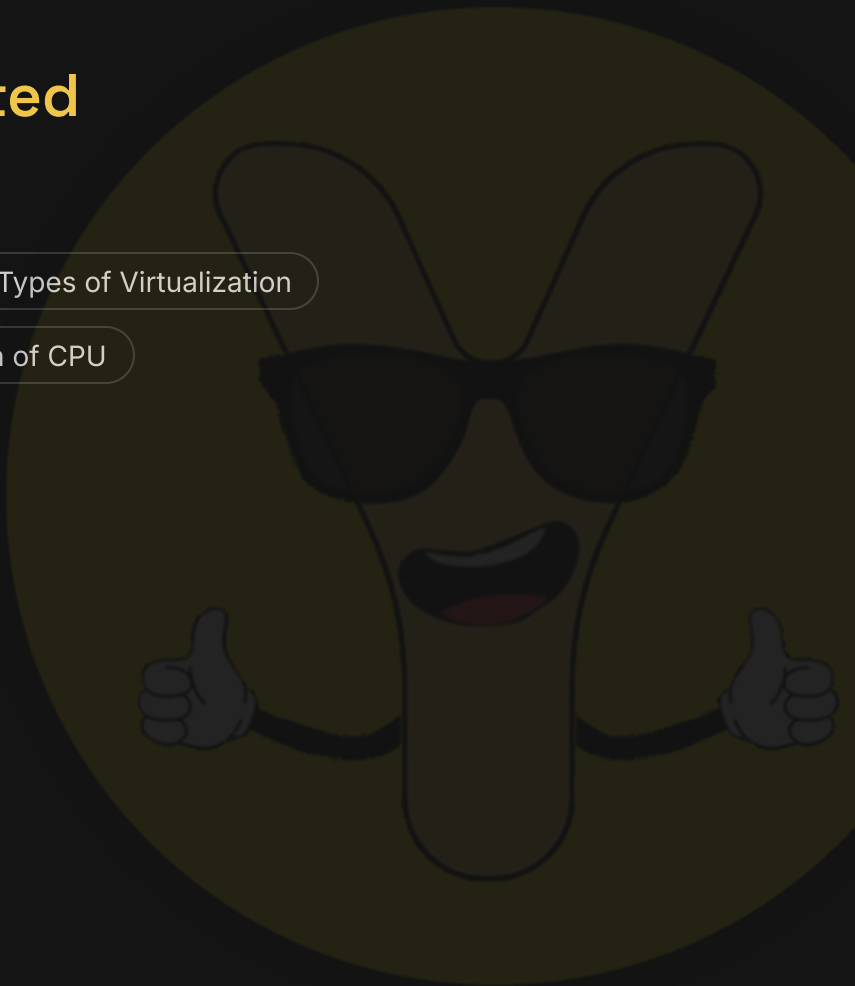
AKTU · 3RD YEAR · SEMESTER 5 · BCAM051

Cloud Computing

Unit 2 — Cloud Enabling Technologies & Service Oriented Architecture

- REST and Systems of Systems
- Web Services
- Publish, Subscribe Model
- Basics of Virtualization
- Types of Virtualization
- Implementation Levels of Virtualization
- Virtualization Structures
- Tools and Mechanisms
- Virtualization of CPU
- Virtualization of Memory
- Virtualization of I/O Devices
- Virtualization Support and Disaster Recovery



DECODED BY

Chirag Jain

Founder, Ycotes · Gold Medalist in Artificial Intelligence, IIT Madras · taught 10,000+ students

REST and Systems of Systems

Representational State Transfer (REST) serves as the architectural foundation for scalable Systems of Systems (SoS) in cloud environments.

- System of Systems (SoS) integrates multiple independent, heterogeneous systems to deliver complex capabilities.
- REST is an architectural style based on stateless, client-server, cacheable communications over standard HTTP.
- Statelessness ensures each request contains all data needed for processing, improving scalability and fault tolerance.
- Uniform Interface uses standard HTTP verbs (GET, POST, PUT, DELETE) to manipulate resources identified by URIs.
- Loose coupling in REST enables independent evolution, deployment, and scaling of constituent systems in an SoS.
- RESTful services leverage lightweight formats like JSON and XML for low-overhead inter-system communication.

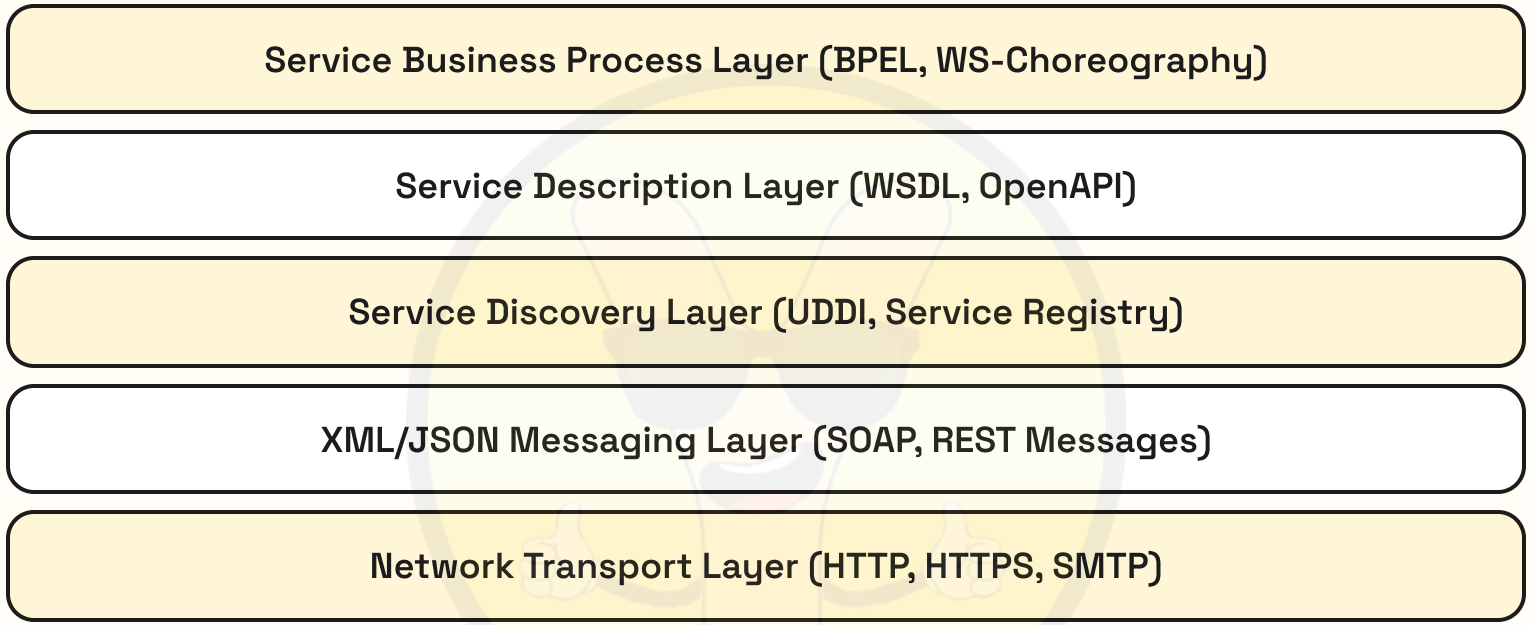
Web Services in SOA

WEB SERVICE

A software system designed to support interoperable machine-to-machine interaction over a network, using standard web technologies and protocols.

- Forms the core realization of Service Oriented Architecture (SOA) principles.
- Enables platform-independent and language-agnostic service consumption.
- Relies on open standards like XML, JSON, HTTP, SOAP, and WSDL.
- Supports service discovery, dynamic binding, and automated orchestration.

Web Services Technology Stack Architecture



Layered architecture of Web Services within Service Oriented Architecture.

Comparison: SOAP vs REST Web Services

Feature	SOAP (Simple Object Access)	REST (Representational State)
Protocol / Style	Strict XML-based Protocol	Architectural Style using HTTP
Data Format	XML only	JSON, XML, HTML, Plain Text
State Management	Can be Stateful or Stateless	Strictly Stateless
Bandwidth / Overhead	High overhead due to XML envelop	Lightweight with low bandwidth
Security Standard	WS-Security, Enterprise grade	HTTPS, OAuth 2.0, JWT
Use Case	Financial & Enterprise Transactions	Web APIs, Mobile Apps, Microservices

Publish-Subscribe Message Exchange Pattern



Decoupled asynchronous communication pattern using message brokers.

Publish-Subscribe Model

An asynchronous messaging paradigm where senders (publishers) do not target specific receivers (subscribers).

- Space Decoupled: Publishers and subscribers do not need to know each other's IP addresses or identities.
- Time Decoupled: Interacting parties do not need to be online or active simultaneously.
- Synchronization Decoupled: Publishers post events without blocking their processing execution thread.
- Topic-Based Filtering: Messages are categorized into channels/topics; subscribers receive all topic messages.
- Content-Based Filtering: Subscribers define rules on message attributes to filter incoming data.
- Commonly implemented using cloud tools like Apache Kafka, RabbitMQ, and AWS SNS.

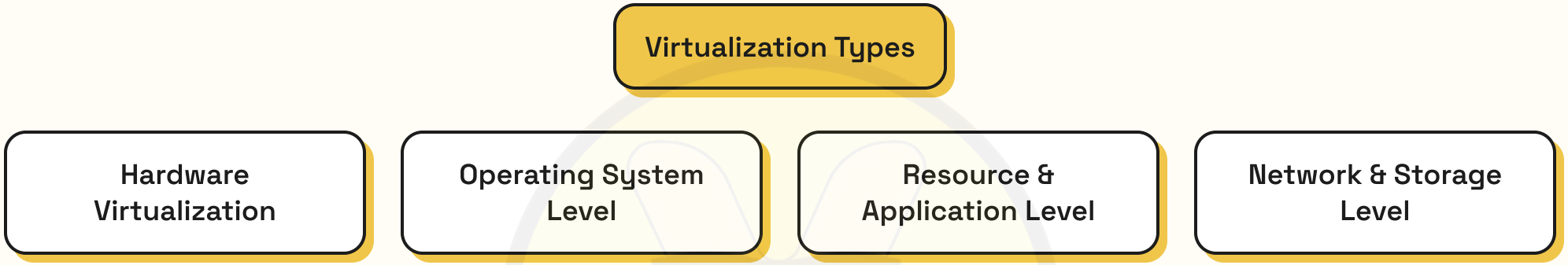
Basics of Virtualization

VIRTUALIZATION

A technology that abstracts physical hardware resources to create multiple isolated simulated environments or virtual hardware devices.

- Introduces a Virtual Machine Monitor (VMM) or Hypervisor layer above raw hardware.
- Converts physical resources (CPU, Memory, Storage, I/O) into logical pools.
- Enables multi-tenancy, high resource utilization, and isolated execution.
- Forms the fundamental technical foundation of cloud computing platforms.

Taxonomy of Virtualization Types



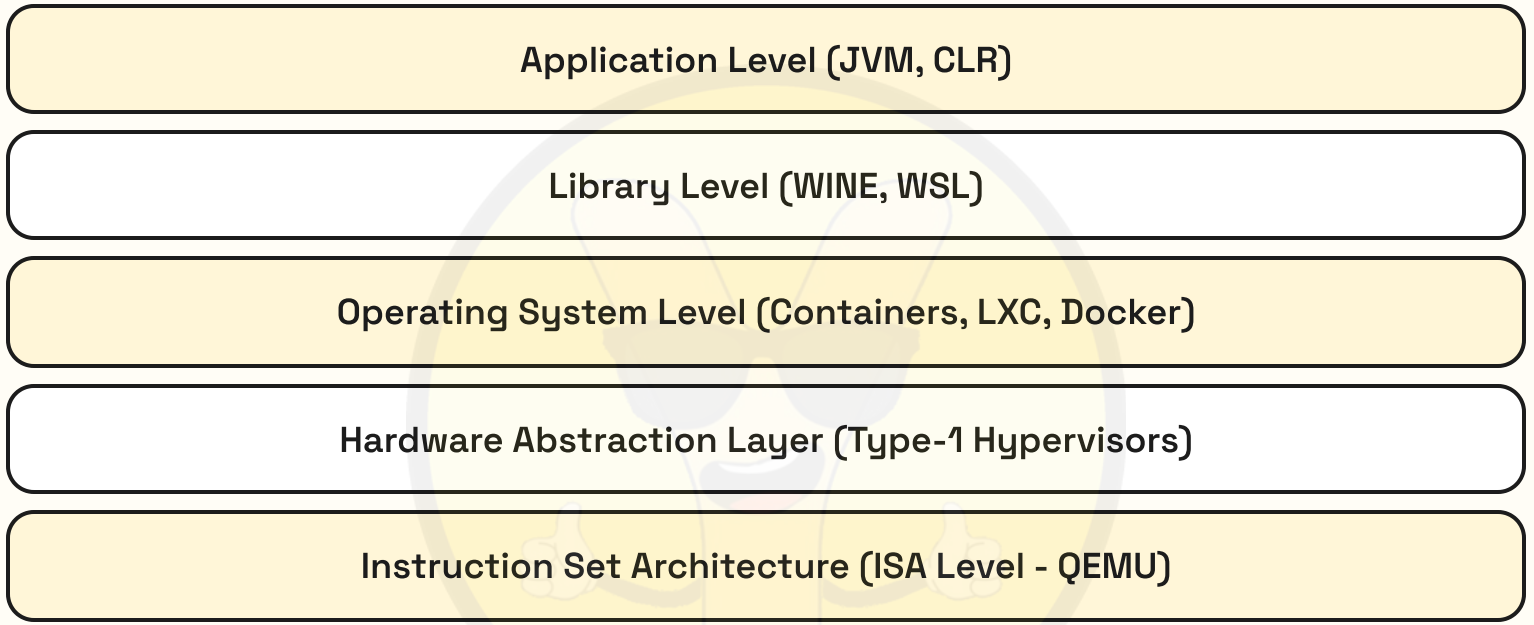
Classification tree of core virtualization techniques used in modern systems.

Types of Virtualization

Virtualization spans across multiple hardware and software abstractions in cloud platforms.

- Hardware Virtualization: Virtualizes complete physical servers to run multiple full OS instances.
- Operating System Virtualization: Containers share a single OS kernel while isolating user space.
- Memory Virtualization: Aggregates physical RAM into a unified logical pool for dynamic allocation.
- Storage Virtualization: Combines multiple physical disks into a single logical storage entity (SAN/NAS).
- Network Virtualization: Creates logical virtual networks (VLANs, VXLANs) decoupled from underlying hardware.
- Application Virtualization: Encapsulates software apps to execute independently of the local host OS.

Implementation Levels of Virtualization



Five software stack levels where virtualization can be implemented.

Implementation Levels of Virtualization

Virtualization can be applied at five distinct levels of the system architecture spectrum.

- ISA Level: Virtualizes hardware at instruction level via dynamic binary translation; high overhead.
- Hardware Abstraction Level: Virtualizes directly over raw hardware; highly efficient (e.g., KVM, ESXi).
- OS Level: Virtualizes OS user space; lightweight, shares host kernel (e.g., Docker, LXC).
- Library Level: Intercepts API calls from application to OS system libraries (e.g., WINE).
- Application Level: High-level language virtual machines executing bytecode (e.g., JVM, .NET CLR).

Virtualization Structures: Type-1 vs Type-2 Hypervisors

Property	Type-1 (Bare-Metal) Hypervisor	Type-2 (Hosted) Hypervisor
Installation Layer	Runs directly on bare physical hardware	Runs on top of a conventional Host OS
Performance	Near-native speed, minimal overhead	Slower due to Host OS context switches
Security	High isolation; smaller attack surface	Dependent on host OS security posture
Resource Access	Direct control of physical hardware	Requests hardware via host OS APIs
Examples	VMware ESXi, Xen, KVM, Hyper-V	Oracle VirtualBox, VMware Workstation

Tools and Mechanisms

Essential software tools and hardware features required to construct and manage virtual environments.

- Hypervisors / VMMs: Core engines managing virtual machine lifecycles and hardware allocation.
- Binary Translation: Software mechanism converting non-virtualizable guest instructions on-the-fly.
- Hardware Assist Extensions: Intel VT-x and AMD-V instructions enabling root and non-root execution modes.
- Paravirtualization Drivers: Specialized guest OS drivers (virtio) that bypass emulation for fast I/O.
- Container Engines: Runtime tools like Docker and Podman managing cgroups and namespaces.
- Cloud Management Tools: Platforms like OpenStack and CloudStack orchestrating hypervisor clusters.

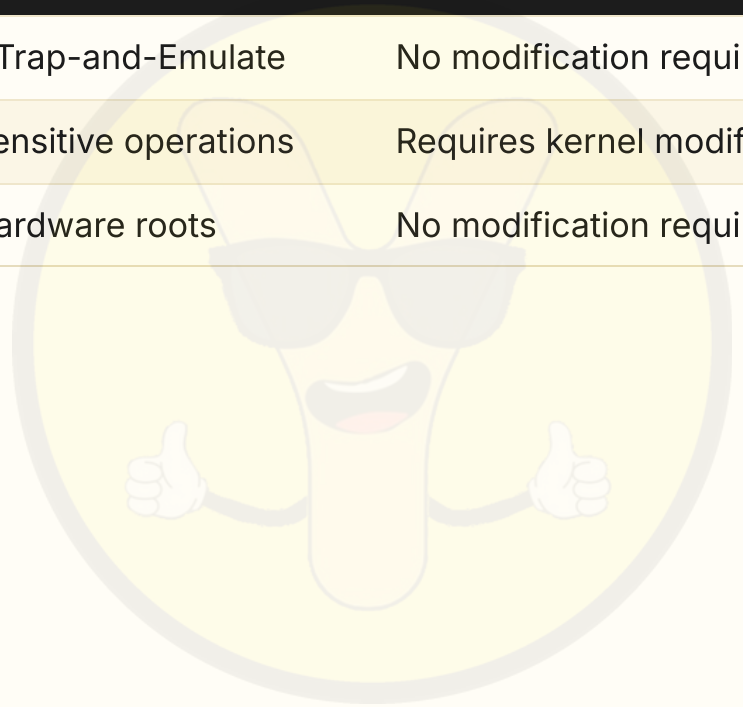
Virtualization of CPU

CPU virtualization ensures multiple Guest OS instances share physical CPU cores safely and efficiently.

- Privilege Rings: x86 architecture uses Ring 0 (Kernel) to Ring 3 (User apps).
- Ring Deprivileging: Guest OS is pushed from Ring 0 to Ring 1 or 3 to prevent unauthorized control.
- Sensitive vs Privileged Instructions: Popek-Goldberg rule states a system is fully virtualizable if all sensitive instructions are privileged.
- Trap-and-Emulate: Hypervisor catches guest sensitive instructions and executes them safely.
- Full Virtualization: Uses Binary Translation for non-virtualizable x86 instructions; guest OS needs no modifications.
- Paravirtualization: Guest OS kernel is modified to replace sensitive instructions with hypercalls.

CPU Virtualization Techniques

Technique	Mechanism	Guest OS Modification	Performance Overhead
Full Virtualization	Binary Translation + Trap-and-Emulate	No modification required	Medium to High overhead
Paravirtualization	Hypercalls replace sensitive operations	Requires kernel modification	Low overhead
Hardware-Assisted	Intel VT-x / AMD-V hardware roots	No modification required	Lowest overhead (Near-native)

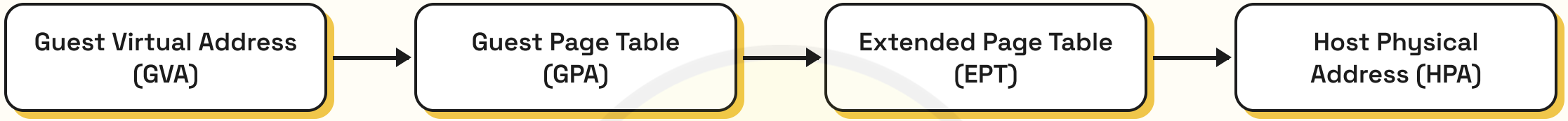


Virtualization of Memory

Memory virtualization creates an illusion of contiguous physical memory for each virtual machine.

- Three Memory Address Levels: Guest Virtual Address (GVA), Guest Physical Address (GPA), Host Physical Address (HPA).
- Mapping Requirement: System must map GVA → GPA → HPA efficiently.
- Shadow Page Tables (SPT): Software technique where hypervisor maintains a direct GVA → HPA lookup table.
- Extended Page Tables (EPT / NPT): Hardware-assisted two-dimensional page walking performed directly by CPU MMU.
- Memory Overcommitment: Allocates more logical RAM to VMs than physical RAM available using swapping.
- Page Sharing: Deduplicates identical physical memory pages (e.g., shared OS code) across VMs.

Memory Address Translation in Virtualization



Hardware-assisted two-stage memory address translation process.

Virtualization of I/O Devices

I/O virtualization manages access to physical network adapters, storage controllers, and peripherals.

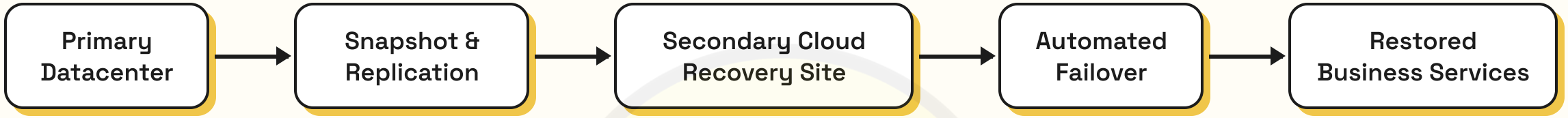
- Emulation: Hypervisor emulates legacy hardware in software; maximum compatibility but very high CPU overhead.
- Paravirtualized I/O: Guest uses split-driver architecture (front-end driver in VM, back-end driver in hypervisor).
- Direct I/O Passthrough: Assigns dedicated physical PCIe devices directly to a specific guest VM.
- SR-IOV (Single Root I/O Virtualization): Hardware standard allowing one physical PCIe device to present multiple virtual functions.
- Shared Memory Buffers: Circular ring buffers used for high-speed inter-VM and hypervisor I/O exchange.
- IOMMU Support: Input-Output Memory Management Unit translates virtual device addresses to physical memory safely.

Virtualization Support and Disaster Recovery

Virtualization simplifies high availability, business continuity, and disaster recovery (DR) mechanisms.

- Encapsulation: Entire VM (OS, apps, configuration, data) is stored as simple disk image files (.vmdk, .qcow2).
- Live Migration: Transfers a running VM from one host to another with zero downtime using pre-copy/post-copy RAM memory transfer.
- Snapshot Management: Captures exact point-in-time state of VM memory and storage for quick rollback.
- Continuous Data Replication: Synchronously or asynchronously mirrors VM snapshots to secondary cloud sites.
- Recovery Point Objective (RPO): Maximum acceptable data loss duration during a disaster event.
- Recovery Time Objective (RTO): Maximum allowable time to restore business operational status.

Cloud Disaster Recovery Process Flow



Workflow for automated cloud disaster recovery and failover.

AKTU Exam Corner: Likely Questions

- **Q1** Explain the REST architectural style and its role in building Systems of Systems (SoS).
- **Q2** Compare SOAP and REST web services with respect to architecture, data format, security, and performance.
- **Q3** Describe the Publish-Subscribe messaging model and explain how it achieves space, time, and synchronization decoupling.
- **Q4** Explain the five implementation levels of virtualization with neat diagrams and concrete examples.
- **Q5** Differentiate between Type-1 and Type-2 hypervisors. List popular examples of each.
- **Q6** Discuss CPU Virtualization. Explain Popek-Goldberg virtualization requirements and contrast Full Virtualization, Paravirtualization, and Hardware-Assisted Virtualization.
- **Q7** Explain Memory Virtualization. Describe address mapping from GVA to HPA using Shadow Page Tables and Extended Page Tables (EPT).
- **Q8** Write short notes on: (a) SR-IOV in I/O Virtualization, (b) VM Live Migration, (c) RPO and RTO in Cloud Disaster Recovery.

Unit 2 done. You are exam-ready.

Revise the Exam corner once more, then open Unit 3 on theycotes.com. Every AKTU subject, unit by unit, free. Notes by Chirag Jain, Ycotes.

