SNS COLLEGE OF TECHNOLOGY

An Autonomous Institution

Coimbatore-35

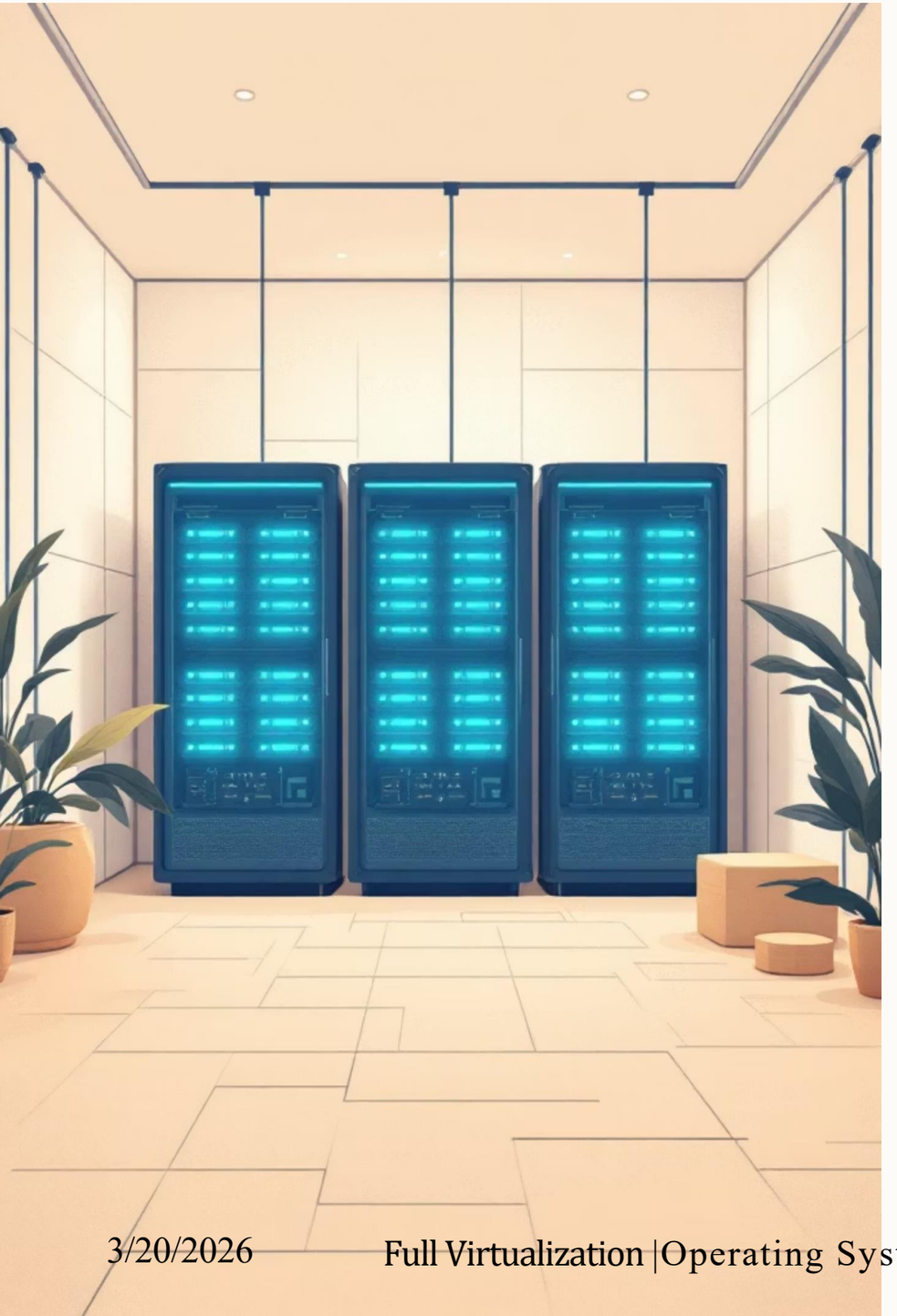
Department of Computer Science and Engineering

23CST206–OPERATING SYSTEMS AND VIRTUALIZATION

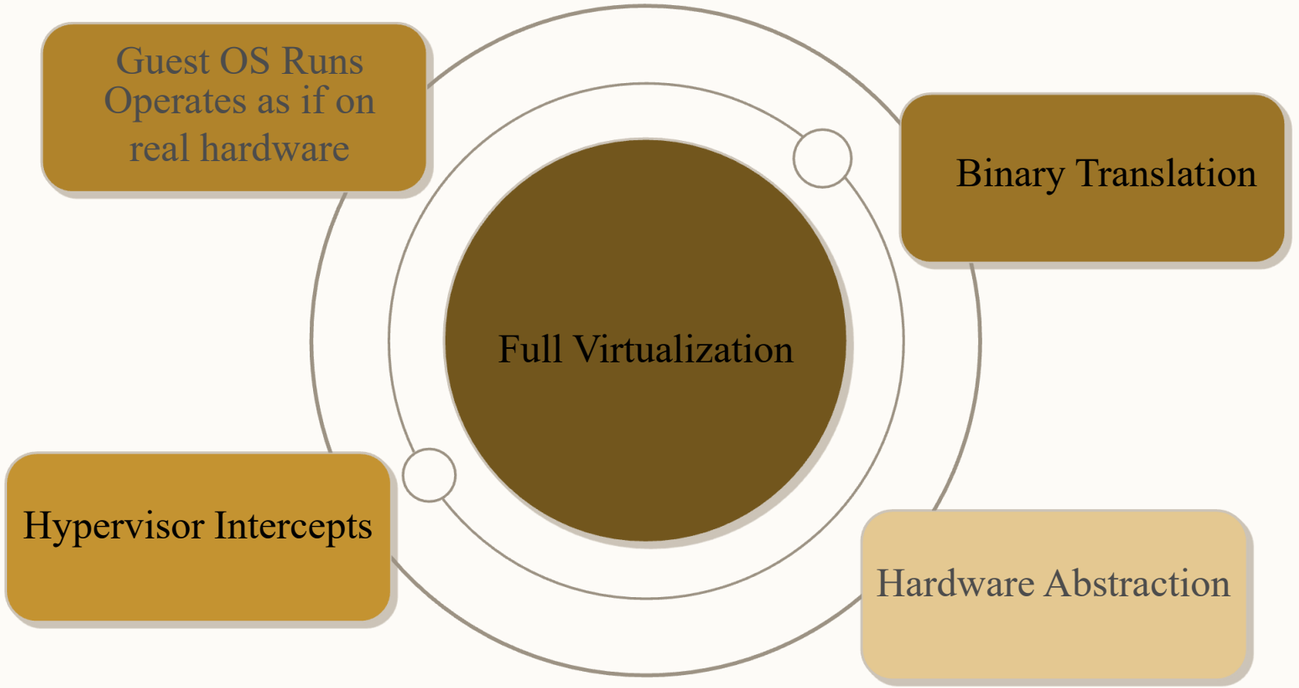
B.E- CSE /IV SEMESTER

UNIT – IV VIRTUALIZATION

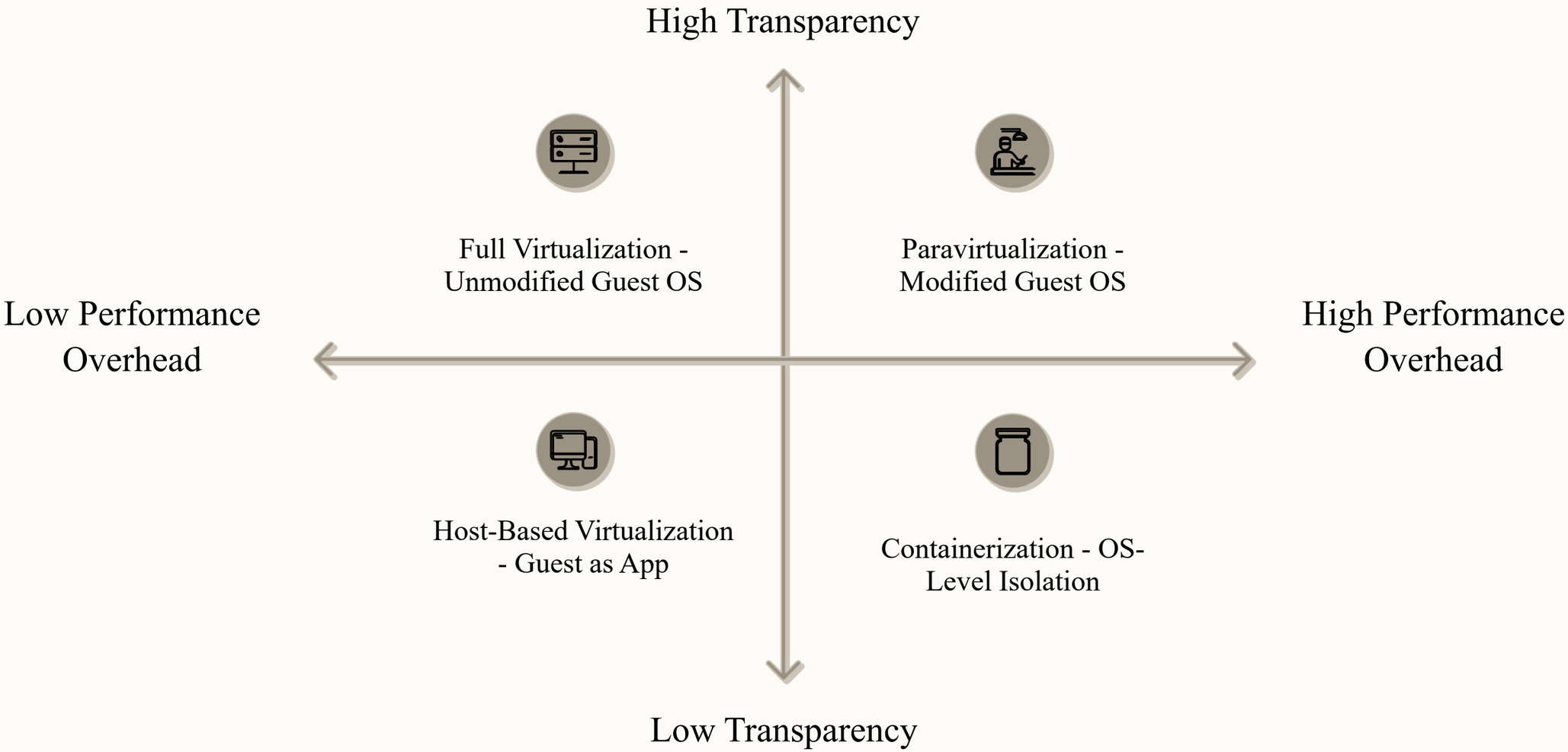
Topic 3:Full Virtualization with Binary Translation



Full Virtualization with Binary Translation



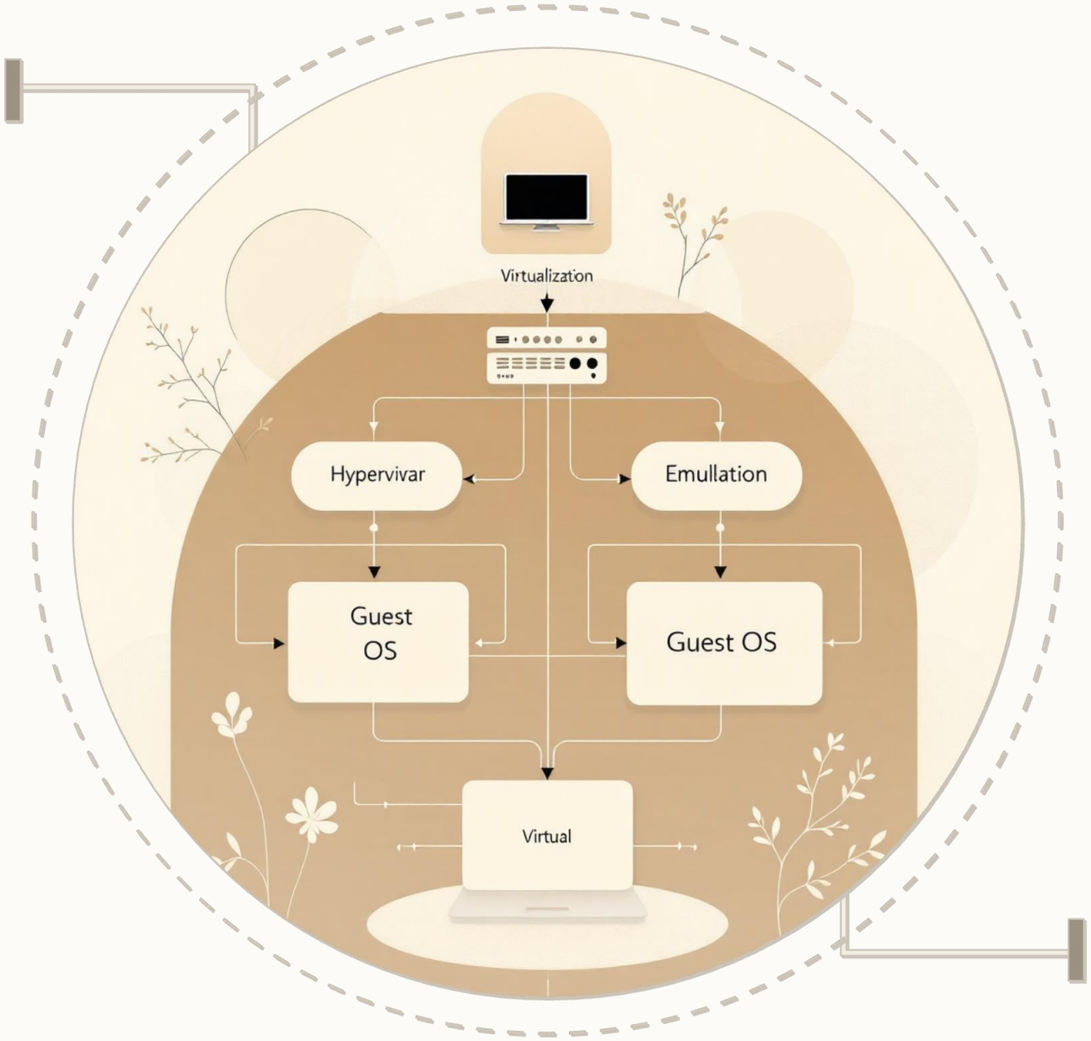
Hardware Virtualization Classification



What is Full Virtualization?

Unmodified Guest OS

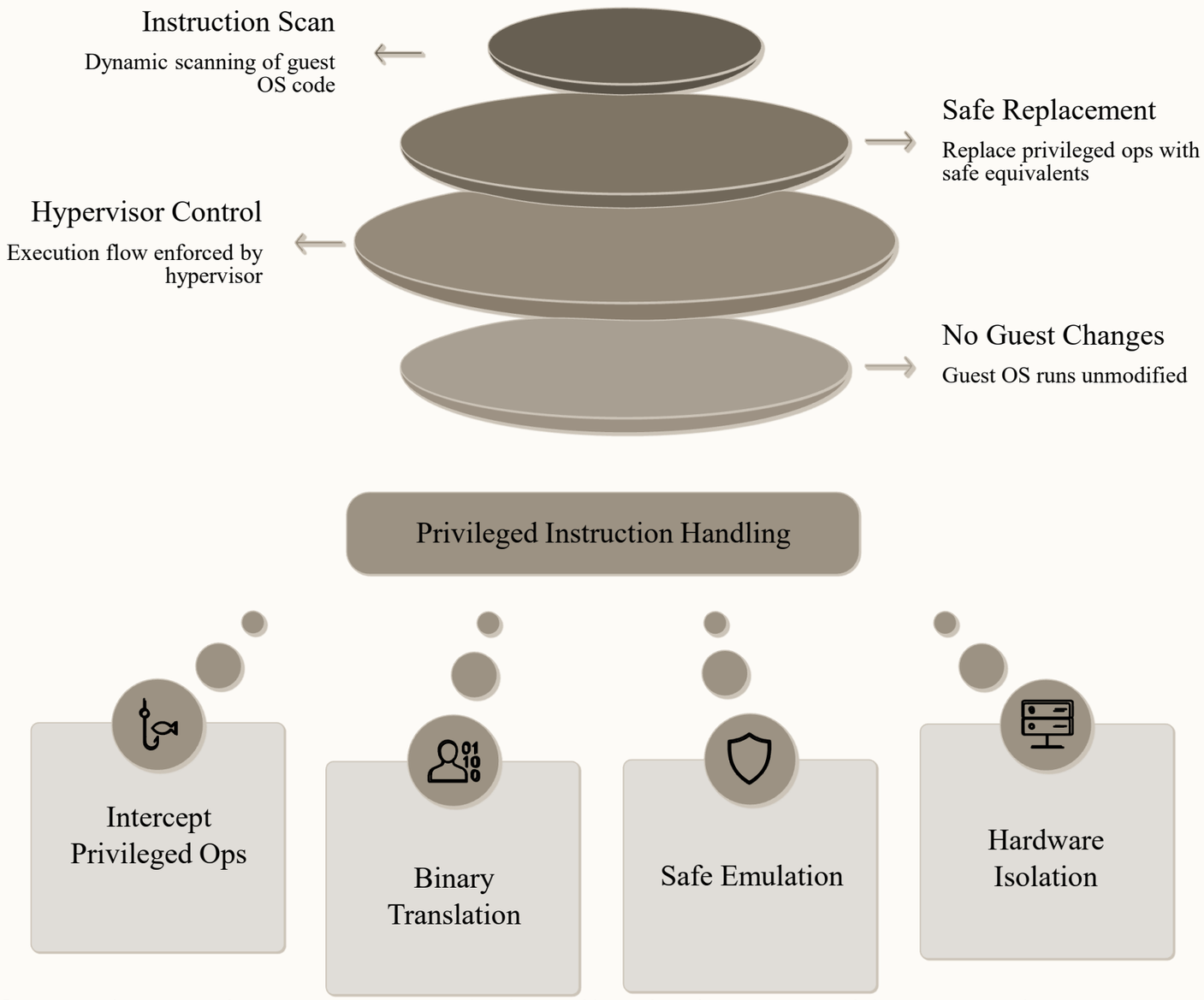
Runs without changes to its codebase



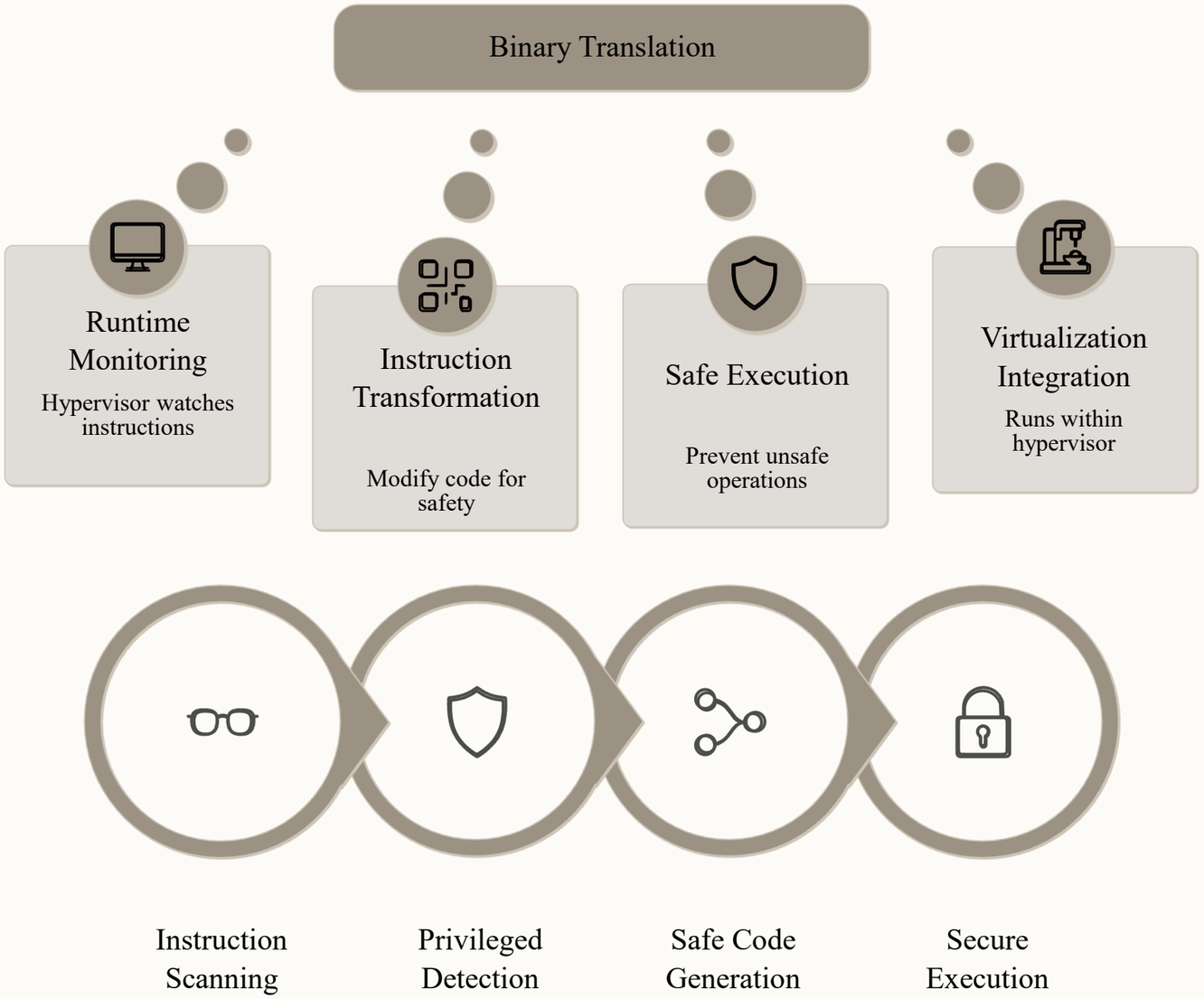
Complete Hardware Emulation

Hypervisor emulates all hardware components

Key Concept: Binary Translation



Understanding Binary Translation



How Full Virtualization with Binary Translation Works

Guest Applications

User programs execute normally

Physical Hardware

Actual CPU, memory, and devices

Guest OS (Ring 1)

Operating system runs at privileged level

Binary Translation Layer

Instruction interception and translation

VMM / Hypervisor (Ring 0)

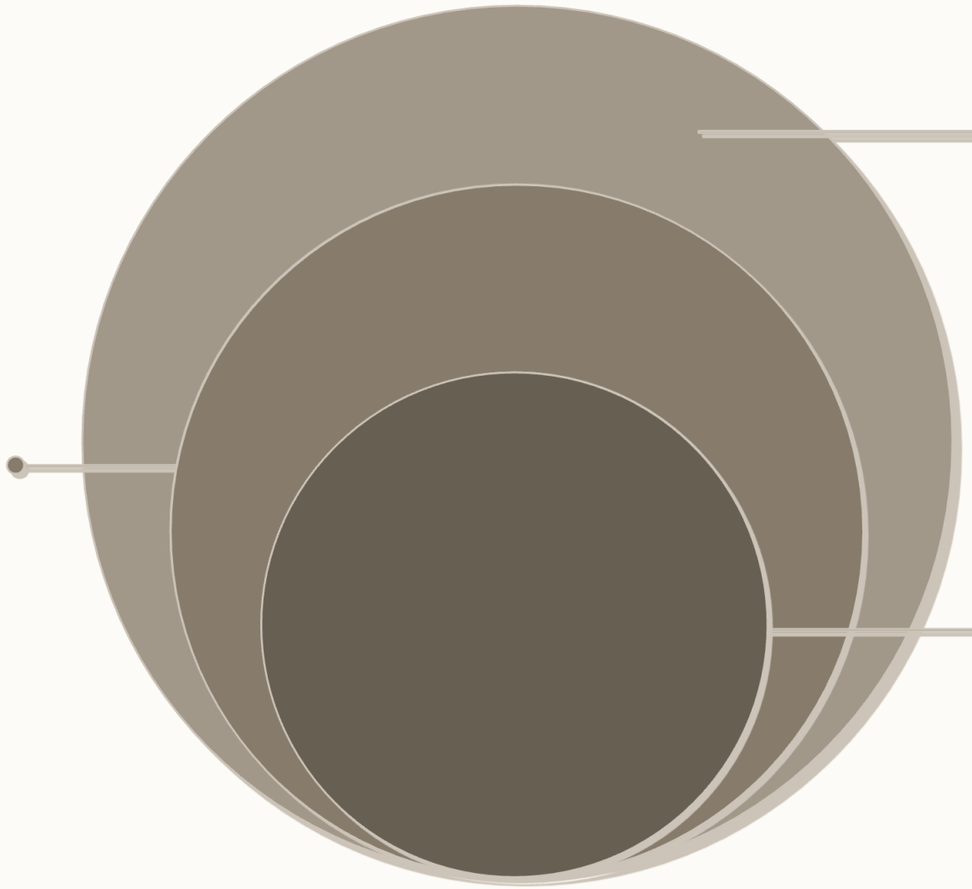
Ultimate control and hardware access

Step-by-Step Execution Flow

The binary translation process follows a systematic approach to ensure guest OS instructions execute safely while maintaining the illusion of direct hardware access.

Hypervisor Check

The hypervisor intercepts and inspects privileged instructions for security.



Translate & Return

Instructions are translated and results returned to the host.

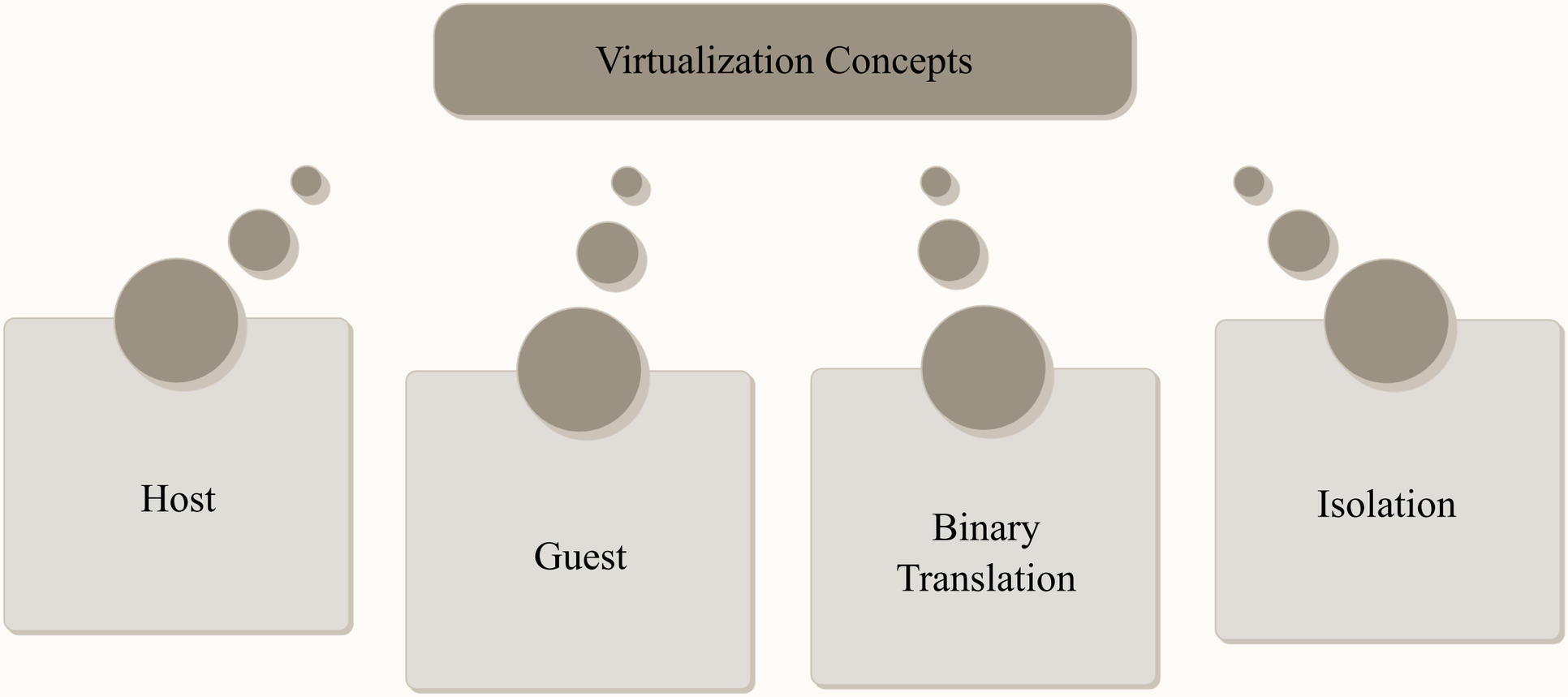
Instruction Execution

The processor runs the initial program instructions.

Real-World Example: VMware Workstation

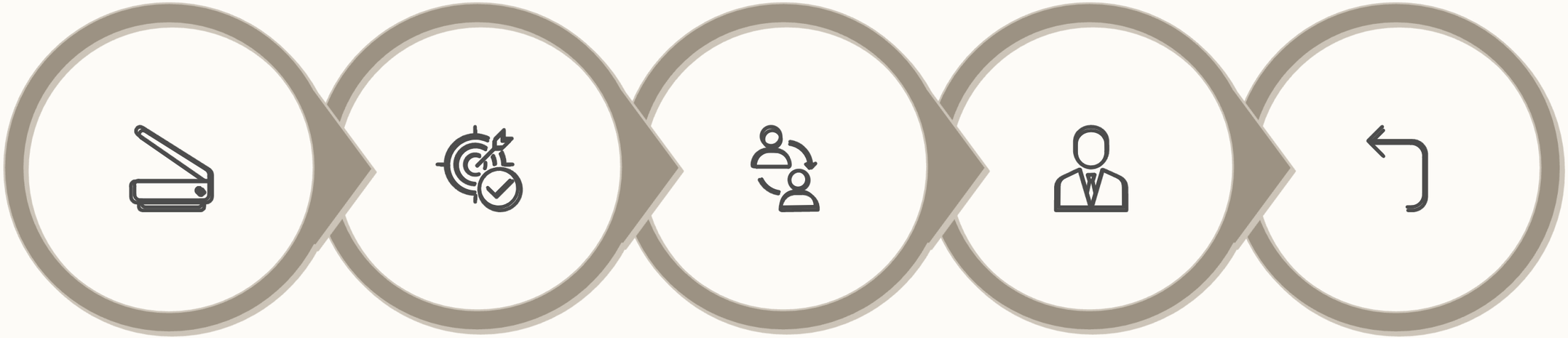
A practical demonstration of full virtualization in action shows how binary translation enables seamless multi-OS environments on single hardware platforms.

Result: Two operating systems run simultaneously, safely, and efficiently on a single physical machine.



Binary Translation Technical Details

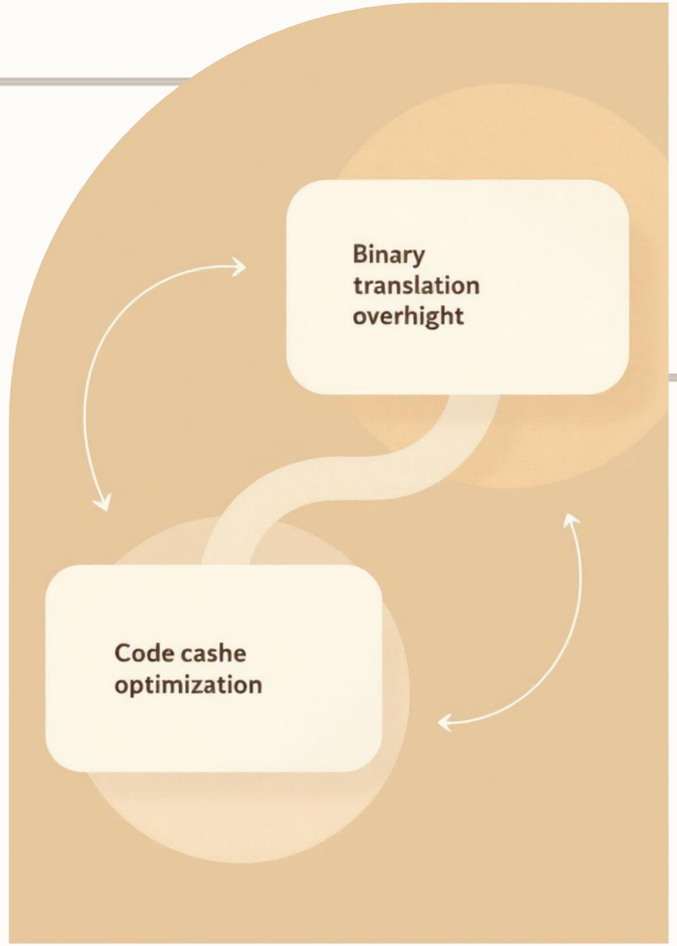
The binary translation mechanism employs sophisticated techniques to maintain performance while ensuring security and compatibility across diverse operating systems.



Performance Analysis of Full Virtualization

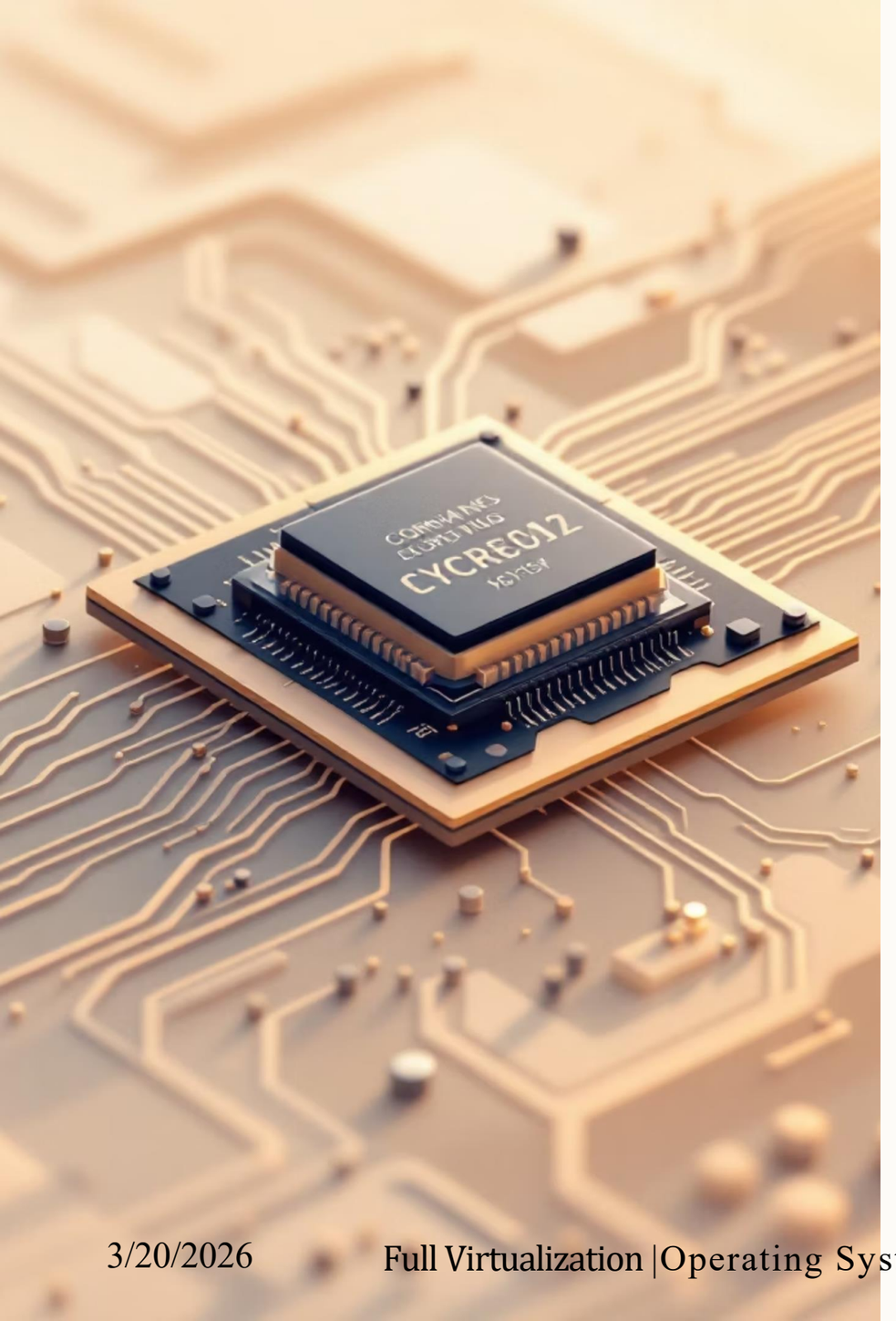
Performance Challenges

Binary translation adds runtime overhead, hurting I/O-heavy apps.



Optimization Techniques

Code caches store hot instructions to cut repeated translation cost.



Design Thinking for Virtualization

Applying structured problem-solving to enable multiple operating systems to share hardware without modification

Problem Statement
How can we allow **multiple OSes** to share hardware **without modifying them**?

Empathize: Understanding User Needs

Diverse OS Needs

Run multiple OSes for dev and legacy apps

Developer Compatibility

Ensure seamless cross-platform workflows

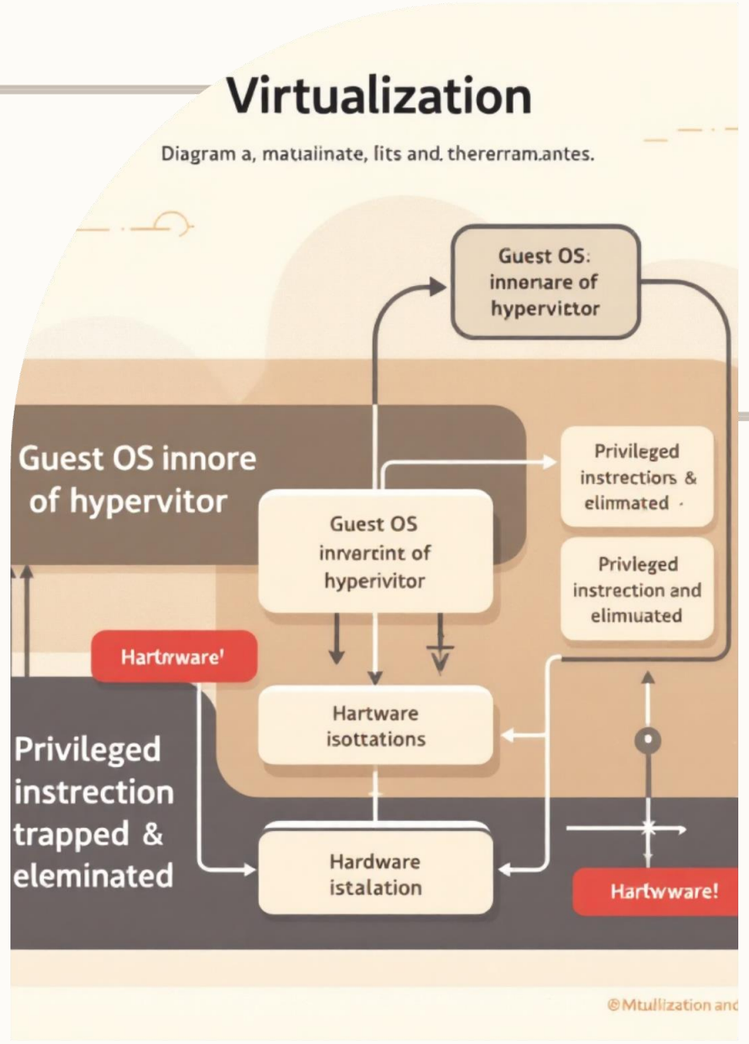
Hardware Efficiency

Maximize utilization with secure isolation

Define: The Core Problem

Challenge

Privileged instructions unsafe when OSes share hardware.



Goal

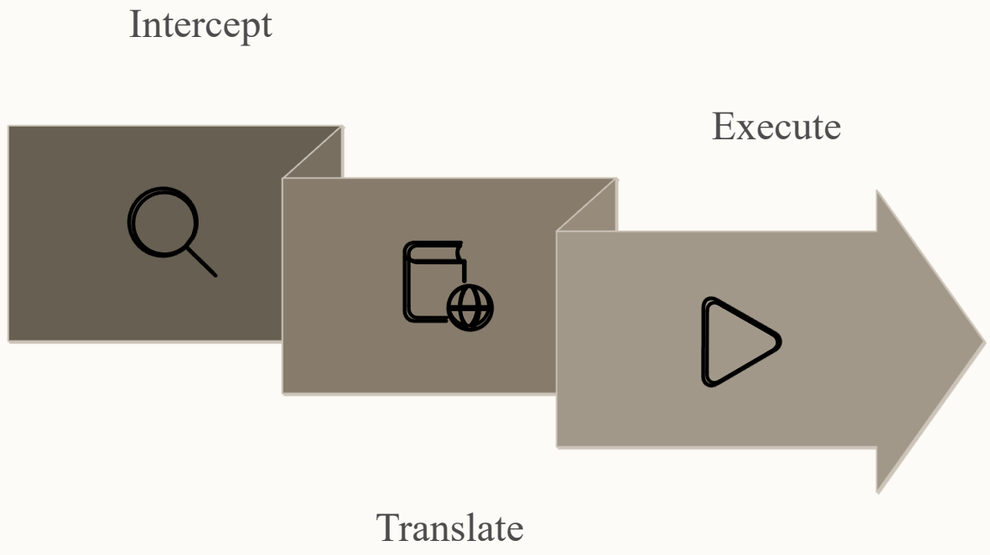
Run unmodified guests with security and high performance.

Ideate: Potential Solutions

Intercept Instructions
Capture unsafe privileged instructions before execution

Dynamic Translation
Translate intercepted instructions to safe equivalents

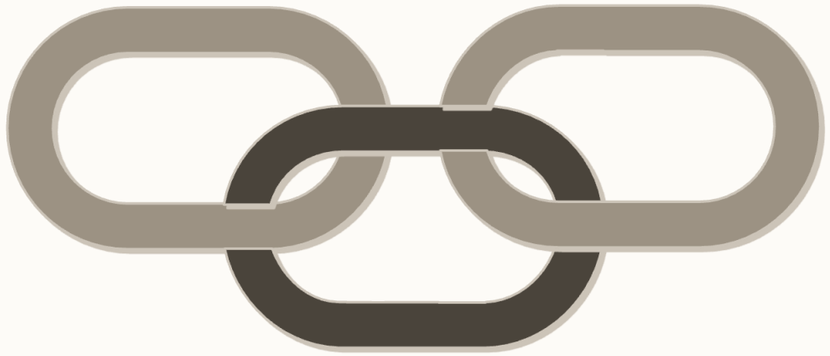
Safe Execution
Execute translated code with full hardware access control



Prototype: Building the Solution

Binary Translator

Replay Execution

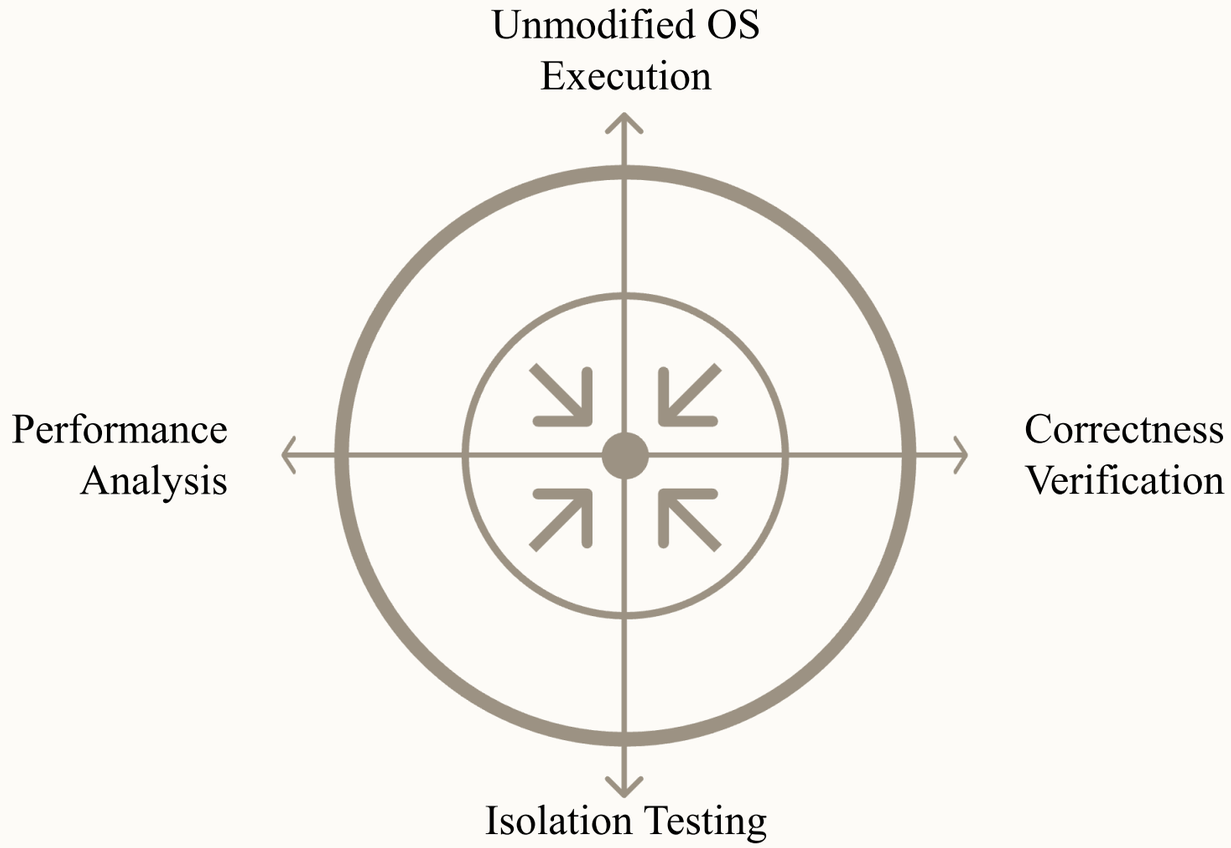


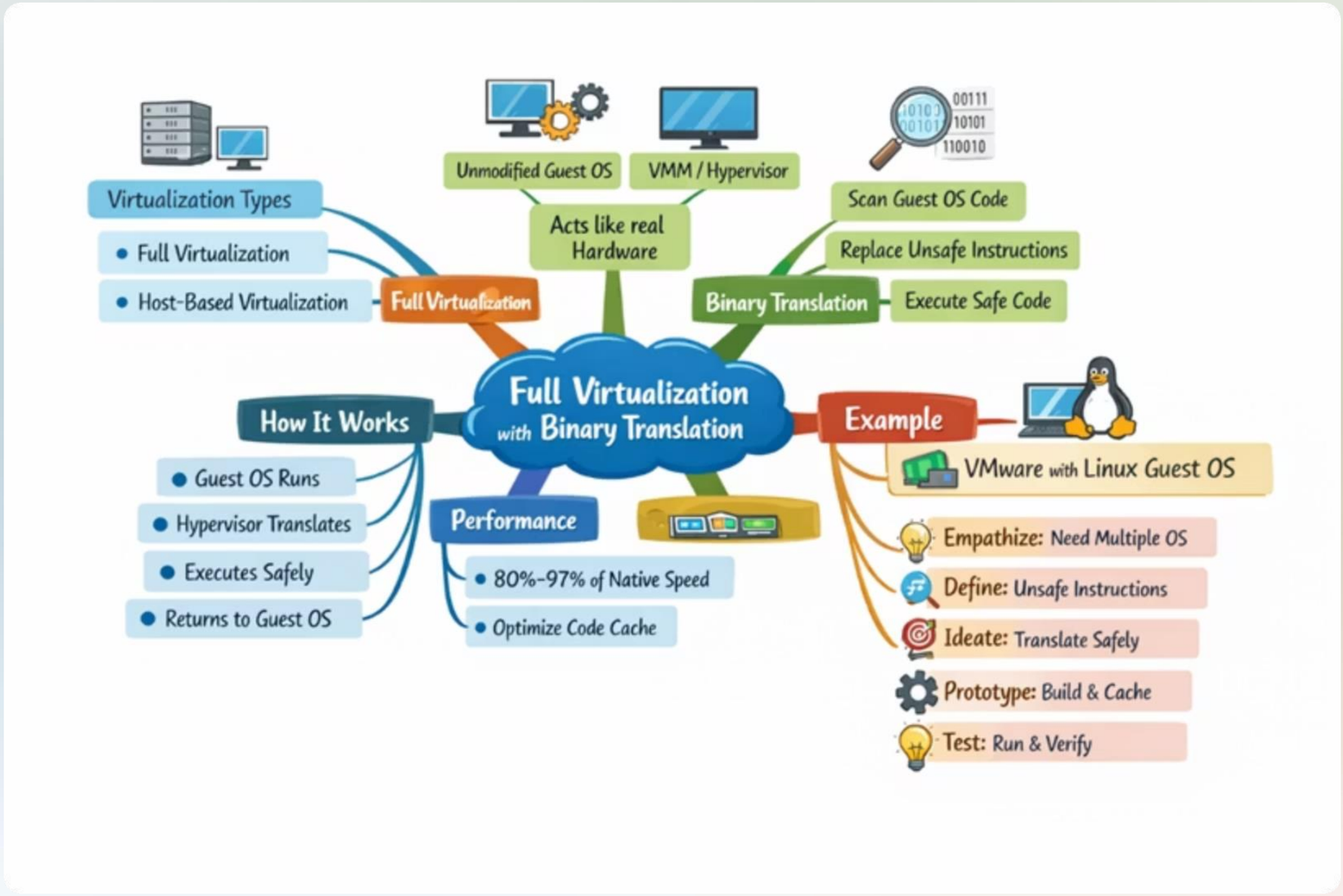
Block Caching

Test: Validation Results

- Result

  Binary translation successfully enables unmodified guest operating systems to run on virtualized hardware while maintaining full compatibility, security isolation, and acceptable performance overhead.




Full Virtualization with Binary Translation
Virtualization Types
Full Virtualization
Host-Based Virtualization
Full Virtualization
Unmodified Guest OS
VMM / Hypervisor
Acts like real Hardware
Binary Translation
Scan Guest OS Code
Replace Unsafe Instructions
Execute Safe Code
How It Works
Guest OS Runs
Hypervisor Translates
Executes Safely
Returns to Guest OS
Performance
80%–97% of Native Speed
Optimize Code Cache
Example
VMware with Linux Guest OS
Empathize: Need Multiple OS
Define: Unsafe Instructions
Ideate: Translate Safely
Prototype: Build & Cache
Test: Run & Verify

PUZZLE (For Classroom / Exam)

Puzzle Question

A user accesses Windows desktop from a low-end device, runs Excel without installing it locally, saves files to a single logical drive that spans multiple disks, and connects through a software-configured network.

Question: Identify the type of virtualization used in each case and justify your answer.

Desktop Virtualization

- Desktop access

Desktop Virtualization

Application Virtualization

- Application execution

Application Virtualization

Single logical drive over multiple disks

Storage Virtualization

Software configured network

Network Virtualization

HINT for STUDENTS:

- Desktop access
- Application execution
- Storage abstraction

- Desktop access
- Application execution
- Storage abstraction
- Network configuration