Virtualization : History and Concepts

CMPE283
Presentation 1

Agenda

- A brief history
- Core concepts
- Architectural layering

But First...

- Some basic questions...

- What is virtualization?
- Why is it important?
- Who is it important to?

Basic Definitions

- Virtualization is
    - A hardware and/or software technology
    - ...that provides for isolation of architectural layers in a computer system
    - ...wherein that isolation is performed in an efficient manner
    - …and that isolation is assumed to be inviolate

Basic Definitions

- What is an architectural layer?

Basic Definitions

- What is an architectural layer?

- An architectural layer is a minimal logical or physical collection of computing resources present in a system, separated from other such collections by purpose, form, or use

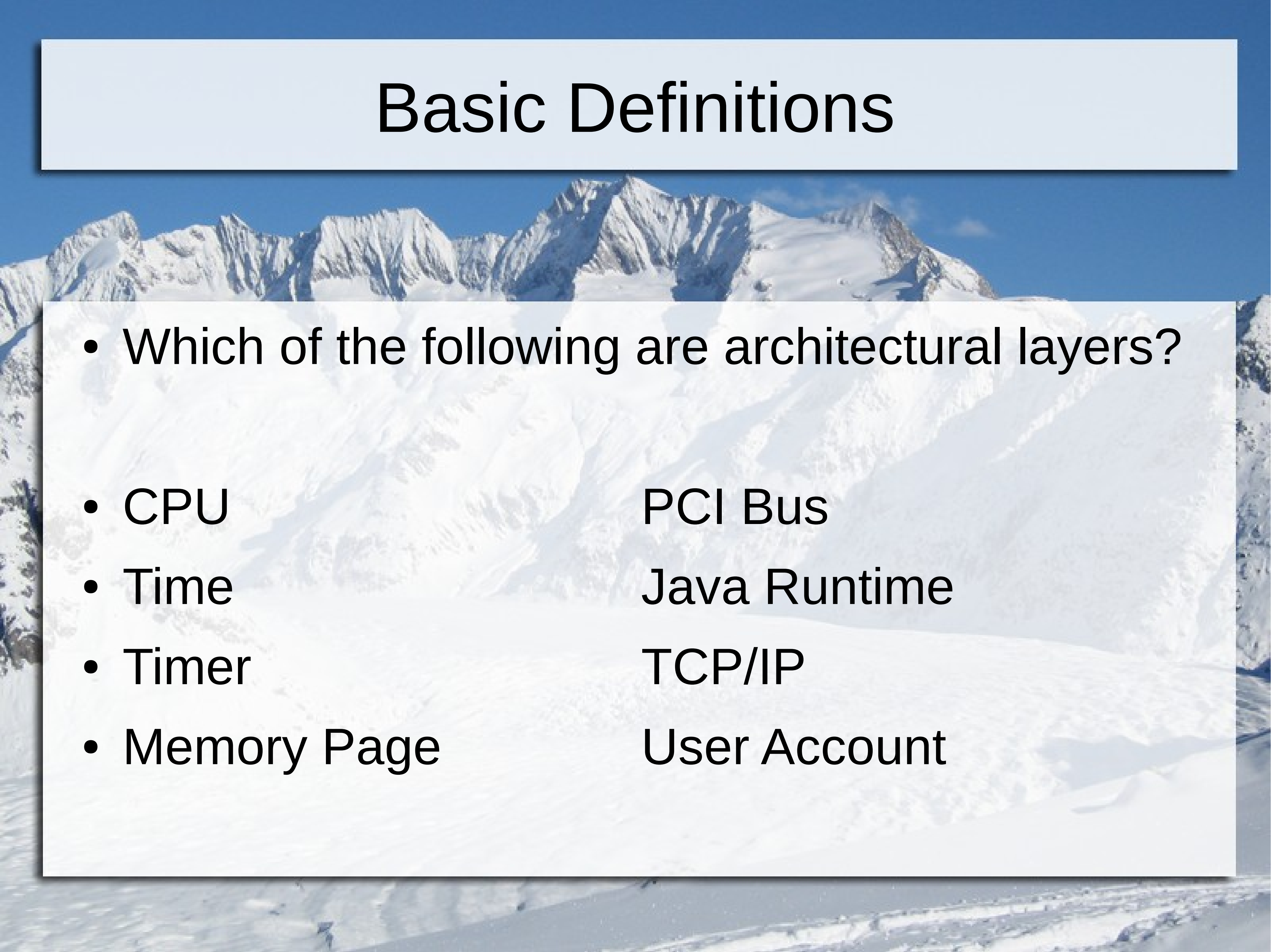
Basic Definitions

- Which of the following are architectural layers in a computing system?

- Hardware
- Operating System
- Physical Devices
- Application Software

Documents

Data



Basic Definitions

- Which of the following are architectural layers?

- CPU
- Time
- Timer
- Memory Page

PCI Bus

Java Runtime

TCP/IP

User Account

Basic Definitions

- If virtualization is a technology that enables isolation of architectural layers...
  - Why is that important? (or even desirable?)

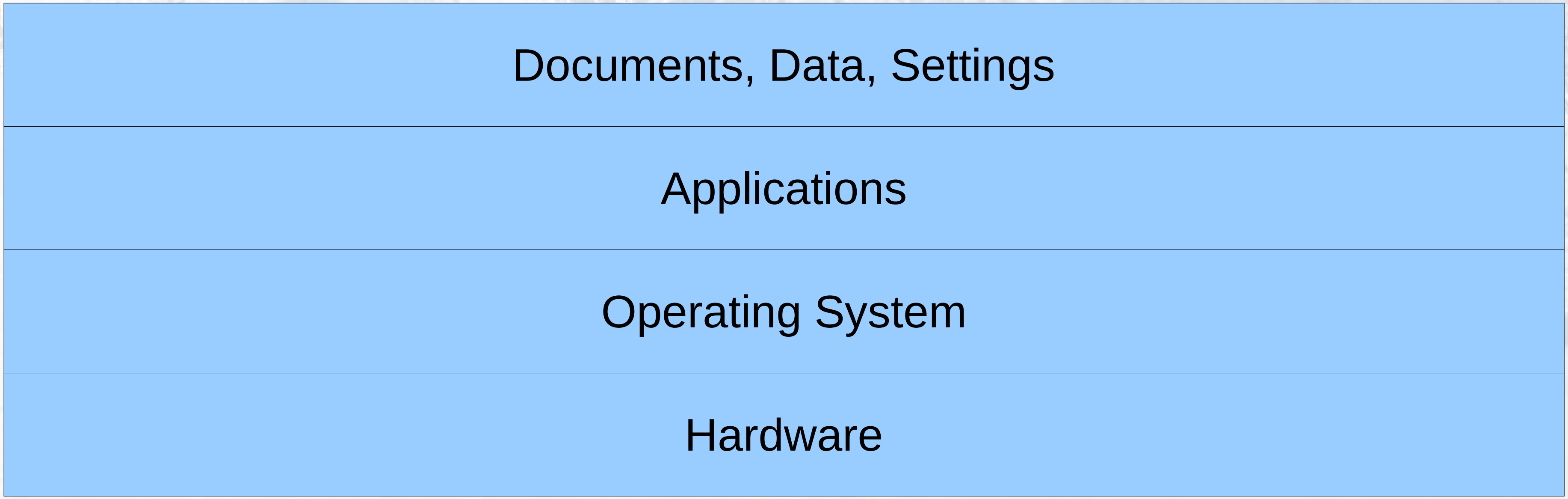
Virtualization

- What is required for resource layer isolation?

- Consider the standard or typical basic computer architecture
  - Isolating each layer from the one below it requires something “special” to be done...

Layering And Isolation

- Consider this basic architecture



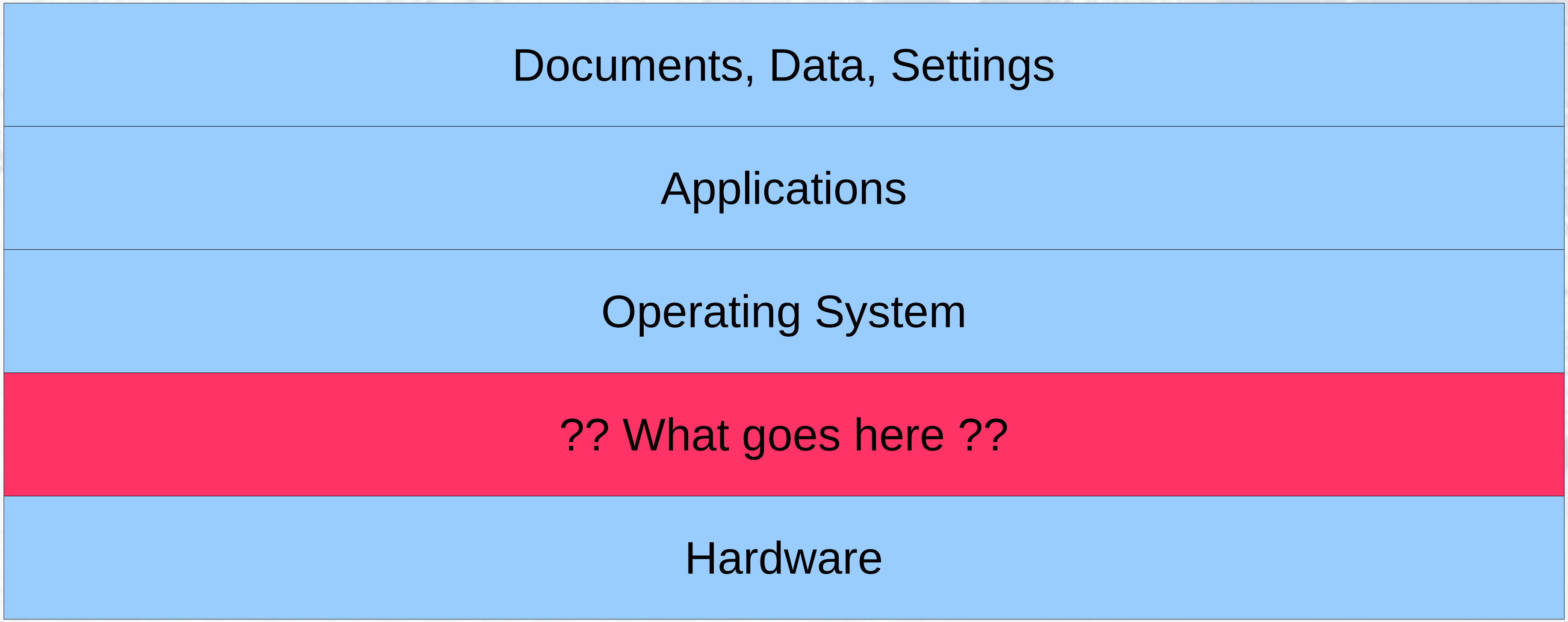
Documents, Data, Settings

Applications

Operating System

Hardware

Layering And Isolation



Documents, Data, Settings

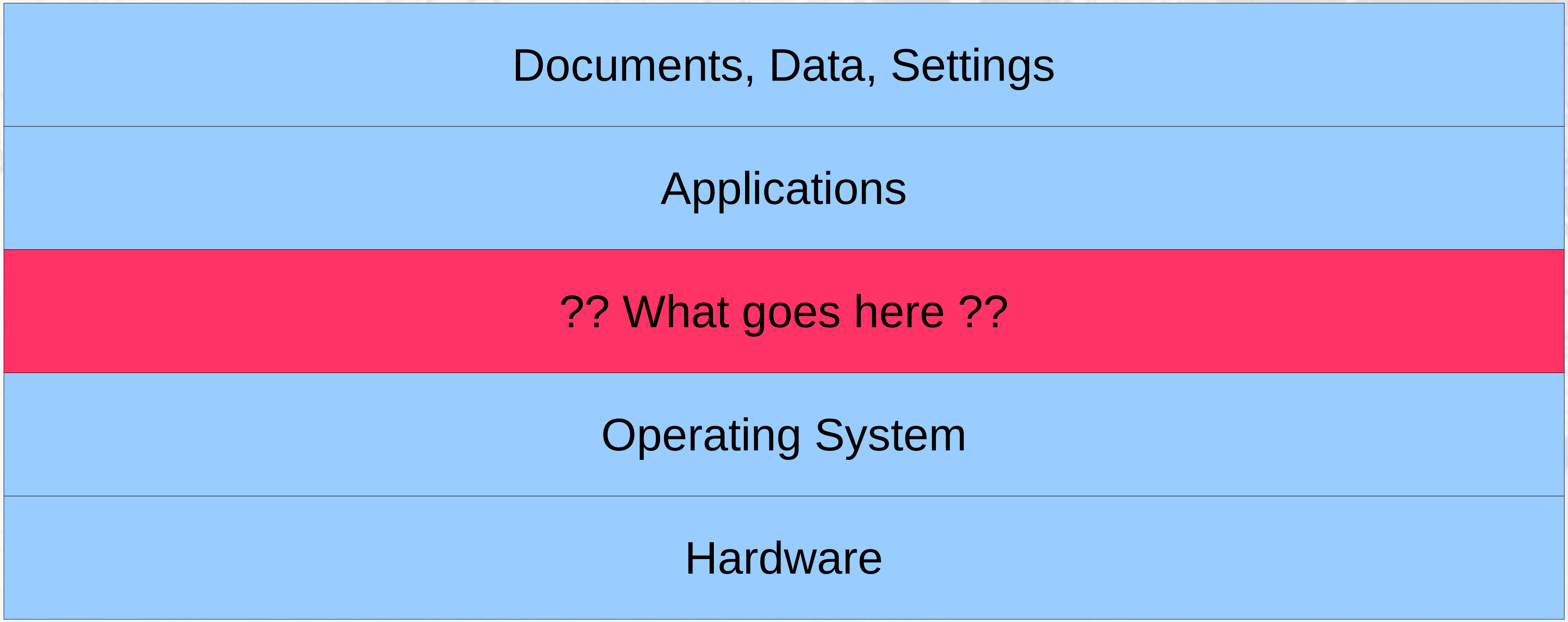
Applications

Operating System

?? What goes here ??

Hardware

Layering And Isolation



Documents, Data, Settings

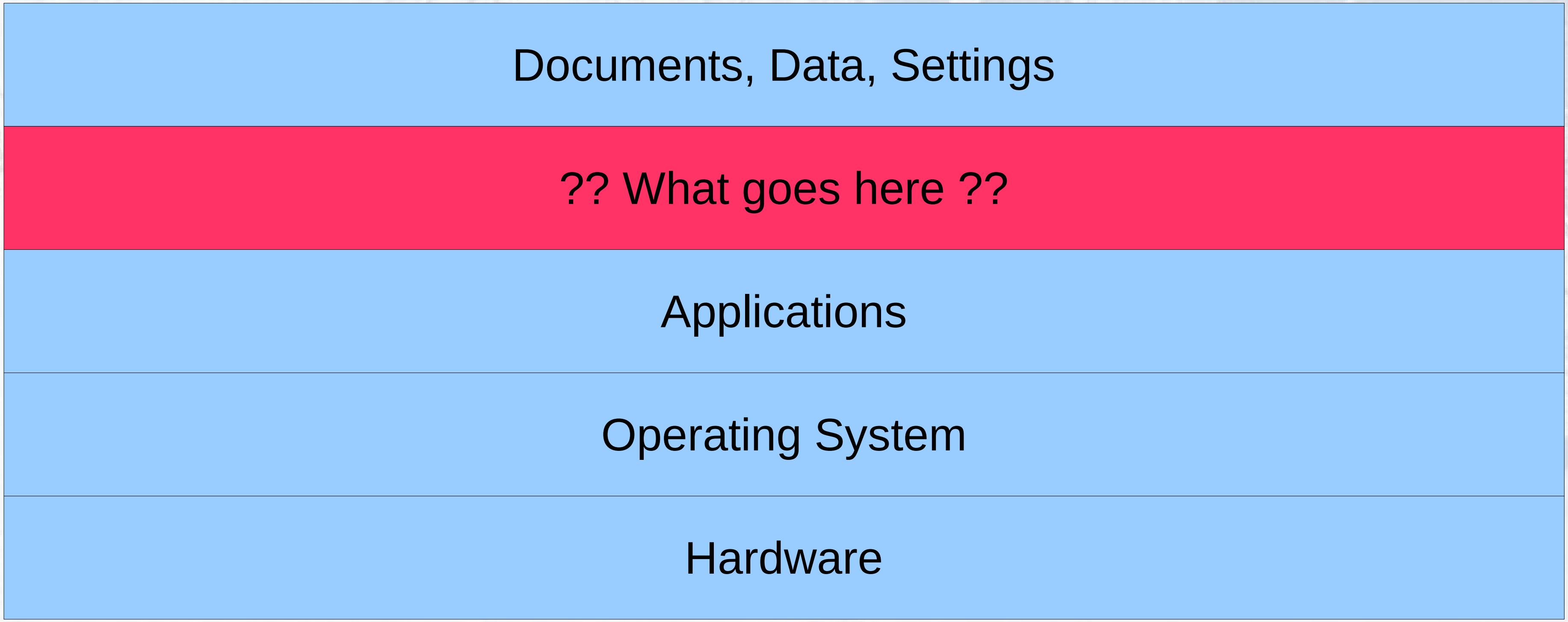
Applications

?? What goes here ??

Operating System

Hardware

Layering And Isolation



Documents, Data, Settings

?? What goes here ??

Applications

Operating System

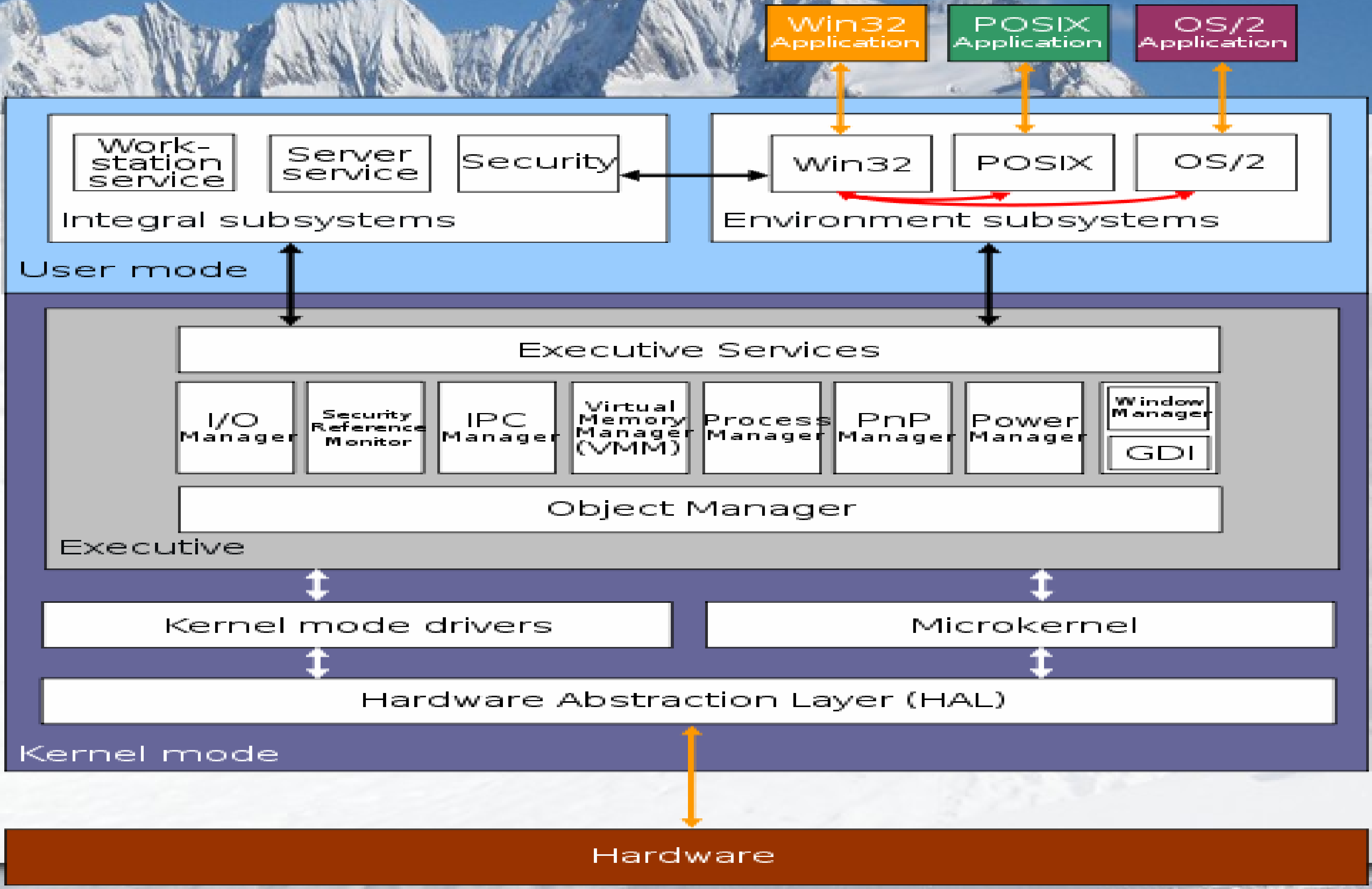
Hardware

Layering And Isolation

- Are there other organizations?

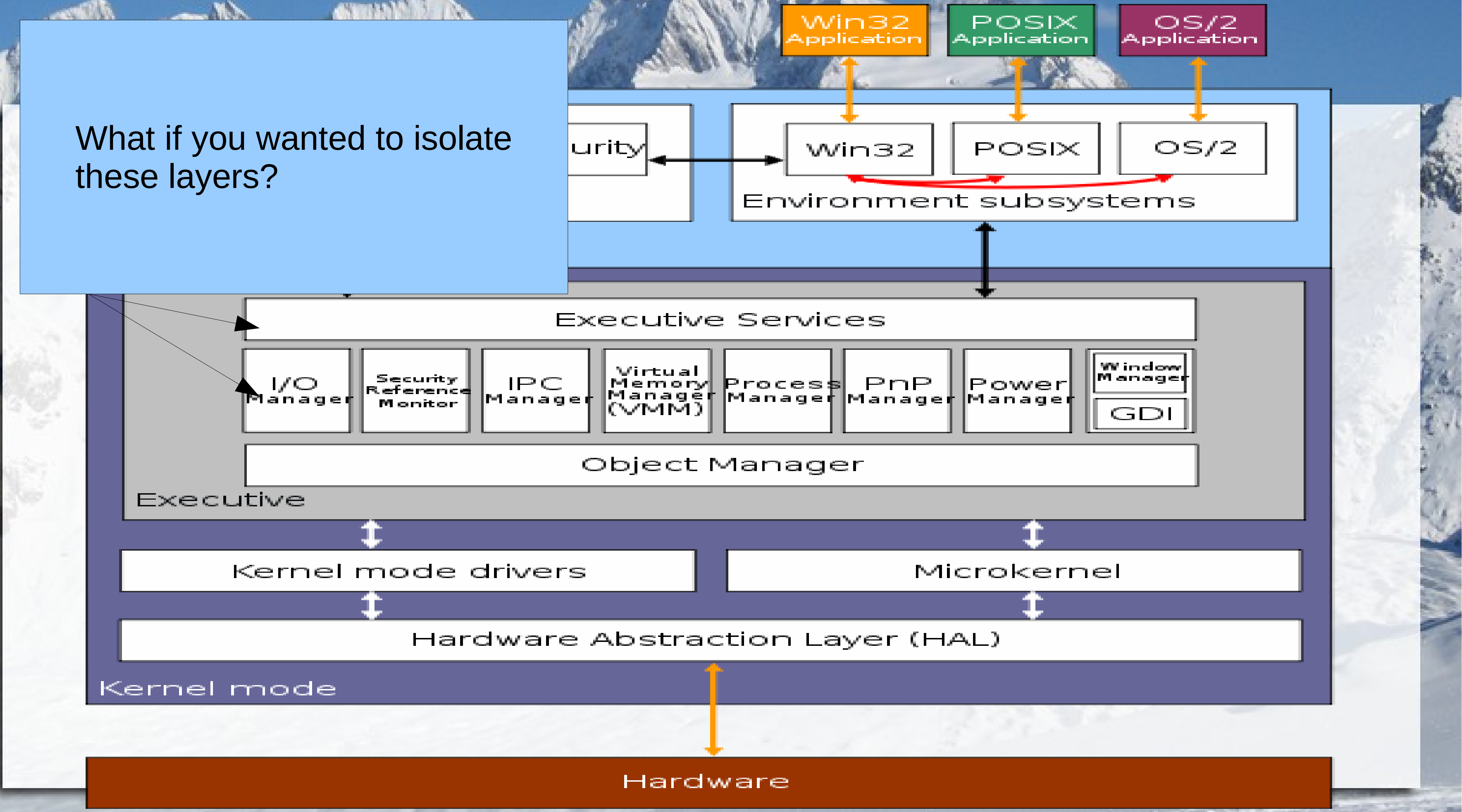
- Of course, there’s always deeper organizations
    - Consider this simplistic breakup of the “Operating System” layer from the previous slide...

Layering And Isolation



Layering And Isolation

What if you wanted to isolate these layers?



Win32 Application

POSIX Application

OS/2 Application

urity

Win32

POSIX

OS/2

Environment subsystems

Executive Services

I/O Manager

Security Reference Monitor

IPC Manager

Virtual Memory Manager (VMM)

Process Manager

PnP Manager

Power Manager

Window Manager

GDI

Object Manager

Executive

Kernel mode drivers

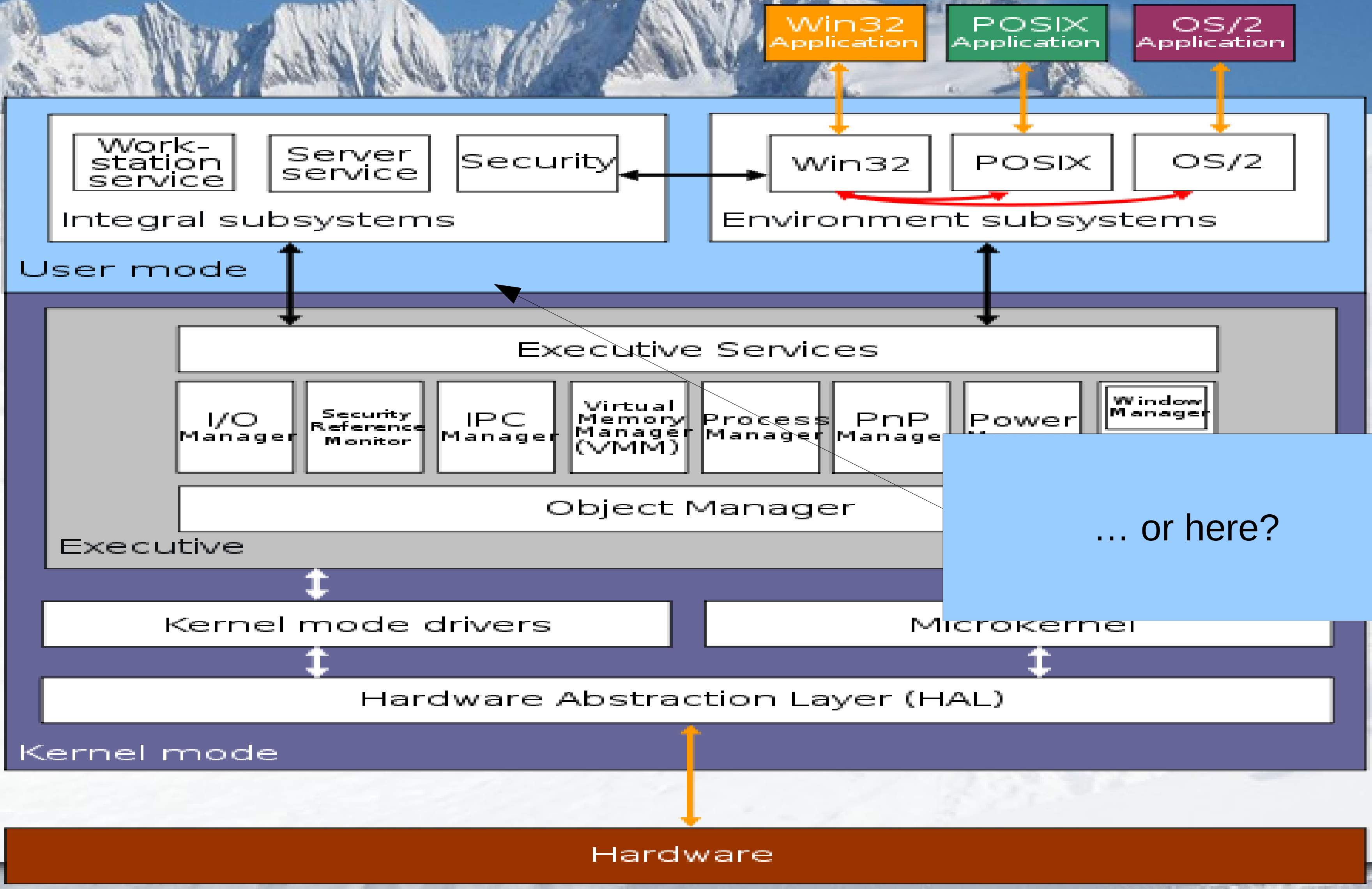
Microkernel

Hardware Abstraction Layer (HAL)

Kernel mode

Hardware

Layering And Isolation



… or here?

Layering And Isolation

- As you can probably guess, architectural layers can be *very* fine-grained
- Generally, however, certain layers are easier (and more relevant) to isolate
    - Hardware from OS
    - OS from process
    - Process from data
    - ...etc...

Layering And Isolation

- Said a different way...
  - You aren’t likely to find a virtualization technology that virtualizes things nobody cares about!
- In this class, we will focus on virtualization technologies that are relevant and interesting for today’s systems

A History Lesson...

A Brief History

- Who invented virtualization?
- There are many contenders vying for that honor
  - 1964 – IBM Research develops CP-40
  - Later evolves to CP-67, then VM
  - Others (Burroughs, Univac – all long gone) researching similar ideas

A Brief History

- IBM credited with first viable implementation
- VM (1972)
  - For use in the System/370 mainframe computer

IBM VM - System/370

A Brief History

- The S/370 is a nice history topic, but its virtualization implementation is not interesting for today's systems
- How about current system platforms?
    - Typically x86 (Intel/AMD), but could also include
        - ARM
        - PowerPC
        - MIPS
        - etc..

A Brief History

- Many modern hardware system architectures were *not* originally designed with virtualization in mind
  - This means that modern systems are not easily virtualizable
  - … which means that they do not easily support isolation of hardware layers (and for some, software layers as well)
- What does it take for a system to be designed with virtualization in mind?

P&G

- In 1974, just as the first virtualized system (IBM’s VM) was gaining popularity, a set of requirements for virtualized systems was proposed
    - Popek and Goldberg’s Virtualization
- Their paper postulated the requirements for a system to be *fully virtualizable*

P&G

- P&G provides requirements governing how CPUs behave in a virtualized environment
  - Instructions in the CPU that must be handled to provide true system isolation
- P&G also provides requirements governing interactions between the CPU/hardware and the next higher layer
  - Typically the operating system, but not always

P&G

- Privileged instructions
    - Those that trap if the processor is in user mode and do not trap if it is in system mode.
- Control sensitive instructions
    - Those that attempt to change the configuration of resources in the system.
- Behavior sensitive instructions
    - Those whose behavior or result depends on the configuration of resources (the content of the relocation register or the processor's mode).

P&G

- “For any conventional third-generation computer, a virtualized system may be constructed if the set of sensitive instructions for that computer is a subset of the set of privileged instructions.”

- This is the first P&G theorem
  - What does it mean?

P&G

- Theorem 2
  - "A conventional third-generation computer is recursively virtualizable if"
    - ...it is virtualizable and
    - ...a VMM without any timing dependencies can be constructed for it.

VMMs

- Wait, what's a VMM?
  - Virtual Machine Monitor
- VMMs
  - Abstracts VM (virtual machine) hardware
  - Typically implemented (at least partly, sometimes fully) in software
  - Provides *fidelity, safety,* and *performance* guarantees

VMMs

- VMM fidelity
  - A VM managed by a VMM should behave exactly the same as if it were running on real hardware
- VMM safety
  - The VMM must remain in complete control of system resources (virtual and physical)
- VMM performance
  - A “statistically large” number of instructions executed in the guest VM must require no VMM intervention

P&G

- Back to theorem 2 ...
  - “A conventional third-generation computer is recursively virtualizable if”
    - ...it is virtualizable and
    - ...a VMM without any timing dependencies can be constructed for it.
- What does it mean to be *recursively virtualizable*?
- … And why are *timing dependencies* mentioned?

P&G - Trivia

- Is the industry-standard x86 architecture P&G satisfiable?

x86

- Is the answer “no”...
  - There are many instructions that violate both parts of the theorem
  - This means that one needs ..assistance..
- ..or is the answer really “yes”...?

Reading

- Popek & Goldberg
  - “Formal Requirements for Third Generation Virtualizable Architectures”, Communications of the ACM, July 1974
- Intel
  - Intel 64 and IA-32 Arch. SDM (Sept 2016)
    - Volume 1
      - Chapter 2.1, Chapter 3, Chapter 5
    - Volume 3
      - Chapter 2