


CSE


Extended OS



CSE

Learning Outcomes


- An appreciation that the abstract interface to the system can be at different levels.
 - Virtual machine monitors (VMMs) provide a low-level interface
- An understanding of trap and emulate
- Knowledge of the difference between type 1 and type 2 VMMs
- An appreciation of some of the issues in virtualising the R3000



CSE

Virtual Machines


References:
 Smith, J.E.; Ravi Nair; , "The architecture of virtual machines,"
Computer , vol.38, no.5, pp. 32- 38, May 2005
 Chapter 8.3 Textbook "Modern Operating Systems"



CSE

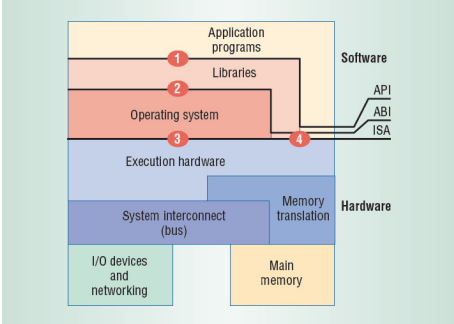
Observations

- Operating systems provide well defined interfaces
 - Abstract hardware details
 - Simplify
 - Enable portability across hardware differences
- Hardware instruction set architectures are another well defined interface
 - Example AMD and Intel both implement (mostly) the same ISA
 - Software can run on both



CSE

Interface Levels




The diagram illustrates the layers of a system and the interfaces between them. The layers are:

- Software:** Application programs, Libraries, Operating system.
- Hardware:** Execution hardware, System interconnect (bus), Memory translation, I/O devices and networking, Main memory.

Numbered labels indicate interfaces:

- 1: Between Application programs and Libraries.
- 2: Between Libraries and Operating system.
- 3: Between Operating system and Execution hardware.
- 4: Between Execution hardware and System interconnect (bus).

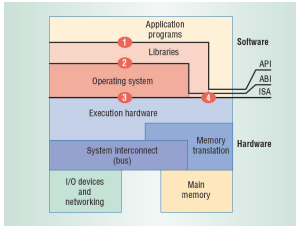
Additional labels on the right side of the diagram include API, ABI, and ISA.



CSE

Instruction Set Architecture

- Interface between software and hardware
 - label 3 + 4
- Divided between privileged and un-privileged parts
 - Privileged a superset of the un-privileged




The diagram illustrates the layers of a system and the interfaces between them. The layers are:

- Software:** Application programs, Libraries, Operating system.
- Hardware:** Execution hardware, System interconnect (bus), Memory translation, I/O devices and networking, Main memory.

Numbered labels indicate interfaces:

- 1: Between Application programs and Libraries.
- 2: Between Libraries and Operating system.
- 3: Between Operating system and Execution hardware.
- 4: Between Execution hardware and System interconnect (bus).

Additional labels on the right side of the diagram include API, ABI, and ISA.



Application Binary Interface

- Interface between programs ↔ hardware + OS
 - Label 2+4
- Consists of system call interface + un-privileged ISA

The diagram shows a layered architecture. The top layer is 'Software', containing 'Application programs' and 'Libraries'. Below it is the 'Operating system'. The bottom layer is 'Hardware', containing 'Execution hardware', 'System interconnect (bus)', 'Memory translation', 'I/O devices and networking', and 'Main memory'. Arrows indicate the flow of data and control between these components, with specific labels 1, 2, 3, and 4 marking key interfaces.

Application Programming Interface

- Interface between high-level language ↔ libraries + hardware + OS
- Consists of library calls + un-privileged ISA
 - Syscalls usually called through library.
- Portable via re-compilation to other systems supporting API
 - or dynamic linking

The diagram is identical to the one for the Application Binary Interface, showing the same layered architecture from software to hardware.

Some Interface Goals

- Support deploying software across all computing platforms.
 - E.g. software distribution across the Internet
- Provide a platform to securely share hardware resources.
 - E.g. cloud computing

OS is an extended virtual machine

- Multiplexes the “machine” between applications
 - Time sharing, multitasking, batching
- Provided a higher-level machine for
 - Ease of use
 - Portability
 - Efficiency
 - Security
 - Etc....

Abstraction versus Virtualisation

The diagram illustrates two concepts: (a) Abstraction, where multiple 'File' boxes are connected to a single physical disk icon, representing a simplified view of hardware. (b) Virtualization, where multiple 'File' boxes are connected to multiple virtual disk icons, representing the mapping of virtual resources to physical hardware.

Process versus System Virtual Machine

The diagram compares two types of virtual machines: (a) Process virtual machine, where a 'Guest' application process runs on a 'Runtime' virtualizing software layer, which runs on a 'Host' OS and hardware. (b) System virtual machine, where 'Guest' applications run on a 'VMM' (Virtualizing software) layer, which runs on a 'Host' OS and hardware.

JAVA – Higher-level Virtual Machine

- write a program once, and run it anywhere
 - Architecture independent
 - Operating System independent
- Language itself was clean, robust, garbage collection
- Program compiled into bytecode
 - Interpreted or just-in-time compiled.
 - Lower than native performance

```

    graph TD
      A[Java Code (.java)] --> B((JAVAC compiler))
      B --> C[Byte Code (.class)]
      C --> D[JVM]
      D --> E[Windows]
      D --> F[Linux]
      D --> G[Mac]
    
```

THE UNIVERSITY OF NEW SOUTH WALES

Comparing Conventional versus Emulation/Translation

```

    graph TD
      subgraph (a)
        A[HLL program] --> B[Compiler front end]
        B --> C[Intermediate code]
        C --> D[Compiler back end]
        D --> E[Object code]
        E --> F[Loader]
        F --> G[Memory Image]
      end
      subgraph (b)
        H[HLL program] --> I[Compiler]
        I --> J[Portable code]
        J --> K[VM loader]
        K --> L[Virtual memory image]
        L --> M[VM interpreter/compiler]
        M --> N[Host instructions]
      end
    
```

THE UNIVERSITY OF NEW SOUTH WALES

Aside: Just In-Time compilation (JIT)

THE UNIVERSITY OF NEW SOUTH WALES

Issues

- Legacy applications
- No isolation nor resource management between applets
- Security
 - Trust JVM implementation? Trust underlying OS?
- Performance compared to native?

THE UNIVERSITY OF NEW SOUTH WALES

Is the OS the “right” level of extended machine?

- Security
 - Trust the underlying OS?
- Legacy application and OSs
- Resource management of existing systems suitable for all applications?
 - Performance isolation?
- What about activities requiring “root” privileges

THE UNIVERSITY OF NEW SOUTH WALES

Virtual Machine Monitors

- Provide scheduling and resource management
- Extended “machine” is the actual machine interface.

THE UNIVERSITY OF NEW SOUTH WALES

IBM VM/370

- CMS a light-weight, single-user OS
- VM/370 multiplex multiple copies of CMS

Virtual 370s

CMS CMS CMS

VM/370

370 Bare hardware

I/O instructions here → CMS CMS CMS ← System calls here

Trap here → VM/370 ← Trap here

THE UNIVERSITY OF NEW SOUTH WALES

Advantages

- Legacy OSES (and applications)
- Legacy hardware
- Server consolidation
 - Cost saving
 - Power saving
- Server migration
- Concurrent OSES
 - Linux - Windows
 - Primary - Backup
 - High availability
- Test and Development
- Security
 - VMM (hopefully) small and correct
- Performance near bare hardware
 - For some applications

THE UNIVERSITY OF NEW SOUTH WALES

Taxonomy of Virtual Machines

Process VMs System VMs

Same ISA Different ISA Same ISA Different ISA

Multiprogrammed systems Dynamic translators Classic system VMs Whole-system VMs

Same-ISA dynamic binary optimizers High-level-language VMs Hosted VMs Codesigned VMs

THE UNIVERSITY OF NEW SOUTH WALES

What is System/161?

THE UNIVERSITY OF NEW SOUTH WALES

Excel Word Mplayer Apollon

Windows Linux ...

Type 1 hypervisor

(a)

Guest OS process

Guest OS

Host OS process

Type 2 hypervisor

Host operating system

(b)

Figure 1-29. (a) A type 1 hypervisor. (b) A type 2 hypervisor.

THE UNIVERSITY OF NEW SOUTH WALES

Type 1 Hypervisor

- Hypervisor (VMM) runs in most privileged mode of processor
 - Manage hardware directly
 - Also termed classic..., bare-metal..., native...
- Guest OS runs in non-privileged mode
 - Hypervisor implements a virtual kernel-mode/virtual user-mode
- What happens when guest OS executes native privileged instructions?

Excel Word Mplayer Apollon

Windows Linux ...

Type 1 hypervisor

(a)

THE UNIVERSITY OF NEW SOUTH WALES

Type 2 Hypervisor

- Hypervisor runs as user-mode process above the privileged host OS
 - Also termed hosted hypervisor
- Again, provides a virtual kernel-mode and virtual user-mode
- Can leverage device support of existing host OS.
- What happens when guest OS execute privileged instructions?

THE UNIVERSITY OF NEW SOUTH WALES

Gerald J. Popek and Robert P. Goldberg (1974). "Formal Requirements for Virtualizable Third Generation Architectures". *Communications of the ACM* 17 (7): 412–421.

- Sensitive Instructions
 - The instructions that attempt to change the configuration of the processor.
 - The instructions whose behaviour or result depends on the configuration of the processor.
- Privileged Instructions
 - Instructions that trap if the processor is in user mode and do not trap if it is in system mode.
- Theorem
 - Architecture is virtualisable if sensitive instructions are a subset of privileged instructions.

THE UNIVERSITY OF NEW SOUTH WALES

Approach: Trap & Emulate?

THE UNIVERSITY OF NEW SOUTH WALES

Virtual R3000???

- Interpret
 - System/161
 - slow
 - JIT dynamic compilation
- Run on the real hardware??

THE UNIVERSITY OF NEW SOUTH WALES

Issues

- Privileged registers (CP0)
- Privileged instructions
- Address Spaces
- Exceptions (including syscalls, interrupts)
- Devices

THE UNIVERSITY OF NEW SOUTH WALES

mfc0 r1, CO_Cause

CG

THE UNIVERSITY OF NEW SOUTH WALES

