Architecture of Virtual Machines

Summarized by: Michael Riera

9/17/2011
Papers in discussion

• Recommendations for Virtualization Technologies in High Performance Computing
  – Nathan Regola, Center for Research Computing
    • Notre Dame
  – Jean-Christophe Ducom, Center for Research Computing,
    • Notre Dame

• The Architecture of Virtual Machines
  – James E. Smith
    • University of Wisconsin-Madison
  – Ravi Nair
    • IBM T.J. Watson Research Center

• Understanding Performance Interference of I/O Workload in virtualized Cloud Environments
  • Xing Pu, Ling Liu, Yiduo Mei, Sankaran Sivathanu, Younggyun Koh, Calton Pu
    – Georgia Tech, Beijing Institute of Technology, P.R. China, Xi’an Jiaotong University, P.R. China
Agenda

• Purpose

• What does it mean to Virtualize
  – Abstraction layers
  – Interfaces
  – Classic Architecture
  – Cross-Compiling (Process VM)
  – Hosted/Non-Hosted VM (System VM, aka. VMM, Hypervisors)
  – Terminology
Purpose

• The purpose of this paper is to give the reader a fundamental broad knowledge on the main components of a Virtual Machine (VM), and its interaction with the Hardware.
• This papers give a high level description of what abstracting (virtualizing processes/OS) the hardware and the ISA layer from a running process.
Virtualizing

• Virtualizing provides a way to:
  – Multiplex under utilized resources such as:
    • CPU Cycles
    • Memory
    • Storage Access
  – Allow multiple users to compute on the same hardware as if they were in sole control
  – A way to translate an instruction set architecture (ISA) to another ISA without needing to recompile, or migrate High level applications
Virtualizing software uses the file abstraction as an intermediate step to provide mapping between the virtual and real disks.
Figure 2. Computer system architecture. Key implementation layers communicate vertically via the instruction set architecture (ISA), application binary interface (ABI), and application programming interface (API).
Process & System VM

• A **process** VM is a virtual platform that executes an individual process.
  – It’s created when needed and terminated when process is completed.
  – Provides a an ABI or API environment
  – Cross Compiling
  – Ex. Sun Microsystem Java VM, Microsoft Common Language (.NET)

• A **system VM, (aka Virtual Machine Monitors)**
  – Provides a complete, persistent system environment that supports an OS along with its many user process.
  – The VMM emulates the hardware ISA
  – Provides resource management for multiple VMs

• **A process or system that runs on the VM is coined as a guest.**
• The Platform that supports the VM is coined as the **host**.
• The virtualizing software that implements a process VM is typically called “**runtime software**”.
Figure 3. Process and system VMs. (a) In a process VM, virtualizing software translates a set of OS and user-level instructions composing one platform to those of another. (b) In a system VM, virtualizing software translates the ISA used by one hardware platform to that of another.
Process & System VM

• Emulator and dynamic binary translators
  – Interpret
    • An interpreter program that fetches, decodes, and emulates the execution of a guest instruction (too SLOW)
  – Dynamic binary translator
    • Converts guest instructions to host instructions in blocks rather than instruction by instruction and saves them for reuse in a software cache
Figure 4. High-level-language environments. (a) Conventional environment where platform-dependent object code is distributed. (b) HLL VM environment where a platform-dependent VM executes portable intermediate code.
Process & System VM

- Co-designed VM
  - Motivation:
    - Functionality and portability
    - For more specific proprietary ISA that are uncomment
    - Appears as hardware, the VMM resides in a region of memory concealed from all conventional software.
    - Includes a binary translator
Process & System VM

**Figure 5. Virtual machine taxonomy.** Within the general categories of process and system VMs, ISA simulation is the major basis of differentiation.