Schedule

- 4/21 Midterm exam
- 4/28 Cloud computing
- 5/12 Computer networks
- 5/19 Database systems
- 5/26 Artificial intelligence
- 6/2 Security
- 6/9 Project presentation
- 6/16 Final exam
Midterm Exam

- 4/21 (Thursday), 16:00 – 17:30
- Chemistry Bldg. #330110

- Closed-book exam

- Scope:
  - Chap. 1, 2, 3, 4, 5, 7, 10
Operating Systems

Jin-Soo Kim (jinsookim@skku.edu)
Computer Systems Laboratory
Sungkyunkwan University
http://csl.skku.edu
What is an OS?
What is an OS?

- Application view
- System view
- Implementation view
Application View

- Provides an execution environment for running programs
- Provides an abstract view of the underlying computer system
  - Processors → Processes, Threads
  - Memory → Address space (virtual memory)
  - Storage → Volumes, Directories, Files
  - I/O Devices → Files
  - ...

System View

▪ Manages various resources of a computer system

▪ Goals
  • Sharing
  • Protection
  • Fairness
  • Efficiency
  • ...

Resources
  ▪ CPU
  ▪ Memory
  ▪ I/O devices
  ▪ Queues
  ▪ Energy
  ▪ ...

Implementation View

- Highly-concurrent, event-driven software
System Calls
OS Structure

User Application

C Library (libc)

System Call Interface

Kernel

Arch-dependent kernel code

Hardware Platform

User Space

Kernel Space
Kernel Internals

User space

shell
ls
shell
ps

trap

Kernel space

System Call Interface

File System Management
I/O Management (device drivers)
Hardware Control (Interrupt handling, etc.)
Memory Management

Process Management
- scheduler
- IPC
- synchronization

Protection

Hardware
Process Management
Process

- An instance of a program in execution

- Two key abstractions
  - Logical control flow
  - Private address space

- How are these illusions maintained?
  - Process executions interleaved (multitasking)
  - Address space managed by virtual memory
Logical Control Flows

- Each process has its own logical control flow.
Context Switching

- **Context switching**
  - Control flow passes from one process to another via a context switch.

![Diagram](image-url)
Process State Transition
Virtual Memory
Private Address Spaces

Private address spaces include:

- Kernel virtual memory (code, data, heap, stack)
- Memory mapped region for shared libraries
- Run-time heap (managed by malloc)
- Read/write segment (.data, .bss)
- Read-only segment (.init, .text, .rodata)
- Unused

Address ranges:
- 0x08048000: user stack (created at runtime)
- 0x08048000 - 0xffffffff: memory invisible to user code
- 0xc0000000 - 0xffffffff: memory mapped region for shared libraries
- 0x0 - 0xc0000000: brk

Memory segments:
- Read-only segment: loaded from the executable file
- Read/write segment: managed by malloc

%esp (stack pointer)
Example

- What happens if two users simultaneously run this application?

```c
#include <stdio.h>
int n = 0;

int main () {
    n++;
    printf ("n = %d, &n = 0x%08x\n", n, &n);
}
```

```
% ./a.out
n = 1, &n = 0x08049508
% ./a.out
n = 1, &n = 0x08049508
```
Physical Addressing
Virtual Addressing

CPU

Virtual Addresses

Page Table

Physical Addresses

Disk

Memory
Virtual Memory

- Allows to run programs much larger than the available physical memory size
- Simplifies memory management
- Provides memory protection
### Protection

#### Page Tables

**Process A:**
- VP 0: Read? Yes, Write? No, Physical Addr PP 9
- VP 1: Read? Yes, Write? Yes, Physical Addr PP 4
- VP 2: Read? No, Write? No, Physical Addr XXXXXXX

**Process B:**
- VP 0: Read? Yes, Write? Yes, Physical Addr PP 6
- VP 1: Read? Yes, Write? No, Physical Addr PP 9
- VP 2: Read? No, Write? No, Physical Addr XXXXXXX
Virtual Machines
Virtual Machine

▪ A fully protected and isolated copy of the underlying physical machine’s hardware (definition by IBM)

▪ Virtual Machine Monitor (or Hypervisor)
  • A thin software layer that sits between hardware and OS
  • Virtualizes and manages all hardware resources
VMware ESX Server

Core Virtualization Layer
Xen Architecture

Hardware (host computer)

domo

Control Plane Software

XenoLinux

Virtual Devices Driver

domo

Control I/F

VirtualDevices

Xen (hypervisor)

Guest OS

domU

User Software

XenoLinux

Virtual Devices Driver

domU

User Software

XenoBSD

Virtual Devices Driver

domU

User Software

XenoBSD

Virtual Devices Driver
Why Virtual Machines?

- Creates the illusion of multiple VMs.
- Strong isolation between VM instances.
- Software compatibility.
- Server consolidation
- Convenient environment for debugging OSes