

COMP3301/7308 – Semester 2 2009

Lecture 1
Course overview, basic concepts

School of Information Technology and Electrical Engineering
The University of Queensland

The course

- Operating systems – at all levels
- Focus on Minix
 - free
 - documented
 - textbook
- NOT a programming course
- BUT, will be expected to program (in C) a lot

2

Course Profile

- Available On-Line
- Please read carefully, especially learning activities and assessment sections.
- COMP7308 –slightly different grading scheme, otherwise same as COMP3301

- The course is similar to previous years in overall theme, but presentation is somewhat different.

3

Staff

- Lecturer:
 - Dr John Williams
- Tutor:
 - Matthew Williams

- *Thanks to Dr Philip Machanick who taught the course for many years, these slides are adopted from his lecture notes of 2007.*

4

Contact

- 2L1T1P1W
- Lectures/Tutorials
 - Fridays 2-5
- Pracs
 - Thurs 9-11 / Thurs 11-1
- “Workshop” has been amalgamated with the practicals as the opportunity for more detailed discussion.

5

Practicals

- There will be some practicals (usually early in each assignment cycle) which provide a structured introduction to each area.
- START TODAY/TOMORROW
- Some later practical sessions will be simply to provide an opportunity to work on assignments, with the tutor available for help.
- Help also available via newsgroup.

6

COMP3301

Teaching plan

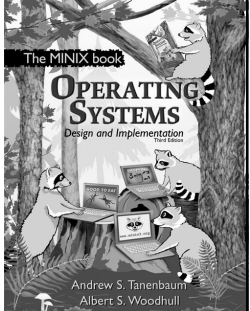
- Similar to textbook:
 - Processes
 - IO
 - Memory management
 - Filesystems

7

COMP3301

The textbook

- A. Tanenbaum and A. Woodhull "Operating Systems Design and Implementation" 3rd Edition
- Covers Minix 3



10

COMP3301

Assessment

- Assignments x 3 @ 25% each
 - Implementing changes in Minix
 - First assignment available next week.
- Final exam @ 25%
 - Theory
 - Understanding of material (as opposed to the practical programming assignments)
 - Exact format still to be decided – maybe on-line in the labs.

9

COMP3301

Functions of an operating system

- What does an operating system actually do?

Banking system	Airline reservation	Web browser	} Application programs
Compilers	Editors	Command interpreter	
Operating system			} System programs
Machine language			
Microarchitecture			} Hardware
Physical devices			

10

COMP3301

Extended Machine

- Controlling IO devices is complex – moving the disk arm, finding appropriate sectors, etc.
 - OS abstracts this process into *read* and *write*
 - Presents a simpler extended machine to the programmer

11

COMP3301

Resource Manager

- Range of devices need to be controlled and managed
 - Different processes all try to access the printer at the same time – OS controls the queue
- Control memory space, networks etc
- OS manages all these resources

12

COMP3301

OS History

- 50's and 60's
 - Punch cards read onto tape
 - Primitive OS reads tape and completes jobs in batches
- OS/360
 - Ran on all IBM machines
 - First to introduce multi-programming
 - Spooling input and output

13

COMP3301

OS History

- Timesharing introduced 1962
 - Each user has a terminal
- MULTICS
 - Plug in to computing power
 - not really commercially successful
 - compiler was late and not very good

14

COMP3301

OS History

- Minicomputers
 - PDP-1: 4K of 18-bit words for \$120 000
 - UNIX initially developed on similar machine
- Microcomputers
 - CP/M
 - MS-DOS
 - GUI

15

COMP3301

Minix

- Written for students to understand!
- Modular
- Version 3 is current – substantially different to Version 2
- Linux was a response to Minix!

16

COMP3301

OS concepts

- Processes
- Files
 - System calls provide services to do with each of these
- The shell
 - interface to system – not really operating system

17

COMP3301

Processes

- Program being executed
 - has address space
 - registers, stack pointer etc.
 - file pointers
 - all stored in process table
- System calls create and terminate processes

18

COMP3301

Processes

- Shell often creates processes
- These are *children* of the shell
- They can create more processes
- Communication is via IPC
- Alarm signals
- Associated with UID

19

COMP3301

Files

- Directories
- Processes have working directory associated with them
- Permissions
- File descriptor
- Mounting
- Special files
- Pipes

20

COMP3301

The Shell

- MINIX shell similar to other UNIX shells
- Pipe
- Redirection
- Background

21

COMP3301

System calls

- Allow kernel functions
- When process makes system call, trap to kernel mode (hardware dependent)
- Kernel executes the functionality

22

COMP3301

System calls for processes

- Managing processes:
 - fork
 - waitpid
 - execve
 - exit
- See book for details:
(T&W: 1.4.1)

23

COMP3301

System calls for Signaling

(T&W: 1.4.2)

- sigaction
- sigprocmask
- sigpending
- kill
- alarm
- pause

24

COMP3301

System calls for file management (T&W 1.4.3)

- open
- close
- read
- write
- lseek
- stat
- fstat
- pipe

25

COMP3301

System calls for directory management (T&W 1.4.4)

- mkdir
- rmdir
- link
- mount
- sync
- chdir

26

COMP3301

System calls for protection and time management

- Protection (T&W 1.4.5)
 - chmod
 - setuid
 - getuid
 - etc!
- Time(T&W1.4.6)
 - time
 - stime
 - times

27

COMP3301

Operating system structure

- System calls are what the programmer sees
- POSIX defines this
- Says nothing about the structure
- What options?

28

COMP3301

Monolithic

- All procedures linked into one big executable
- All can call all

29

COMP3301

Layered

- Provides more structure:
 - eg:

Layer	Function
5	The operator
4	User programs
3	Input/output management
2	Operator-process communication
1	Memory and disk management
0	Processor allocation and multiprogramming

From THE system (Dijkstra, 1968)

Virtual Machines

- Provide multiple virtual machines to upper layers
- Can be further operating systems
- Read about VM/370

31

Exokernels

- Similar to virtual machine
- Each machine allocated a subset of resources

32

Client-server model

- Functionality provided by user-space servers
- Kernel handles communication between clients and servers
- How do user programs do IO?
- Why?
- Why not?

33

Summary

- Functions of operating systems
- Concepts (processes, files etc)
- System calls
- Operating system structure
- We assume that you have encountered all of these things before in some detail in COMP2303.

34

Tutorial Questions for Next Week

- We'll go through the answers to these in lectures next week.
- You should attempt these questions before the lecture.
- Wikipedia, and other web-based resources, will have many of the answers you want.
- Tutorial material is examinable.

35

Tutorial 1

- Q1. (Tanenbaum 1.7) Which of the following should be allowed only in kernel mode? Explain your answer in each case.
- Disable all interrupts
 - Read the time-of-day clock
 - Set the time-of-day clock
 - Change the memory map
- Q2. What structure do the following operating systems have? What are some of the reasons for why they have been designed that way?
- Mach
 - Linux
 - L4
- Q3. (Tanenbaum 1.25) The client-server model is popular in distributed systems. Why? Can it also be used in a single-computer system? Explain.
- Q4. What does it mean to say that an operating system is POSIX-compliant? Is MINIX3 POSIX compliant?

36

COMP3301

Minix and emulators

School of Information Technology and Electrical Engineering
The University of Queensland

COMP3301

Intro to Minix

- How Minix is structured
- Using Minix
- Practical tips on getting Minix running

38

COMP3301

History of Minix

- AT&T licensed UNIX
- Minix Version 1: 1987
 - modular
 - readable (?)
 - commented
 - no hard disk required
- Version 2: 1997
 - substantial changes in supported hardware

39

COMP3301

History of Minix

- Version 3:
 - 2005
 - restructured kernel
 - modularity and reliability
 - kernel is under 4000 lines...
 - should be understandable

40

COMP3301

Structure of Minix

- Follows client-server model
- Structured into four layers:
 - Kernel
 - Device drivers
 - Server processes
 - User processes

41

COMP3301

Structure of Minix

Layer							
4	Init	User process	User process	User process	...	User processes	} User mode
3	Process manager	File system	Info server	Network server	...	Server processes	
2	Disk driver	TTY driver	Ethernet driver	...		Device drivers	} Kernel mode
1	Kernel			Clock task	System task	Kernel	

42

COMP3301

Kernel layer

- Handles messages between processes
- Access IO ports
- Interrupt handling
 - (Requires kernel mode)
- Clock task
- Provides kernel calls
 - Done by system task
- Mostly C, some assembly

43

COMP3301

Device drivers

- Request IO
 - Done through system task
- Can't access IO directly!

44

COMP3301

Servers

- Provide services to user processes
- process manager (pm)
 - provides system calls like fork, exec etc
 - manages memory
- file system (fs)
 - implements read, open etc
- information server
 - provides status information
- reincarnation server
 - starts/restarts device drivers

45

COMP3301

User processes

- User programs
- Compilers, editors etc
- Can't access kernel directly
- Must use appropriate services and drivers

46

COMP3301

Kernel v System Calls

- Kernel calls provided at the lowest level
 - never used by user processes
 - low-level
- System calls are what the user application uses
 - high-level

47

COMP3301

Booting Minix

- What happens when we turn the power on?
 - In PCs, BIOS defines where to look
 - loads *boot* program
 - this copies boot image into memory
 - this is essentially the kernel
- See details of what gets started when in book

48

Emulators (Virtualisers)

- Why use emulators?
 - Don't crash real hardware
 - Emulate hardware we don't have
 - Controllable environment
- What emulators?
 - QEmu
 - Bochs
 - VMWare

49

Installing Minix

- Use VMWare Player
- Minix CD image (from website)
- specify image in VMWare Player
- Boot!
- Follow instructions for setup
- Details in prac

50

The source code

- Structure
- Where to look for things
- Go to /usr/src
- All the source is under here

51

/usr/include

- /usr/include v /usr/src/include
- sys/
- minix/
- ibm/
- Textbook has a lot of details
 - Read it!

52

/usr/src

- kernel/
- drivers/
- servers/
- lib/
- tools/
- boot/

53

Recompiling the kernel

- Go to /usr/src/tools
- Run make to see options
 - image
 - install
 - hdbot
 - clean
 - and others!

54

Recompiling the kernel

- make install
- boot it
 - Default is to boot the newest image
 - If it's faulty, specify different image in boot monitor