From last time

An OS may contain managers for Devices, Network, Filestore, Memory, & Processes. Which would be in an OS for:
– A process control computer with a sensor for monitoring, an actuator for control, and a network connection for reporting to and receiving commands from a control centre?
– A dedicated, network-based filing machine or ”file server”?

– A computer dedicated to controlling the communications passing between two networks; that is, a ”gateway”?
– An autonomous lap-top personal computer?
– A single-user workstation with services available across a network?

– A machine dedicated to managing and answering queries on a database?

MANCHESTER 1824
The University of Manchester

COMP25111: Operating Systems

Lecture 4: Operating System Concepts

Will Toms

School of Computer Science, University of Manchester

Autumn 2016

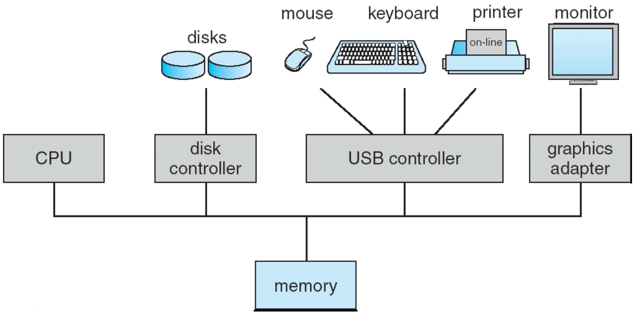
Overview & Learning Outcomes

Overview of (multi-programming) OS
– functions & components

Processes

Protection

Components of a simple PC



– details of devices are hidden from Apps
– several things can be happening at once

What does an Operating System Do?

Manage Resources:
– multiple devices → deal with concurrency
– sharing
– protection

Provide services:
– multiple Apps → provide concurrency
– abstraction
e.g. filestore, not disk drive
e.g. variable size stack
e.g. reliable network connection

Process = Thread + Address Space

Process: a program in execution (not a program on the disk)

Address Space: all memory locations the process can use

Thread: “of execution” – sequence of instructions obeyed

Multi-threading: multiple threads within the same process

Many processes exist at any time

Windows XP: `<CTRL><ALT><DEL>`

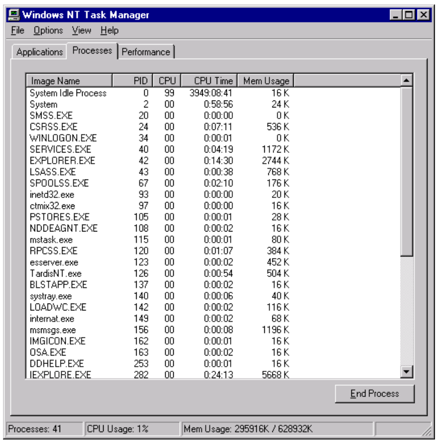
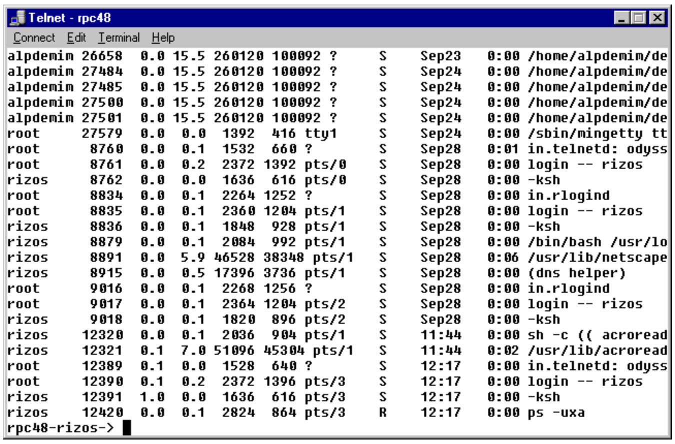


Image Name	PID	CPU	CPU Time	Mem Usage
System Idle Process	0	99	3949:08:41	16 K
System	2	00	0:58:56	24 K
SMSS.EXE	20	00	0:00:00	0 K
CSRSS.EXE	24	00	0:07:11	536 K
WINLOGON.EXE	34	00	0:00:01	0 K
SERVICES.EXE	40	00	0:04:19	1172 K
EXPLORER.EXE	42	00	0:14:30	2744 K
LSASS.EXE	43	00	0:00:38	768 K
SPOOLSS.EXE	67	00	0:02:10	176 K
inetd32.exe	93	00	0:00:00	20 K
ctmix32.exe	97	00	0:00:00	16 K
PSTORES.EXE	105	00	0:00:01	28 K
NDDEAGNT.EXE	108	00	0:00:02	16 K
mstask.exe	115	00	0:00:01	80 K
RPCSS.EXE	120	00	0:01:07	384 K
esserver.exe	123	00	0:00:02	452 K
TardisNT.exe	126	00	0:00:54	504 K
BLSTAPP.EXE	137	00	0:00:02	16 K
systray.exe	140	00	0:00:06	40 K
LOADWC.EXE	142	00	0:00:02	116 K
internat.exe	149	00	0:00:02	68 K
msmsgs.exe	156	00	0:00:08	1196 K
IMGICON.EXE	162	00	0:00:01	16 K
OSA.EXE	163	00	0:00:02	16 K
DDHELP.EXE	253	00	0:00:01	16 K
IEXPLORE.EXE	282	00	0:24:13	5668 K

Processes: 41 | CPU Usage: 1% | Mem Usage: 295916K / 628932K

Many processes ctd.

Linux: `ps uxa`



```
alpdemim 26658  0.0 15.5 260120 100092 ?     S    Sep23   0:00 /home/alpdemim/de
alpdemim 27484  0.0 15.5 260120 100092 ?     S    Sep24   0:00 /home/alpdemim/de
alpdemim 27485  0.0 15.5 260120 100092 ?     S    Sep24   0:00 /home/alpdemim/de
alpdemim 27500  0.0 15.5 260120 100092 ?     S    Sep24   0:00 /home/alpdemim/de
alpdemim 27501  0.0 15.5 260120 100092 ?     S    Sep24   0:00 /home/alpdemim/de
root     27579  0.0  0.0  1392  416 tty1     S    Sep24   0:00 /sbin/mingetty tt
root      8760  0.0  0.1  1532  660 ?        S    Sep28   0:01 in.telnetd: odyss
root      8761  0.0  0.2  2372 1392 pts/0    S    Sep28   0:00 login -- rizos
rizos     8762  0.0  0.0  1636  616 pts/0    S    Sep28   0:00 -ksh
root      8834  0.0  0.1  2264 1252 ?        S    Sep28   0:00 in.rlogind
root      8835  0.0  0.1  2360 1204 pts/1    S    Sep28   0:00 login -- rizos
rizos     8836  0.0  0.1  1848  928 pts/1    S    Sep28   0:00 -ksh
rizos     8879  0.0  0.1  2084  992 pts/1    S    Sep28   0:00 /bin/bash /usr/lo
rizos     8891  0.0  5.9 46528 38348 pts/1   S    Sep28   0:06 /usr/lib/netscape
rizos     8915  0.0  0.5 17396 3736 pts/1    S    Sep28   0:00 (dns helper)
root      9016  0.0  0.1  2268 1256 ?        S    Sep28   0:00 in.rlogind
root      9017  0.0  0.1  2364 1204 pts/2    S    Sep28   0:00 login -- rizos
rizos     9018  0.0  0.1  1820  896 pts/2    S    Sep28   0:00 -ksh
rizos    12320  0.0  0.1  2036  904 pts/1    S    11:44   0:00 sh -c (( acroread
rizos    12321  0.1  7.0 51096 45304 pts/1   S    11:44   0:02 /usr/lib/acroread
root     12389  0.1  0.0  1528  640 ?        S    12:17   0:00 in.telnetd: odyss
root     12390  0.1  0.2  2372 1396 pts/3    S    12:17   0:00 login -- rizos
rizos    12391  1.0  0.0  1636  616 pts/3    S    12:17   0:00 -ksh
rizos    12420  0.0  0.1  2824  864 pts/3    R    12:17   0:00 ps -uxa
rpc48-rizos-> 
```

Address Spaces

e.g. ARM/MU0 assembler addresses start at 0

But, several programs can be in memory at the same time - each assuming this

OS may pause a running program, swap it out of memory & later swap it back to somewhere different

Relocation - how to make each program think it has sole use of memory

Relocation example: a C program

```
int x;
main (int argc, char *argv[]) {
  x= atoi(argv[1]);
  printf("%d %p\n", x, &x);
}
```

e.g. `./a.out 7` from two different Linux shells
both output: `7 0x8049678`

Different programs seem to use the same address

Virtual Machine

OS provides “Virtual Machine”
– more convenient abstraction than real machine
– Apps think they use the hardware on their own

Virtual Machine enforces Protection:
– System v. Program
– Program v. Program

OS needs hardware support – execution mode:
– User mode
– System (Privileged, Supervisor) mode

Privileged Operations

OS components run in System mode

OS runs Apps in User mode

H/W prevents certain operations in User mode:

– memory operations?

– CPU allocation?

– I/O operations?

– file operations?

– network operations?

System call

How do Apps use protected resources?

System call: interface between Apps & OS

like method/function call – parameters, caller waits for result

via "gatekeeper" mechanism (H/W + OS)
– turns on System mode
– calls OS routine from list
– parameters etc. checked
– action performed
– returns to User mode

Details vary between OSs, underlying concepts similar

System Call example

Unix "read" has 3 parameters: the file, where to put the data, how many bytes to read

```
read(int fd, char *buf, int num_bytes);
```

Not the **C library function**:

```
fread(void *ptr, size_t size, size_t n, FILE *stream);
```

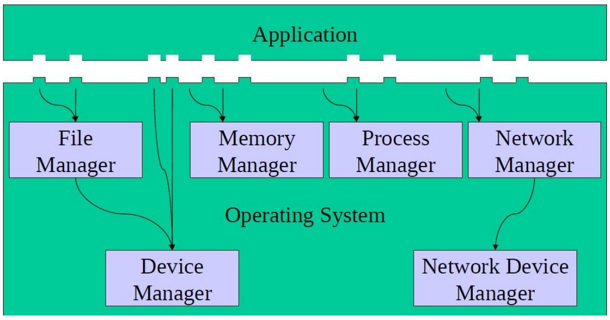
– library functions can do more
– not all library functions correspond to system calls

Many languages do not allow system calls to be made directly

OS Components

A system so large and complex can be created by partitioning into smaller pieces

Most OSs have different structures



OS Components provide services

Process Management: creation, deletion, CPU allocation, ...

Memory Management: Allocate and deallocate memory space; Keep track of what parts of the memory are being used, ...

Device (I/O) Management: read & write bytes

File (and Secondary Storage) Management: ...

Network Management: ...

User interface: GUI, command line interpreter (shell)

User/App use services

e.g. User types `run myprog` (just `myprog` in Unix)

– read command (command interpreter/shell)
– find program file (how big?)
– allocate memory
– read file into memory
– find libraries
– start myprog running
– finish "cleanly"

Also: accounting, security, error detection/reporting, ...

Engineering an OS...

Monolithic systems (no structure - the "big mess")

Layered approach (bottom = H/W, highest = U.I)
Layers selected so each only uses functions, operations & services of lower layers.
Lower layers ("**kernel**") contain most fundamental functions to manage resources.

Big OS Kernels have problems (complexity, debugging)
several Mbytes (linux 2-3)

Microkernels keep only minimal functionality in the OS

Summary of key points

Process = Thread + Address Space

Protection: Virtual Machine
– H/W support: User mode v. System mode
– System calls for Priviledged operations

OS Structure
– Components (Managers): Process, Memory, I/O, File, ...
– Layered, Kernel, Micro-Kernel

Next time: Process Management

Your Questions

For next time

Which of the following operations would you expect to be privileged (available only in System mode) & which available in User mode?
– halt the processor?
– system call?
– write an absolute memory location?
– load register from memory?
– disable interrupts?
– load stack pointer?
– write to segment or page not present in memory?
– change memory management register value?
– write to Program Status Register?
– write to interrupt vector table?

Exam Questions

Why do computers typically have two modes of operation, namely user mode and system mode (also known as supervisor or kernel or privileged mode)? (2 marks)

Explain briefly what is a system call (2 marks)

What does it mean to say that a system is constructed using the ”micro-kernel approach”? (2 marks)

Glossary

Device
Resource
Concurrency
Process
Address space
Thread
Multi-threading
Relocation
Virtual Machine
System/Supervisor/Priviledged mode
User mode
System call
Library function
Manager
Monolithic OS
Layered OS
OS Kernel
Microkernel OS

Reading

OSC/J: Chapters 1 & 2

MOS: Sections 1.5-1.11 (skim through the system call details)

(both books use some concepts in these sections that will be clarified later on)