COMP 3511
Operating Systems

Lab 02

Outline

- Review Questions
- Nachos start and exercise_1
- Nachos Introduction
- Notes on Nachos
- Nachos exercise_2

Nachos start and exercise_1

- Step 1: login
  - If the computer is turned on, you would see the logon screen. Type your CSD account and password to login. If successfully login, you will see the prompt below:

    csl2wk14:XXX:1>
  - csl2wk14 is the hostname of the sample computer, and XXX is the sample username. 1 indicates the number of your command, which will increase automatically.

Nachos start and exercise_1

- Step 2: see what directory you are in.
  - Type the command below:

    pwd
  - The directory will be shown. For example, “/homes/XXX”

Nachos start and exercise_1

- Step 3: download the nachos source code.
  - Type command below:

wget http://course.cs.ust.hk/comp3511/lab/lab02/os2015_nachos_exp0.tar

  - If you see "`os2015_nachos_exp0.tar' saved [283648/283648]" at the bottom line, then the source code is downloaded successfully.
  - Hint: If in the bottom line is "os2015_nachos_exp0.tar. 1" rather than "os2015_nachos_exp0.tar", this means the file "os2015_nachos_exp0.tar" already exists before downloading. And the file downloaded will be automatically renamed.

Nachos start and exercise_1

- Step 4: check whether the sources code is download successfully
  - ls
  - If you see the file “os2015_nachos_exp0.tar”, then the source code is downloaded successfully.

Nachos start and exercise_1

- Step 5: extract the source code
  - Type: tar xvf os2015_nachos_exp0.tar
  - The source code will be extract and a new folder “os2015_nachos_exp0” will be created automatically.
  - Hint: If you type “tar xvf os2” and then press the “Tab” key, the file name “os2015_nachos_exp0.tar” might be automatically shown. This feature is called “automatically filling”.

Nachos start and exercise_1

- Step 6: enter the source code directory
  - Type: cd os2015_nachos_exp0
  - Then type: pwd
  - Then you will see you are in directory “/home/xxxxx/os2015_nachos_exp0”.
  - Hint: the directory name is too long? Just type “cd os2” then press the “Tab” key.

Nachos start and exercise_1

- Step 7: compile the source code
  - Type: make
  - Many messages will scroll up. If no error reports, then it means success. Then the file “nachos” will be generated automatically.
- Step 8: check the target binary file
  - Type: ls
  - If you see the “nachos” file, then the compilation is successfully. The file name is displayed in green color, which means it is executable.
  - Hint: You could use “ls -l” command to see the detailed information of the files, such as file size.

Nachos start and exercise_1

- Step 9: run nachos
  - OK, this is the final step. Type: ./nachos
  - If you see the messages below, then congratulations to you.

OOPS: I'm called nachos ???
No threads ready or runnable, and no pending interrupts.
Assuming the program completed.
Machine halting!

Ticks: total 10, idle 0, system 10, user 0
Disk I/O: reads 0, writes 0
Console I/O: reads 0, writes 0
Paging: faults 0
Network I/O: packets received 0, sent 0

Cleaning up...

Nachos Introduction

- Nachos is an instructional software that allows students to study and modify an operating system.
- The only difference between Nachos and a *real* operating system is that Nachos runs as a single UNIX process, whereas real operating systems run on bare hardware.

Nachos Introduction

- Nachos simulates the general low-level facilities of typical hardware, including interrupts, virtual memory and interrupt-driven I/O.
- All hardware devices like the disk, console and the MIPS CPU are simulated in Nachos.

Nachos Introduction

- Nachos enables us to test out the concepts we learn about thread management, multiprogramming, virtual memory, file systems and networking.

- The Nachos code supplied to you is organized into these parts in the different sub-directories under ***nachos*** directory.

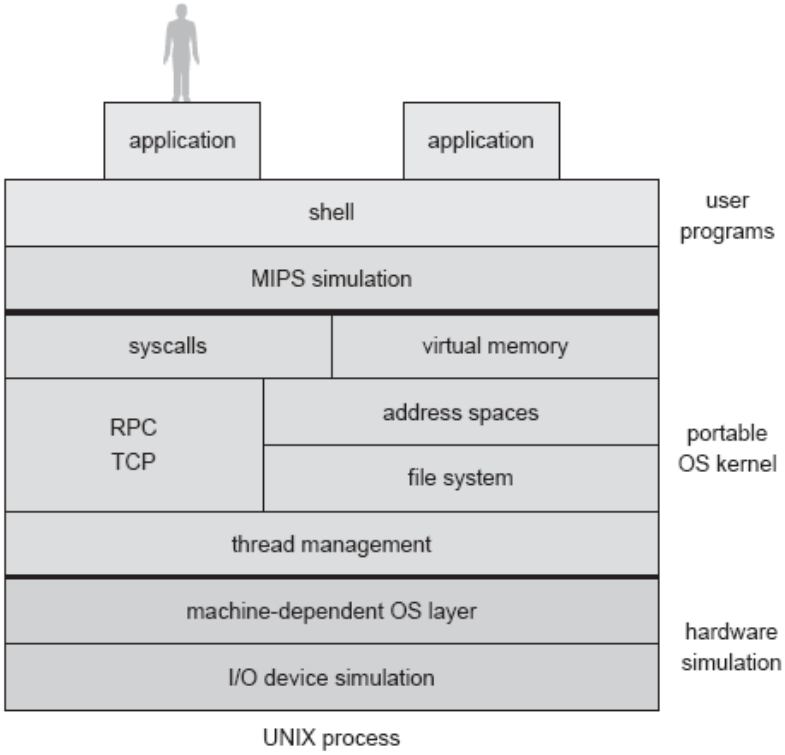
Nachos Introduction

- Nachos is written in C++ and is well-organized, making it easy for you to understand the operation of a typical operating system.

- If you have any difficulties with C++, please refer to the "C++ quick introduction" provided in our LAB1.

Nachos Introduction

- Nachos Software Structure
  - http://course.cs.ust.hk/comp3511/nachos_intro.html



Nachos Introduction

- Nachos runs on a simulation of real hardware.
- The hardware simulation is hidden from the rest of Nachos via a machine-dependent interface layer.
- For example, Nachos defines an abstract disk that accepts requests to read and write disk sectors and provides an interrupt handler to be called on request completion.

Nachos Introduction

- The Nachos kernel code executes in native mode as a normal (debuggable) UNIX process linked with the hardware simulator.
- The simulator surrounds the kernel code, making it appear as though it is running on real hardware.

Nachos Introduction

- Nachos simulates each instruction executed in user mode.
- Whenever the kernel gives up control to run application code, the simulator fetches each application instruction in turn, checks for page faults or other exceptions, and then simulates its execution.

Nachos Introduction

- When an application page fault or hardware interrupt occurs, the simulator passes control back to the kernel for processing, as the hardware would in a real system.

- Thus, in Nachos, user applications, the operating-system kernel, and the hardware simulator run together in a normal UNIX process.

Notes on Nachos

- You are strongly advised to read the various documentation available about Nachos on our course web
- You are also advised to read the source code of Nachos.

Notes on Nachos

- It is best if you can first read the header files (*.h) to understand the various objects and their member functions, before reading the implementation in the corresponding *.cc files.

- **READ THE COMMENTS IN THE SOURCE CODE CAREFULLY. THEY WILL HELP YOU IN UNDERSTANDING THE CODE**

Notes on Nachos

- Do not try and expect to understand every line of the source code.
- You should spend your time wisely, understanding the object structures and definitions of the member functions, and using them.
- The best method to understand the code is to compile and execute it.
- **Please learn Nachos materials and document on course web**

Nachos exercise_2

- Step 1: Download the Nachos source code for exercise

wget http://course.cs.ust.hk/comp3511/lab/lab02/os2015fall_nachos_sample.tar.gz

Nachos exercise_2

- Step 2: Extract the source code

  tar xvzf os2015fall_nachos_sample.tar.gz

- Step 3: Enter the source code directory

  cd os2015fall_nachos_sample

- Step 4: Compile: make

- Step 5: Run: ./nachos

- you should see the output:

```
Round robin scheduling
Susan arrives, waiting to read
Bill arrives, waiting to write
Mary arrives, waiting to read
Bill is writing the page of 0 in the database
Bill is writing the page of 1 in the database
Bill is writing the page of 2 in the database
Bill is writing the page of 3 in the database
Bill is writing the page of 4 in the database
Bill is writing the page of 5 in the database
Mary is reading the page of 0 in the database
Mary is reading the page of 1 in the database
Mary is reading the page of 2 in the database
Mary is reading the page of 3 in the database
Mary is reading the page of 4 in the database
Susan is reading the page of 0 in the database
Susan is reading the page of 1 in the database
Susan is reading the page of 2 in the database
Susan is reading the page of 3 in the database
Susan is reading the page of 4 in the database
Susan is reading the page of 5 in the database
No threads ready or runnable, and no pending interrupts.
Assuming the program completed.
Machine halting!

Ticks: total 130, idle 10, system 120, user 0
Disk I/O: reads 0, writes 0
Console I/O: reads 0, writes 0
Paging: faults 0
Network I/O: packets received 0, sent 0

Cleaning up...
```

Nachos exercise_2

- Then we will do some modification to Nachos
- Enter the *thread* directory: cd threads
- Open the *test.2.cc* by using Vi or other editor:

  vi test.2.cc

Nachos exercise_2

- [At Line 37] There are 3 people: Mary, Susan and Bill. It looks like this:

```
Thread *t_1 = new Thread("Mary");
Thread *t_2 = new Thread("Susan");
Thread *t_3 = new Thread("Bill");

t_1->Fork(read,  5);
t_2->Fork(read,  6);
t_3->Fork(write, 6);
```

Nachos exercise_2

- Delete Line 43 and Line 39 to “kill” Bill. Then it looks like:

```
Thread *t_1 = new Thread("Mary");
Thread *t_2 = new Thread("Susan");


t_1->Fork(read,  5);
t_2->Fork(read,  6);
```

- Save and exit

Nachos exercise_2

- Enter the root directory of the source code:

  cd ..

- Recompile: make

- Run: ./nachos

- You should see Bill disappear from the output

- you should see the output:

```
Round robin scheduling
Mary arrives, waiting to read
Mary is reading the page of 0 in the database
Mary is reading the page of 1 in the database
Mary is reading the page of 2 in the database
Mary is reading the page of 3 in the database
Mary is reading the page of 4 in the database
Susan arrives, waiting to read
Susan is reading the page of 0 in the database
Susan is reading the page of 1 in the database
Susan is reading the page of 2 in the database
Susan is reading the page of 3 in the database
Susan is reading the page of 4 in the database
Susan is reading the page of 5 in the database
No threads ready or runnable, and no pending interrupts.
Assuming the program completed.
Machine halting!

Ticks: total 90, idle 10, system 80, user 0
Disk I/O: reads 0, writes 0
Console I/O: reads 0, writes 0
Paging: faults 0
Network I/O: packets received 0, sent 0
Cleaning up...
```