

CMPT 300 Project #0: Nachos Survey

Objective

In this project you will read the Nachos source code and answer questions about it. You will also modify Nachos code to implement the programming assignment.

Getting Started

First, you have to install and compile Nachos. See the instructions on the course's webpage. After successful compilation of Nachos, run the executable to make sure it produces the expected results.

Reading Nachos Source Code

It is suggested that you read the files in the order described below. As you do so, read the corresponding sections in Thomas Narten's "A Road Map Through Nachos" and Archana Kalra's "Salsa – An Operating Systems Tutorial". The links to both of these documents are available from the course's webpage.

First, read the code for Nachos operating system. It is located in subdirectory threads. When you compiled Nachos the Makefile turned THREADS directive on (there are several pointers to makefile tutorials on the webpage for the course). Notice what portions of code in the files below were included due to this directive. Note also that other Nachos features like USER_PROGRAM, FILESYS, and NETWORK were turned off. Notice what command line arguments you can give to Nachos and what global data structures are created. Start reading the code by going through the following files

- threads/main.cc, threads/threadtest.cc – main function and simple test of thread routines;
- threads/system.h, threads/system.cc – Nachos startup/shutdown routines;

Read through the following files to see how Nachos implements and schedules threads. Study the Scheduler and Thread classes:

- threads/thread.h, threads/thread.cc – thread data structures and thread operations such as fork(), yield(), sleep(), and finish();
- threads/scheduler.h, threads/scheduler.cc – manages the list of threads that are ready to run;
- threads/switch.h, threads/switch.s – assembly language routines for creating, and deleting threads and context switch. Skim through these files. You are not expected to understand the details.
- threads/list.h, threads./list.cc – generic list management
- threads/utility.h, threads/utility.cc – some useful definitions and debugging routines.

After reviewing Nachos OS part, look at the Nachos simulated machine. The code for the machine is located in machine directory. Concentrate on the files listed below:

- `machine/machine.h`, `machine/machine.cc` – emulates the part of the machine that executes user programs: main memory, processor register, etc.;
- `machine/mipsim.cc` – emulates the integer instruction set of a MIPS R2/3000 CPU.
- `machine/interrupt.h`, `machine/interrupt.cc` – manage enabling and disabling interrupts as part of the machine emulation;
- `machine/timer.h`, `machine/timer.cc` – emulate a clock that periodically causes an interrupt to occur;
- `machine/stats.h` – collect execution statistics.

Debugging Nachos Source Code

There are three main ways to study and debug your Nachos program:

- **using Nachos DEBUG/ASSERT function:** the function itself is specified in `threads/utility.h`. The output produced by this routine is controlled by the command line options given to nachos at run-time. These options are described in `threads/main.cc` and `threads/system.cc`. The debugging option is “-d”; Run “nachos -d t” to see what your code is doing. You can add your own DEBUG calls and your own options
- **using printf:** you can use printf statements to output the values the variables assume at different times of program execution. **Caveat:** the output is not displayed immediately, it is. Use `fflush()` to force immediate printing;
- **using gdb:** gdb is a general purpose source level debugger. Using it is a comprehensive way of debugging your Nachos programs. Check the course’s webpage for reference materials on gdb. You should try using gdb before asking the TA or the instructor for help with your code.

Problems

These questions concern Nachos OS:

- 1) What module (file) and what function is function ThreadTest started from?
- 2) What does the following statement do?
`t->fork(procedure, parameter);`
What are the parameters to fork and how are they used?
- 3) What does the following statement do?
`DEBUG('t', "Entering SimpleTest");`
How are the two parameters for DEBUG used?
- 4) What does `yield()` do? What happens if `yield()` is executed and there is no other thread to run?
- 5) When does a thread execute `Finish()`? Can a thread execute `return()` instead? Can it execute `exit()` instead?
- 6) What does `scheduler::ReadyToRun()` do?
- 7) What does `scheduler::Run()` do and how is it different from `scheduler::ReadyToRun()`?
- 8) What data structure is defined in `list.h` and `list.cc` and how is it used in scheduler?
- 9) What module and what portion actually removes data structures of a finished thread and why the thread itself cannot do the cleanup?

10) What module keeps track of time used by nachos programs? Why cannot standard Unix time be used for this purpose?

Submitting Your Project

Write your answers in a text file answers_*****.txt (* as student ID)

Disk Space Usage

Nachos with compiled binaries may take over 4 Meg of disk space.

If you exceed your disk quota you will not be able to login to your account. Seek help of the system administrators if you have quota problems. You can check your disk quota usage with the following command:
prompt% quota -v