CS 4513 Distributed Computing Systems — WPI, B Term 2005
Craig E. Wills — Project 3 (15 pts)
Assigned: Thursday, December 8, 2005 — Due: Tuesday, December 13, 2005

Introduction

This project is intended for you to gain first-hand experience with clock synchronization across a set of distributed machines. Different methods are available to determine the current time as maintained by a machine and to synchronize this time with another clock.

Synchronization of Clocks on WPI Machines

In the first part of this project you will be observing the degree of synchronization between clocks on all WPI machines that you are able to access. The simplest way to obtain the current time on a Unix machine is to execute the command *date*, which prints out the current time with second granularity. The *gettimeofday()* system call can be used to obtain finer granularity, but second granularity will be good enough for this project.

There are different mechanisms for obtaining the current time of another machine. Different Internet time protocols have been defined. For example, the "daytime" protocol is defined in RFC-867 for a server process to listen on port 13 using either the TCP or UDP protocols and respond with a printable date string when contacted. A simple way to contact this server using the TCP protocol is with the *telnet* client, which can be used to connect to an arbitrary port. The following example shows how this service can be run from any machine to obtain the time on the machine ccc1.

```
% telnet ccc1 13
Trying 130.215.36.142...
Connected to ccc1.
Escape character is '^]'.
05 DEC 2005 17:02:30 EST
Connection closed by foreign host.
```

Unfortunately, most machines do not support this service and so attempts to connect to port 13 on many WPI machines (and machines elsewhere) will fail.

An alternate approach for obtaining the time on a remote machine is to use a remote execution mechanism such as *ssh* to run the *date* command on a remote machine (you will need to verify your password on the remote machine). For example:

```
% ssh ccc1 date
cew@ccc1's password:

Mon Dec  5 17:04:12 EST 2005
```

You can compare the time between your local machine and a given remote machine by "sandwiching" a local *date* command before and after obtaining the date from another machine. This sequence can be done using the shell command separator ';'. For example:

```
% date; telnet ccc1 13; date
Mon Dec  5 17:05:27 EST 2005
Trying 130.215.36.142...
Connected to ccc1.
Escape character is '^]'.
05 DEC 2005 17:05:27 EST
Connection closed by foreign host.
Mon Dec  5 17:05:27 EST 2005
```

This example shows the time on the machine ccc1 is synchronized at a second granularity with the local machine time.

In your write-up for this part of the project use these examples to obtain the *relative* time difference on all machines you can access on the WPI campus. Report the approach you use and the results for each pair of machines that you verify. Another WPI machine known to be running the daytime protocol is `davis.wpi.edu`

Synchronization of a Computer's Clock with Standard Time

Just because all of the computer clocks in a lab or at a site are synchronized does not mean they are synchronized with standard time. The time standard for the United States is maintained by the Time and Frequency Division of the National Institute of Standards and Technology (NIST). More information can be found at `http://tf.nist.gov/service/its.htm`. You are encouraged to explore this Web site.

For this portion of the project you will checking/setting the time on your local computer with the standard time. This page contains a link to a list of NIST Internet Time Servers, which can be queried using the daytime protocol on port 13. You will want to follow the links to software and instructions for the type of machine you are using for this portion of the project. Follow the FTP site link if you are on a non-Windows machine.

For a Windows machine, follow the instructions. For a non-Windows machine, there are a number of files at the FTP site. You will need to download the files `tcp_time_client.c` and `sw.c`. Once you have retrieved these files simply compile them to create an executable to run on your machine. The code compiles fine if using *cc*, but if using *gcc*, you will need to add an argument of "0" to three calls of the routine *exit()* in `tcp_time_client.c`. You will also get non-fatal warnings in compiling `sw.c` with *gcc*.

Run the executable and it will contact a NIST machine from a hardwired list to obtain the standard time and compare it with the local clock on your machine. The list in the code

is out-of-date and you can substitute it with a more current list. Alternately, you may need to use the "-u" option to try an alternate server in the list to avoid time-out problems.

If there is a difference in the local and returned time, you will be prompted to set the clock. Unless you are running as root on your machine, you will be not be able to set the clock. This client uses port 13 with a particular time format. You can download instructions for more details on its format and how it works.

In your write-up for this portion of the project indicate the machine on which you downloaded the client and the results in running the client. If the client machine was not the same as standard time what is the difference?

Synchronization of Clocks in the Internet

In the final portion of the project, you will examine the synchronization of clocks across the Internet. While you could try to contact the daytime service on different machines around the Internet, it is unlikely that you will know lots of machines to and even if you do that these machines will support the service.

However, a large network of servers that often supply a time is available through the Web. Generally when a Web server responds to a request it includes a header indicating its current time. Time synchronization between a Web server and client can be important, particularly if the server supplies an expiration time for downloaded content. You can use the "Date" header to compare the time at Web servers across the Internet.

For this portion of the project, you can use the command *wget*, which is a client that retrieves the content of a URL given on the command line. The "–server-response" option to the *wget* command causes the headers of the response to be displayed. In addition, the "–delete-after" option causes the saved content to be removed after retrieval so it does leave files in your directory. The following shows a sample invocation to contact the WPI Web server.

```
% wget --server-response --delete-after www.wpi.edu
--17:31:54--  http://www.wpi.edu/
           => 'index.html'
Resolving www.wpi.edu... done.
Connecting to www.wpi.edu[130.215.36.202]:80... connected.
HTTP request sent, awaiting response...
 1 HTTP/1.1 200 OK
 2 Date: Mon, 05 Dec 2005 22:31:54 GMT
 3 Server: Apache/1.3.27 (Unix) mod_pubcookie/3.0.1 mod_auth_pam/1.1.1
           PHP/4.3.11 mod_ssl/2.8.12 OpenSSL/0.9.7f
 4 Connection: close
 5 Content-Type: text/html

    [ <=>                                   ] 17,158         3.27M/s

17:31:54 (3.27 MB/s) - 'index.html' saved [17158]

Removing index.html.
```

The "Date" header gives the time on the Web server at the time the request is serviced. You could sandwich a call to this command with local invocation of the *date* command to compare the time difference between the local machine and the given Web server, however the *wget* command prints needed timing information of when the request is sent and the response received. You should think about why the time of the request and of the response is needed for comparing the accuracy of the clocks. Note that Web servers generally return the time for Greenwich Mean Time (GMT), which is five hours before Eastern Standard Time.

In your write-up for this portion of the project, turn in the results for each Web server you test. You must test at least 20 servers. You should test a mix of popular and unpopular servers that you know about. Is there a correlation between server popularity and accuracy of its clock? Why? As an aid in coming up with Web sites to test, you might consider contacting the Web servers at various colleges and universities. The URL `http://www.utexas.edu/world/univ/state/` has links to a large number of such sites in the United States. You might try both large and small sites on this list.

Submission

Create a text file and answer the questions posed in the handout. Turn in this file using *turnin* with project *proj3*.