ICT375 Advanced Web Programming

Lab 1 - Introduction and Apache Installation, Configuration and Administration

---

Learning Objective:

1. To be familiar with the Unix server `ceto.murdoch.edu.au` used for practical work in this unit.
2. Install, configure and run the Apache web server on a Unix platform.
3. To be familiar with some basic administration issues in Apache.

Access to Software:

The laboratory machines run Windows operating system. The tutorial exercises are to be conducted on `ceto.murdoch.edu.au`, which is running a Linux operating system. The laboratory machines can access ceto via Secure Shell (SSH) sessions. Ceto can also be accessed remotely using SSH, from Windows, Macs, or Linux operating systems.

Internal students (using the laboratory machines) have access to the PuTTY program on Windows for SSH sessions, and WinSCP to securely copy files to and from `ceto.murdoch.edu.au`.

External students can download PuTTY from http://www.putty.org/ and WinSCP from http://winscp.net/eng/index.php, as well as any other required software and install the relevant application software on their own machine. Internal students may do the same at any off campus locations, but they are required to attend the tutorials to complete the exercises below.

The software named Totalvalidator can be downloaded from website http://totalvalidator.com. This software should be installed on your local machine and used to validate your HTML code to XHTML5.2 standard.

Instructions:

For all instructions below, replace the number 12345678 with your own 8-digit student number, and the number 15678 with the port number supplied to you by your tutor.

In the examples, the **`words in teletype bold like this`** are commands you type in the SSH sessions. The `words in teletype like this` (i.e. not bold) are what you will see in response to a command, but do not need to be typed in.

Logging on to `ceto.murdoch.edu.au`

Using an SSH session, connect to the address `ceto.murdoch.edu.au`. When prompted for a username and password, use your normal Murdoch student login credentials (Student ID and Password). You should have received information in regard to your Murdoch login from IT Services in the University.

For those using a Linux machine, you can use your Student ID in the initial ssh command:

`command prompt:~>` **`ssh 12345678@ceto.murdoch.edu.au`**

In this case you will be prompted for your password only. NOTE: you should never invoke an ssh session as the superuser/administrator. Doing so could result in a major security breach.

Windows users can fill in the appropriate details in the GUI provided by PuTTY.

Part A (Apache Installation):

Installing the web server

You will compile and build a version of the Apache httpd web server distribution in your own home directory on `ceto`. There is a copy of the distribution in the directory `/student/share/ICT375`. Copy it into your home directory, unzip it and unpack the tar file, then configure the `makefiles` and `make` the installation. To do that, type in the following set of commands on your ssh session command line when logged into `ceto.murdoch.edu.au`, in the order specified below.

Before you start the installation, you need to find out your *home directory* on `ceto`. Type the command `cd` followed by the command `pwd` to find out your home directory. In the following examples for student 12345678, the home directory is assumed to be:

`/home/student/accounts/12345678`

[*Note that all commands below, between one "`ceto:~>`" prompt to the next, are supposed to be typed* ***one per line****. Beware of this because your terminal may show the commands in two lines if there isn't enough space to fit it in one line in your window.*

Eg: the "./configure --prefix=/home/student/accounts/12345678/apache" command below may look like it is on two lines, but please ensure the "--" and "prefix=/home..." are joined (without any space or break) in a single line.

*Note that the 2nd command "***`cp /student/share/ICT375/lab01/httpd-2.2.23.tar.bz2 ./`***" has a period and forward slash ("./") at the end.*

Obviously, you should replace 12345678 *in any command below with your own student number.***]**

```
ceto:~> cd
ceto:~> cp /student/share/ICT375/lab01/httpd-2.2.23.tar.bz2 ./
ceto:~> bunzip2 httpd-2.2.23.tar.bz2
ceto:~> tar -xvf httpd-2.2.23.tar
ceto:~> cd httpd-2.2.23
ceto:~/httpd-2.2.23> ./configure --prefix=/home/student/accounts/12345678/apache
Take note of the output and then type...
ceto:~/httpd-2.2.23> make
Take note of the output and then type...
ceto:~/httpd-2.2.23> make install
```

The purpose of the putting your own directory in the "`./configure ...`" line is to ensure that the final server you build will operate from your home directory only, rather than the default `/usr/local/apache`. The last output you should see after the above sequence of commands is:

```
...
Installing man pages and online manual
mkdir /home/student/accounts/12345678/apache/man
mkdir /home/student/accounts/12345678/apache/man/man1
mkdir /home/student/accounts/12345678/apache/man/man8
mkdir /home/student/accounts/12345678/apache/manual
Installing build system files
make[1]: Leaving directory '/home/student/accounts/12345678/httpd-2.2.23'
```

To save space, please delete the Apache distribution files in your directory as follows:

```
ceto:~> rm -r ~/httpd-2.2.23*
ceto:~> ls
apache
ceto:~>
```

Too much disk space will be wasted if all students keep copies of the same files individually. If you encounter problems in the future and needs to re-install, you can repeat the steps above.

Configuring the web server

You should now have a web server ready to run. You now need to change the `httpd.conf` file so that your server is configured appropriately for this unit (just follow the instructions below for now). To make the changes, you will need to use a text editor[1]. You may use the more powerful (but less intuitive) `vim` or `emacs` if you are familiar with those editors, or simpler ones like pico or nano[2]. Open the `httpd.conf` file by typing the following commands on ceto:

```
cd ~/apache/conf
pico httpd.conf
```

The three major categories of changes that needs to be made here are:

- Change the port number from the standard 80 to one unique to you. Check with your tutor for the port number allocated to you. Please only use your allocated number for all of your work on `ceto.murdoch.edu.au` or we will have mass confusion among all of you when doing your work. [Note: If you enrolled late and you have not been allocated a port number, please contact the tutor.]
- Change the process control directives so that it starts up a lot less processes than a normal web server. This is to save memory space and reduce processor loading on the server machine. We need to rationalize processor and memory usage.
- Change the user account that the web server runs as, so that it can access your files in your account.

To make the configuration changes in `httpd.conf`, look at the contents of the file in your text editor. Most of them will be comments starting with '#', but a few of the lines without the '#' will be the actual directives. You need to change the following directives to the values below. Remember to replace 12345678 with your own student number, and 15678 with the port number allocated to you. For the `ServerAdmin` directive, put down your email address. Lastly, remove the '#' comment mark at the beginning of the ServerName directive. So you should have the following directives in your `httpd.conf` (Note: these directives appear in the order below in the `httpd.conf` file, however they will have comment lines between them):

ServerRoot "/home/student/accounts/12345678/apache"
Listen 15678
User 12345678
Group *# leave this as it is*
ServerAdmin <your email address>
ServerName ceto.murdoch.edu.au:15678
DocumentRoot "/home/student/accounts/12345678/apache/htdocs"

For the User directive, replace 12345678 with your `ceto` account name. This directive will make your web server run as the same user and group as your account on `ceto`. This is to ensure you will not have permission problems operating the server; you are ensuring that the server accesses the web-site only in directories you own. Note that this is only necessary for the sake of this unit, and NOT something you usually do when configuring a working server.

Now we want to limit the number of spare child servers that your web server would keep. Firstly you need to uncomment (i.e. remove the '#') the following line in `httpd.conf` file (this line is located near the end of the file):

1 As ceto is a Linux server, and you are logged in via an ssh session, there is no GUI availability for any applications so you will need to use command line editors in your ssh session window.

2 You will need to learn how to use these editors in your own time. Your tutor has information on basic commands for vim.

```
#Include conf/extra/httpd-mpm.conf
```

Then edit the file `httpd-mpm.conf` in the sub-directory `extra` using the following figures:

```
<IfModule mpm_prefork_module>
        StartServers            3
        MinSpareServers         1
        MaxSpareServers         3
        MaxClients              10
        MaxRequestsPerChild     0
</IfModule>
```

Running the web server

To run the web server, you use the `apachectl` script in the `bin` directory of your installation. To start the server, type the command (Note: `apachectl` stands for apache control):

```
ceto:~> ~/apache/bin/apachectl start
```

Accessing the web server and changing the web site

To use your running web server, start up any web browser, and access the URL `http://ceto.murdoch.edu.au:15678/` - again replace 15678 with the port number allocated to you. You will see a page with the starting line "**It works!**" You can find this HTML page `index.html` in the `htdocs` directory of your installation.

To change the contents of the web site served by your web server, just make changes to `index.html` file and/or to the `htdocs` directory in your apache directory. Eg: if you put a file `mypage.html` in the `htdocs`, then the URL is `http://ceto.murdoch.edu.au:15678/mypage.html`. You may also put sub-directories in the `htdocs` directory.

You can change file and directory contents of the `htdocs` directory by using WinSCP (or sFTP or scp) to access `ceto`, or by using a text editor to explicitly create files on the command line on ceto.

Stopping and restarting the web server

To stop the server, use the `apachectl` script again:

```
ceto:~> ~/apache/bin/apachectl stop
```

FROM NOW ON, YOU SHOULD STOP THE SERVER AT THE END OF EVERY LAB SESSION (WHETHER ON CAMPUS OR NOT). If you do not, we will end up with a very large number of servers running; these could take up a lot of memory space for nothing. We could face the possibility of `ceto` being unusable and requiring clean-up by the system administrator. This will prove *very inconvenient around assignment time*.

If necessary, you may make changes to configuration file `httpd.conf`. The server reads the configuration files when it starts. So remember that none of the changes you make will take effect until you restart the server. To restart a running apache server you can issue the command:

```
ceto:~> ~/apache/bin/apachectl restart
```

Manuals

The standard Apache online manuals are found in the directory `manual/` in your installation directory. To make this directory accessible from your web browser, you must first change your standard configuration. Go to your `httpd.conf` file, search for the line below and uncomment it to make the `manual` directory available:

Include conf/extra/httpd-manual.conf

After restarting your server, you will be able to access the manual pages using a web browser by accessing the `URL http://ceto.murdoch.edu.au:15678/manual/.` These are the same web pages as available on the official online web site at `http://httpd.apache.org/docs/2.3/`

Hints for Unix shell novices

As ICT286 is a prerequisite unit for this unit, you should be familiar with most of the Unix/Linux commands required for tutorial and assignment work. Here are a few useful hints about Unix and the default shell you are using on `ceto`:

- Use the up and down arrow keys to go through the list of previous commands.
- Press the tab key to automatically complete a partially typed command name or file name. The shell will complete the commands based on your path, and files based on what is in your current directory. This makes typing long command and file names easy. For example, in the command "**`bunzip2 httpd-2.2.23.tar.bz2`**" you can just type "**`bunzip2 http`**" and then press tab.
- A few useful symbols used in filenames:
  - **~** means your home directory (so `~/.cshrc` means the .cshrc file in your home directory)
  - `~12345678` means the home directory for user 12345678
  - **./** designates the current working directory (so `./configure` means execute the configure script in the current working directory)
  - **../** designates the directory above the current directory (i.e. the parent directory)
- If necessary, refresh your memory about Unix/Linux shell commands by briefly reviewing the 'Intsructions' section in the tutorial worksheets for Labs 1-7 from ICT286.

Before proceeding with Part B:

Complete the above instructions; make sure the changes you have made to `httpd.conf` are correct. Then follow the instructions above to stop your web server, start it again; stop and restart again.

Note:

The purpose of eac student install and configure your own apache installation is so that you can try different things. DO NOT BE AFRAID TO EXPERIMENT WITH DIFFERENT SET-UPS AND CONFIGURATION, or to move and change files. Make sure you backup previous work before you make these changes. In the worse-case, you can delete your whole `apache` directory and do the installation again - it takes less than 5 minutes once you are familiar with the steps.

The only thing you should be wary of is starting up too many server processes. If you think about what you're doing, read the manuals on the configurations you are changing, or consult your tutor, then it should not cause too many problems. Even if you mess up, then learn and move on. These lab exercises are not worth much if all you do is follow instructions as stated above but do nothing else.

Part B (Apache Configuration and Administration):

Monitoring requests to your web server

You can look at what requests your web server has received by checking the `access_log` file in the `logs` directory of your installed your web server. You can use any text editor (e.g. pico or vim) or viewer (such as `more` or `less`) to view this file:

```
ceto:~> less access_log
```

To quit the `less` program, press "q". Use a web browser to view your server's main page again (as given in **Accessing the web server** section from Part A exercises). Now view the `access_log` file again, and you will see new lines at the bottom corresponding to your new access.

Look up the manual on the entries to an `access_log` file, and see how you can configure it to display data you are interested in. [Optional: You will also notice access to this file called `favicon.ico`. If you have not previously learnt about this web-site file, do your own search on the web to learn about it now.]

To keep a continuous display of new request entries, instead of having to open and close a text viewer, you can use the command:

```
ceto:~> tail -f access_log
```

Keep the SSH window with this command visible. Use your browser to access the server again, and you will see that the lines will appear on the SSH window screen on each new access.

Another commonly used log file is the `error_log` file, located in the same `logs` directory as `access_log`. These logs are errors the server encounters. Use the same methods as given for `access_log` above to view this error file. This is potentially a more important file when you are testing your configurations. Keep this file in mind.

Looking at the server's resource usage

While the server is running, you will be able to see the status of processes doing the job and how many resources they are using. Two ways of doing so are by the `ps` command and the `top` command.

For example, to look at the processes involved in the web server you started up, you can type the command (for userID 33701):

```
ceto:~> ps -lA | grep 33701
```

You will see something like the following:

```
F S   UID   PID  PPID  C PRI NI     ADDR     SZ    WCHAN TTY      TIME CMD
 8 S 33701 14703     1  0  41 20        ?    305        ? ?        0:00 httpd
 8 S 33701 14704 14703  0  41 20        ?    338        ? ?        0:00 httpd
 8 S 33701 14705 14703  0  41 20        ?    305        ? ?        0:00 httpd
 8 S 33701 14706 14703  0  47 20        ?    305        ? ?        0:00 httpd
 8 S 33701 10193 10188  0  51 20        ?    353        ? pts/5    0:01 tcsh
```

The `httpd` with PID (process ID) of 14703 is the parent process, while the other 3 are the child processes waiting to serve requests. If the server has been running for a while, these child process IDs will change as they get killed off and new ones get started (see Topic 1 on Apache's process model). Some useful fields to know for the command display:

`UID` : user identification - every user has a number which is different from the username. Your username is of the form 12345678, but the UID (in the example above) is of the form 33xxx. You can find out your own UID by typing the "**`id`**" command on the command line.

PID : process identification - unique number for every process.

PPID : parent process ID - the parent process identification. In the example above, you can see that all child `httpd` processes are spawned by the parent process 14703.

CMD : command executed by the user.

To look at just **apache** web servers running on the machine, use the command:

```
ceto:~> ps -lA | grep httpd
```

A better (but much more processor intensive) way of looking at the processes is by using the `top` command:

```
ceto:~> top
```

The top few lines of the results are the overall and available system resources and the summary usage by all processes. Below that is a display of the processes sorted by highest resource usage in real time. By default, it refreshes every 5 seconds. If you access your web server to get a HTML file, you will see your own process pop up to near the top of the screen for a few seconds before going back to idle.

To quit the display from the `top` command, type “q”.

A good way to monitor your web server resource usage is by typing “u” after starting `top` and then type in your user name (eg: 12345678). Then the display will be restricted to your processes only. To illustrate what the display can show you, bring up a web browser window but keep the top monitor on the screen at the same time. Now access a web page on your server (preferably one which you haven't accessed before) and you will see the CPU and memory usage of one of the child **httpd** processes increase.

Some useful fields to know for the `top` monitor screen (besides fields already listed for `ps` above):

USER : username - yours is of the form 12345678.

SIZE : total memory used in kb - physical and virtual memory.

%CPU : estimated average percentage of total CPU used since the last screen update.

%MEM : estimated average percentage of total physical memory used since the last screen update.

TIME : total amount of CPU time used in hour:minutes format for idle httpd processes, this will stay 0:00 until it gets requests.

USE THE `top` COMMAND SPARINGLY SINCE IT CHEWS UP A LOT OF MEMORY AND CPU TIME! If you start `top` and you see many others already have theirs started up, it may be a good idea to stop yours and try again later.

There are a lot more options to `ps` and `top`. Look up the manual entries for them by typing:

```
ceto:~> man ps
ceto:~> man top
```

Exercises:

Complete the above instructions (i.e. in configuring your Apache server) and then the following exercises.

1. Start putting your own files into your installation's `htdocs` directory. Add sub-directories and put files in them as well. Access the files via a browser to make sure your server is doing its job properly. Understand the conversion rules your web server is using to convert between the URL string and the file system's pathnames.
   [You may use your project web-site from ICT286 in `htdocs` provided you have a backup copy of it!! Note: some server-side scripts such as Perl/CGI or PHP will not work unless you configure your server to handle them.]

2. Explain a standard entry in your access_log file. Refer to the online manuals (especially the tutorials section) for more details of logging and the format of the log files.

3. Using the Unix process monitoring mechanisms described above, look at the changes in system resource usage for your server, as well as other servers. Change the process control directives in your `httpd.conf` file, and see how they affect the processes.

4. Following the instruction given in Lecture 1 (b) and referring to the Apache website, configure your web server to do the following:
   a) Turn on `extended mod_status` and view your web server usage and status using a web browser.
   b) Turn on server-side includes (SSI). Create a simple SSI file and see if your server can process and serve it properly.
   c) Restrict access to a directory on your web site. Users should only be able to access the directory if they supply a username and password.

Assessable Tasks:

You should attempt to complete all of the above exercises. However, you only need to demonstrate / submit for assessment Exercises 1, 4a and 4b above. NOTE: even though not all exercises are being assessed, you are encouraged to attempt to complete all of the above exercises in the order given. They are presented to help you develop your understanding of the concepts, and work with, the technology. This will be of benefit toward your work on future lab exercises and the assignments.

Internal students must demonstrate to your tutor a running server and your capability to access different files from your web-site using a browser, in addition to Exercises 4a and 4b above. This can be done during the next lab session, but no later than Week 3, Monday 11/03/2019.

External students must submit their work with their first assignment submission. You can demonstrate the above (a running server, accessing your web-site using a browser, and performing the required tasks for Exercises 4a and 4b above) by providing edited configuration files and screen captures. These should be placed in appropriate sub-directories and included in the submitted assignment zip file. Assignment submission is Week 7, Monday 8/04/2019.

Final Reminder:

Remember to stop your servers on ceto BEFORE you terminate your ssh session. This should be done by typing `apachectl stop` on the command line.

Also remember to close your ssh session BEFORE logging off your lab or home computer!