B336 Advanced Internet Computing

Introduction to Web Serving Using Apache

Learning Objectives

- Understand the reasoning behind the choice of using Apache as part of this unit.

- Understand the basic configuration issues of Apache.

Why Use Apache in this Unit?

- We need an example web server for you to play with to learn about web service issues.
- It needs a complete set of features so you can see the full power of a web server before starting to implement a simple one yourself (next week)
- Apache is the most popular web server on the web today
    - Estimated 60% of sites on the web uses Apache
    - You will improve your CV considerably having direct exposure to it.
- It’s free, and it’s Open Source.

Some Examples of Other Browsers

- Some other major web browsers:
  - Miscrosoft's IIS
  - Netscape's Enterprise and iPlanet
  - NCSA HTTPd

- http://serverwatch.internet.com/webservers.html

Apache Installation

- We will go through it in next week's labs, using a linux server "`red.murdoch.edu.au`".
    - More details about what to do in the labs exercises.

- You will install and run the server from your own directories, but you will all use different port numbers, and they will not be the standard port 80 for HTTP access.

Port Numbers

- So the situation we will have is that if someone uses the URL

  `http://red.murdoch.edu.au/`

  Then they will get to the perpetually running, superuser-installed web server on red.

- This is because the client defaults to if the port number is not typed in by the user. The actual URL for the above is:

  `http://red.murdoch.edu.au:80/`

Port Numbers

- To get to your web server, which you will run only when you need to, the URL will be

  `http://red.murdoch.edu.au:12345/`

  Or whatever port number you are assigned.

- This way, we can have multiple web servers running on one machine at the same time.
  - Later you will learn about setting up *Virtual Hosts*, which is an even more sophisticated way of serving multiple domain names and addresses using one machine.

Forks and Threads

- Two important concepts in web serving are:
  - **forking**: when one process (called the parent) creates a complete copy of itself to execute certain things
  - **threads**: parts of a single process which can execute independently without worrying about each other.
- A web server must serve more than one request at a time, so there needs to be mechanisms like forking and threading in programming languages, and in the operating system to allow them to do this.

Apache‘s *Pre-Forking* Model

- Apache uses a *pre-forking* model
  - A running Apache always has a *parent* process running.
  - The parent process is responsible only for forking *child* processes, it does not serve any requests itself.
  - The child processes process connections, before dying - hopefully gracefully, if nothing goes wrong.
  - The parent spawns new or kills off old children in response to changes in the load on the server.

Forks and Threads

– We will not be going into the programming of forking and threading in this unit.
  - The unit **B310 UNIX and Network Programming** will cover them very extensively.

– For this unit, we will study more on implementing what happens to the request/response communication, rather than how to do multiple ones at a time.

– But do keep the basic ideas in mind, as you will need to deal with administration issues for a web server like Apache.

Starting-Up Apache

- The Apache program is called `httpd` (HTTP daemon).
- Starting and stopping the Apache process is done by typing the command "`httpd`" at the command-line.
- The Apache distribution also comes with a script called "`apachectl`" which you can use to control apache.
    - Egs. apachectl start
      apachectl stop
      apachectl restart

The Apache Start-up Process

- When the Apache server starts up, it will determine what the appropriate configuration (eg. what port number to listen to) is by reading some configuration text files
    - by default in the `conf` directory your installed it in.
- The files are read in the following order:
  `httpd.conf, srm.conf, access.conf, magic, mime.types`

The Apache Configuration Files

- These days, most directives for resource (srm.conf) and access (access.conf) control are put in the main httpd.conf file as well
    - srm.conf and access.conf exist only for historical reasons
- The `magic` and `mime.types` files you usually leave as they are.

httpd.conf

- This is the file to tell the server how to run - ie. it determines the configuration.
- It is read by httpd process on start-up
  - any changes made to this file won't take effect until the server is restarted.
- Lines starting with "#" are comments and ignored when the server reads the file
  - Comments in the httpd.conf file contain very detailed description of the directives that comes with the standard installation. You should read them to get more details.

Example httpd.conf

```
ServerType standalone
ServerRoot "/usr/local/apache”
Port 80
ServerName www.it.murdoch.edu.au

<Directory />
    Options FollowSymLinks
    AllowOverride None
</Directory>

<Directory "/export/home/staff/hiew/public_html">
    Options +ExecCGI +Includes +Indexes
    AllowOverride All
</Directory>

...
```

httpd.conf

- This is the file to tell the server how to run - ie. it determines the configuration.
- Lines starting with “#” are comments and ignored when the server reads the file
  - Comments in the httpd.conf file contain very detailed description of the directives that comes with the standard installation. You should read them to get more details.
- Besides the comments, all other lines are *directives* (ie. instructions for the server to follow)
  - Each line (except blank lines) contains one directive

`ServerType standalone`

- ServerType is either inetd, or standalone.

- The inetd mode is only supported on Unix platforms. It is the operating system's Internet daemon process which collates and distributes all requests to ports it knows about
  - You need to be the superuser to register with inetd.
  - Also, you lose some control over configuring the resource management for your server (eg. how many child processes to have) if you get inetd to be your initial start-up point.

```
ServerRoot "/usr/local/apache"
```

- The top of the directory tree under which the server's files are kept.
  - Configuration files: `$ServerRoot/conf`
  - Log files: `$ServerRoot/logs`

```
Port 80
```

- The port to which the standalone server listens

```
User nobody
Group nobody
```

- The user and group that the server process runs as.
- This determines what files it can permissions to access - when you want the web server to serve files in a certain directory, the user `nobody` must have access to that directory as well.

```
StartServers 5
MaxClients 150
MinSpareServers 5
MaxSpareServers 10
MaxRequestsPerChild 30
```

- Some settings for the child processes serving requests.

```
DocumentRoot "/usr/local/apache/htdocs"
```

- The directory where your web-site files are

```
ServerName www.it.murdoch.edu.au
```

- name which is sent back to clients if it's different than the one the program would get.

Scope of Directives

- The examples above are all *server-wide* directives.
    - They apply to the server itself.

- Some directives are only relevant to certain parts of the server. These directives are placed in *context* tags.
    - Eg. httpd.conf on page 15, the second <Directory> tag encloses directives that apply ONLY to my public_html directory (and all sub-directories in it).

Configuration for your installation

- In next week's labs, you will be given a set of standard configurations for your own installation on `red.murdoch.edu.au`.
- You can then try out different configurations of your own.
- Keep in mind that they will be different from the standard examples above, since it is not normal we have so many different installations running at the same time, and all of which do not have superuser/administrator priviledges.

Reference Documentation

- The lectures and labs will only be able to give you a basic introduction to the web serving features of Apache.
- To operate it properly, you will frequently have to refer to the official online documentationn - which includes quite a lot of tutorials.
- You will be shown the location of the documentation on your own installation in the labs. You can also get it from the Apache site `http://httpd.apache.org/docs/`