CIA Practicum Assignments
Webservers

E.P. Schatborn N.P.H. Sijm A. van Inge C. Dumitru

October 9, 2012

Abstract

Webservers are an important way of putting information out on the Internet or on an Intranet. A well known webserver is the Apache HTTP server. We will compile and install an Apache webserver under Ubuntu on your experimentation machine. The goal is to learn the various configuration options of Apache and to understand the concepts.

1 Apache

The Apache webserver originated from the public domain webserver of the NCSA[1]. After the development of the NCSA webserver stopped in 1994, system administrators started creating their own patches for it. A group of system administrators decided to create a webserver based on the NCSA webserver with the most useful patches included.

Since then the Apache webserver has been completely rewritten, and from version 2 onwards the old NCSA code has been fully replaced and is no longer part of the source. One year after its introduction the apache webserver was already the most popular webserver and it stayed that way for a long time, though since around 2006 there seems to be a shift[2].

→ Can you think of reasons for this change? Explain.

2 Installing Apache

The most recent release of the Apache webserver is version 2.4.3. You can download it from `http://httpd.apache.org/`.

→ Older source trees like 2.2.* and 2.0.* are still maintained. Can you think of reasons why?

[1] The National Center for Supercomputer Applications.

[2] `http://news.netcraft.com/archives/web_server_survey.html`

To compile Apache you will be using the standard sequence of `./configure` , `make` and `make install` . Be sure to set the correct options for `configure` when you start compilation. (Hint: read the requirements section carefully . . .)

There are many modules that can be used with Apache. They can either be compiled into the binary, or they can be compiled as Dynamic Shared Objects (DSO) which can be loaded at runtime. Make sure that `mod_ssl` is enabled.

Be sure to first remove all other installations since they might interfere with yours, then compile and install Apache in `/opt/local/apache`. Make sure that the webserver starts at boot, using the method provided by Apache.

Apache uses `httpd.conf` as a configuration file. It contains a lot of comments. Read them all carefully and look up things that are not clear to you. Then configure the basic settings like the webmaster email address, the webserver name, the root directory for the web documents and the port the webserver listens on. You can check the syntax of the configuration file using `httpd -S`.

3 Virtual hosts

A much used functionality that Apache provides is the hosting of multiple domains from a *single* webserver. This "virtual host" functionality resembles the MTA setup we worked with last week. The virtual domains exist only as resource records on the DNS server and as data on the webserver.

→ Implement virtual hosting in the webserver for the virtual domains you created in the assignment about MTA's.

→ Create a simple html page for each virtual host make sure that apache can correctly serve it.

→ Use `curl` to display the contents of a full HTTP/1.1 session served by your server. Explain the meaning of each header.

4 Encryption

It may be necessary to encrypt an http session so that others can't listen in. It is essential for online banking or financial transactions. This is what the https protocol is used for. It is a connection over port 443 using SSL/TLS encryption.

→ Configure your webserver to support SSL/TLS.

- It is not difficult to set up your server to support SSL/TLS. It *is* however difficult to do it *correctly*. Be sure you know what you are doing and read the Apache documentation on SSL carefully[3].
- Read the SSL configuration file carefully before you start. Adapt it to your needs and make sure the Apache SSL module is loaded.

→ What encryption standards does the webserver support using the standard configuration file?

→ Describe how you created your own certificate for your webserver.

→ You can test your secure webserver using a web browser, but you can also use `openssl` or `curl`. Test your webserver using these tools.

→ Can you enable SSL for all your virtual hosts? Explain.

5 Webserver security

The security of a webserver is largely derived from the access rights given to documents on the server. Apache has a reasonably good reputation in the area of security. It is mostly the code that users[4] add to the webserver that creates security problems.

→ Investigate what configuration options there are that control user access right.

  → What ways are there to use these options on documents?

→ Now create two webpages, one with a simple SSI instruction and one with a simple Perl/Python/Ruby CGI script

  → Set up your webserver so that only code on these pages can be executed.

6 Bonus

6.1 Webserver performance

Webserver performance is a very important issue as this directly impacts user experience and can affect operation costs.

[3] Apache HOWTO quote:

> [...] but always try to understand the stuff before you use it. Nothing is worse than using a security solution without knowing its restrictions and coherences.

[4] The users are the people using your webserver to make their content available on the Internet.

→ Investigate what configuration options there are that can potentially improve the performance of the webserver. Also look at how you can check the (current) load on the webserver using the `mod_status` module.

→ Using a standard benchmarking tool (e.g. ab, siege, etc.) evaluate the performance of your server before and after optimizations for both the static page and the dynamic page. Try to maximize the number of requests per second. Explain all the changes made.

6.2 Logging

Checking the logging information on your webserver is important to discover problems and/or attacks on your webserver. The Apache webserver has different configuration options to adapt the log information to your needs. This allows you to extract useful information from the logfile.

→ Define your own log format containing information you deem important. Use *Conditional Logging* to add User-agent and referrer information to request logs that generated an error.