Práctica 4: Instalación y Gestión de Servicios en Sistemas Distribuidos.

Programación y Administración de Sistemas

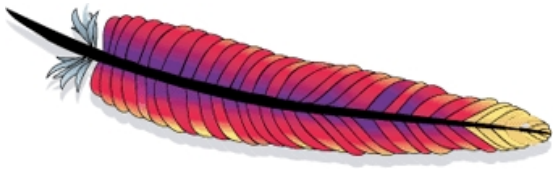
Segundo curso de Grado en Ingeniería Informática

Javier Sánchez Monedero

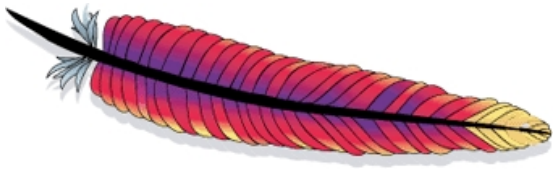
Dept. de Informática y Análisis Numérico

9 de mayo de 2012

Índice



Índice



Objetivos

Objetivo principal

Conocer a grandes rasgos cómo funciona la WWW a través del servidor web Apache.

Objetivos detallados

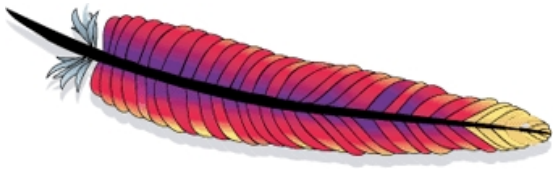
- Conocer los protocolos y lenguajes que hacen posible la navegación web.
- Conocer las partes del paquete software del servidor Apache: programas, ficheros de configuración, documentación, etc.
- Manejo básico de la configuración de Apache.

Entrega

Entrega y defensa de la práctica

Durante la práctica guarda en un fichero de texto las órdenes usadas y las directivas de configuración empleadas indicando para qué sirven. La entrega de la práctica será esta *chuleta* de directivas de Apache que podrás usar en la defensa para hacer algún cambio de la configuración de Apache oportuno. En el propio fichero de ejemplo que viene con Apache tienes comentarios sobre las directivas en inglés. Puedes encontrar más información en la documentación en castellano.

Índice

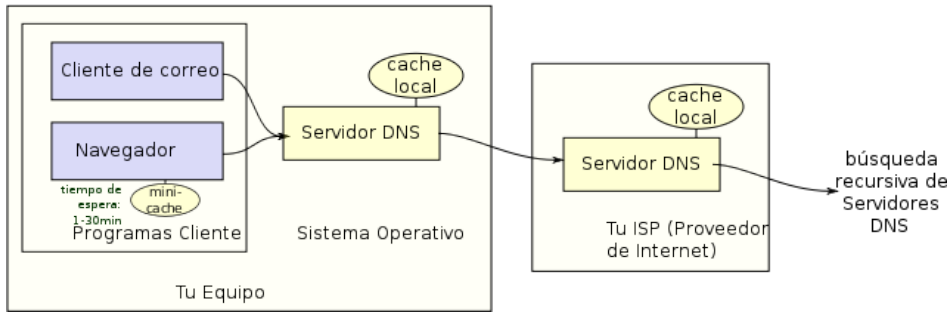


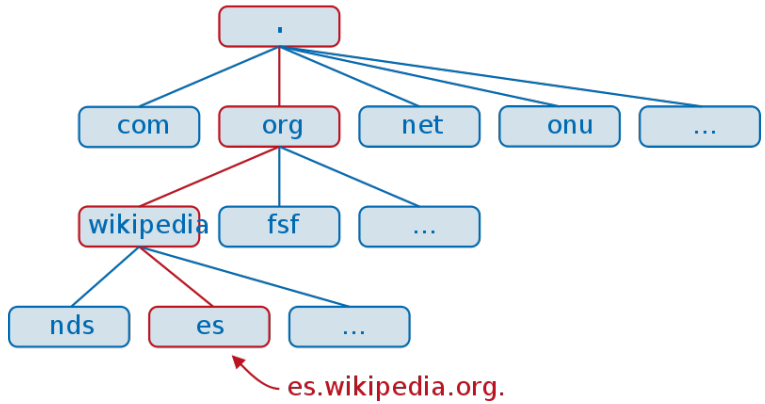
Internet con un enfoque descendente

- ¿Qué pasa cuando le das a intro en el navegador?
  - URI (identificador uniforme de recurso): "`http://es.wikipedia.org`"
- ¿Qué es un servidor? cliente↔servidor = petición↔respuesta
- Varios servidores intervienen cuando navegáis (Router, DNS, Web, etc.)

¿Dónde están las cosas?

- Para averiguarlo, una serie de servidores se pasan mensajes de acuerdo a un protocolo
- DNS (sistema de nombre de dominio)
- El DNS devuelve la dirección IP asociada a un dominio
- Muchos dominios pueden estar asociados a una misma IP, así servidores web como Apache permiten alojar varios dominios simultáneamente (*Virtual Hosts*).
- También se pueden asociar múltiples IPs a un dominio para hacer balanceo de carga, por ejemplo.





Metáforas de algunos conceptos

- Protocolo: comedor universitario
- Dirección IP: códigos postales + calle + número + piso. Ejemplo: 150.214.110.212
- Dominio: nombre único y entendible por humanos para no memorizar direcciones IP (entre otras cosas). Ejemplo: uco.es

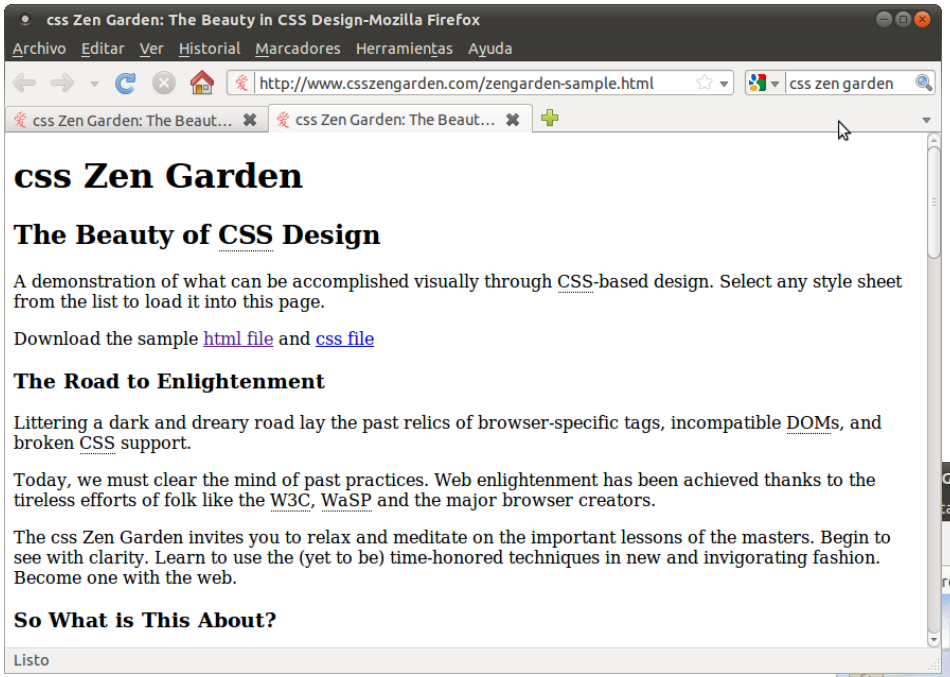
¿Quién muestra las Webs?

- El sistema operativo ya sabe dónde está el servidor web (dirección IP que le pasó el DNS).
- El navegador usa el protocolo HTTP para *pedirle* un recurso (por ejemplo un fichero HTML) al servidor Web, que analiza, y posteriormente hace peticiones adicionales para los gráficos y otros ficheros que formen parte de la página.
- El navegador *renderiza* (muestra) los datos recibidos del servidor tal y como describen el código HTML, CSS y otros lenguajes, y se incorporan las imágenes y otros recursos.

¿Cómo se programa una Web?

- El contenido se escribe en HTML o en XHTML (lenguaje extensible de marcado de hipertexto)
- La forma se describe con CSS (hojas de estilo en cascada)
- El XHTML y CSS que describen una web es lo que el servidor envía al navegador (junto con imágenes, vídeos, etc.)
- Las Webs las programan personas desarrollando código directamente o con la ayuda de programas de escritorio o aplicaciones web.

¿Cómo se programa una Web?



Contenido (XHTML)

Contenido+Forma
(XHTML+CSS)

WWW (World Wide Web)

- La World Wide Web (WWW) o Red informática mundial es un sistema de distribución de información basado en hipertexto o hipermedios enlazados y accesibles a través de Internet.
- Con un navegador web, un usuario visualiza sitios web compuestos de páginas web que pueden contener texto, imágenes, vídeos u otros contenidos multimedia, y navega a través de ellas usando hiperenlaces.
- La Web fue creada alrededor de 1989 por el inglés Tim Berners-Lee con la ayuda del belga Robert Cailliau mientras trabajaban en el CERN en Ginebra, Suiza, y publicado en 1992.

Modelo cliente-servidor

- La web funciona siguiendo un modelo cliente-servidor en una red TCP/IP, local o interconectada con las demás redes a través de Internet.
- Cliente: el navegador solicita a un servidor Web el envío de páginas de información. Lo que recibe es un documento de texto HTML, que deberá interpretar.
- Servidor: Atiende las peticiones procedentes de los clientes HTTP.

HTTP (HyperText Transfer Protocol)

- HTTP (HyperText Transfer Protocol): Protocolo de Transferencia de Hipertexto, creado para compartir datos científicos a nivel internacional y de forma instantánea.
- Es el método más común de intercambio de información en la web.
- Descrito en el RFC 1945 (1996) y ampliado y modificado en otros RFCs, el más utilizado es el 2616 (1999) que define la versión 1.1
- Existe una versión segura: HTTPS
- Hoy en día HTTP(S) no se usa sólo para transferir hiper-texto, sino también otros datos (imágenes, vídeo, monitorizar un portaaviones, etc.)...aunque a menudo no sea el protocolo más óptimo para la tarea.

HTTP: Comunicación

- Especificado en el RFC 2616. El contenido de los mensajes son líneas de texto, que contienen ordenes y parámetros con la sintaxis definida.
- Cada transacción es una comunicación distinta (sin estado).
- Dos tipos de mensaje: petición (*request*) y respuesta (*response*).
- Formato del mensaje:
  - Línea de comienzo: tipo de mensaje (orden HTTP con sus parámetros, *request* o *response*).
  - Líneas de encabezado (si son obligatorias) o cero (si son opcionales, acabadas con un CR-LF
  - Separador (CR-LF)
  - Contenido o cuerpo del mensaje

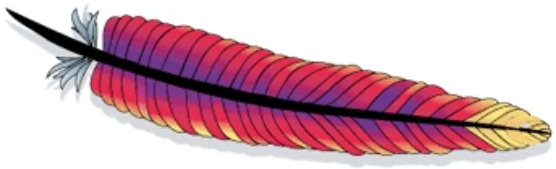
HTTP: Métodos de petición (request)

- Formato de petición básico: método URI versión
- El método indica al servidor que debe hacer con el URI [RFC 2396].
- La versión indica la versión del protocolo que el cliente entiende.
- Ejemplo: `GET /index.html HTTP/1.0`
- La versión HTTP/1.0 contempla 3 métodos: `GET`, `HEAD` y `POST`.
- La versión HTTP/1.1 añade: `PUT`, `OPTIONS`, `DELETE`

HTTP: Métodos de respuesta (response)

- Formato de respuesta básico: `versión código_estado texto_explicativo`
- Ejemplo: HTTP/1.1 405 Method Not Allowed
- Ejemplo: HTTP/1.1 200 Ok
- Los códigos se clasifican en 5 grupos:
  - Códigos 1xx: informativos
  - Códigos 2xx: éxito de la solicitud
  - Códigos 3xx: redireccionar la solicitud
  - Códigos 4xx: error generado por el cliente
  - Códigos 5xx: error generado por el servidor

Índice



Servidores HTTP

- Datos de evolución de uso de servidores web a lo largo de los años: http://news.netcraft.com/archives/category/web-server-survey/.
- Comparativa de servidores web: http://en.wikipedia.org/wiki/Comparison_of_web_servers.

Servidores HTTP

- NCSA HTTPd: Uno de los primeros, además gratuito. Actualmente el proyecto está abandonado y el sitio web oficial recomienda utilizar Apache.
- Apache: basado en NCSA, es el servidor más utilizado en Internet. Dispone de muchos módulos para ampliar su funcionalidad.
- lighttpd: optimizado para eficiencia. Se usa en sitios como Youtube, Wikipedia, Sourceforge...
- nginx: menos funcional, pero muy eficiente. Suele utilizarse como servidor proxy-inverso para diversos protocolos (HTTP, SMTP, POP3 e IMAP)
- Cherokee: otro servidor muy eficiente, multi-plataforma y con un panel de administración muy cómodo.
- Internet Information Server (IIS): Desarrollado por Microsoft para sistemas Windows Server.

El servidor Apache

Apache

El servidor HTTP Apache es un servidor web HTTP de código abierto, para plataformas Unix (BSD, GNU/Linux, etc.), Microsoft Windows, Macintosh y otras, que implementa el protocolo HTTP/1.12 y la noción de sitio virtual.

Ventajas

- Modular
- Software libre
- Multi-plataforma
- Extensible
- Popular (fácil conseguir ayuda/soporte)

Desventajas

Algunos le acusan de no ser eficiente, arquitectura del código obsoleta, etc.

El servidor Apache: módulos

Algunos módulos:

- `mod_ssl` - Comunicaciones Seguras vía TLS.
- `mod_rewrite` - reescritura de direcciones
- `mod_deflate` - Compresión transparente con el algoritmo *deflate* del contenido enviado al cliente.
- `mod_auth_ldap` - Permite autentificar usuarios contra un servidor LDAP.
- `mod_cband` - Control de tráfico y limitador de ancho de banda.
- `mod_perl` - Páginas dinámicas en Perl.
- `mod_php` - Páginas dinámicas en PHP.
- `mod_python` - Páginas dinámicas en Python.
- `mod_ruby` - Páginas dinámicas en Ruby.
- ...

El servidor Apache: LAMP

LAMP

- Linux, el sistema operativo;
- Apache, el servidor web;
- MySQL, el gestor de bases de datos;
- Perl, PHP, o Python, los lenguajes de programación.

El servidor Apache: Documentación I

- Versión 2.4 de la documentación del Servidor de HTTP Apache: http://httpd.apache.org/docs/2.4/
- El Servidor Apache y Programas de Soporte: http://httpd.apache.org/docs/2.4/programs/
- Ficheros de configuración: http://httpd.apache.org/docs/2.4/configuring.html
- Puertos y direcciones de escucha: http://httpd.apache.org/docs/2.4/bind.html
- Iniciar y Parar el servidor Apache: http://httpd.apache.org/docs/2.4/stopping.html
- Índice de directivas de configuración: http://httpd.apache.org/docs/2.4/mod/directives.html
- Índice de Módulos: http://httpd.apache.org/docs/2.4/mod/

El servidor Apache: Documentación II

- Secciones de Configuración:
  http://httpd.apache.org/docs/2.4/sections.html
- Mapear URLs a partes del sistema de ficheros:
  http://httpd.apache.org/docs/2.4/urlmapping.html

Descarga, instalación y compilación

Utiliza el script siguiente para descargar, configurar la instalación e instalar Apache:

```
mkdir $HOME/httpd
cd $HOME/httpd
wget http://ftp.wayne.edu/apache/httpd/httpd-2.4.2.tar.bz2
tar jxvf httpd-2.4.2.tar.bz2
cd httpd-2.4.2
./configure --prefix=$HOME/httpd/binarios
sed -e 's/rv = apr_file_link/\/\/rv = apr_file_link/' \
  support/rotatelogs.c > support/rotatelogs.c.new
mv support/rotatelogs.c support/rotatelogs.c.bak
mv support/rotatelogs.c.new support/rotatelogs.c
make
make install
cd $HOME/httpd/binarios
```

Ejercicios I

1. Recorre las carpetas del servidor observando qué se guarda en ellas ellas.
2. Inicia y para el servidor Apache con `apachectl`. Recuerda que tendrás que llamar a `apachectl restart` cada vez que hagas un cambio en la configuración.
3. Cambia las siguientes directivas:
   - El número de puerto para poder arrancar el servidor en tu máquina: Listen
   - El nombre de la máquina: ServerName
   - La carpeta desde la que se servirán recursos: DocumentRoot
   - Usuario y grupo para el demonio.
4. **IMPORTANTE**: para evitar que Apache siga ejecutándose al salir de nuestra sesión, debemos añadir el fichero `.bash_logout` al directorio $HOME con una línea similar a `killall httpd -9`.

Ejercicios II

5. Prueba a hacer un telnet al puerto del servidor (telnet localhost 8080) y a mandarle mensajes HTML (escribe GET / HTTP/1.0 y dale dos veces a intro). Puedes probar a escribir MELON y darle dos veces a intro.
6. Cambia el fichero por defecto que se envía al cliente: DirectoryIndex
7. Redirecciona la dirección /u a www.uco.es.
8. Crea un *Host* virtual de manera que cuando un cliente se conecte al servidor usando `localhost:8080` el servidor muestre una web, y cuando se conecte usando `IPMAQUINA:8080` muestre otra. Personaliza todas las directivas (directorio raiz, nombre de los ficheros de log y la directiva `<Directory>`).
9. Protege una carpeta del servidor para que sólo una serie de usuarios puedan acceder a esta carpeta.