Servizo HTTP

Sumario

- 1 Características xerais dun servidor web
  - 1.1 Funcionamento
  - 1.2 Os *Uniform Resource Locators*
  - 1.3 Conexións simples ou múltiples
- 2 Protocolo HTTP
  - 2.1 Solicitudes
  - 2.2 Respostas
  - 2.3 As cookies
- 3 Instalación e configuración do cliente HTTP
- 4 Instalación e configuración do servidor Web
  - 4.1 Instalación
  - 4.2 Configuración xeral
  - 4.3 Hosts virtuais
    - 4.3.1 Hosts virtuais baseados en direccións IP
    - 4.3.2 Hosts virtuais baseados en nomes
  - 4.4 Acceso mediante clave
    - 4.4.1 Restrición de acceso a usuarios
    - 4.4.2 Restrición de acceso a máquinas
  - 4.5 HTTP seguro
    - 4.5.1 Habilitar o módulo mod_ssl
    - 4.5.2 Creación dun certificado
    - 4.5.3 Configuración do HTTPS
  - 4.6 Outras configuracións
    - 4.6.1 Soporte para PHP
    - 4.6.2 Listado do contido dun directorio
    - 4.6.3 Personalizar as mensaxes de Apache
    - 4.6.4 XAMMP

Características xerais dun servidor web

O protocolo HTTP (*Hypertext Transfer Protocol*) permite a transferencia de páxinas web entre un servidor web (servidor HTTP) e un navegador (cliente HTTP). É o método máis común de intercambio de información polo que os neófitos, frecuentemente, confúndeno coa totalidade da rede, é dicir, úsano como sinónimo de Internet.

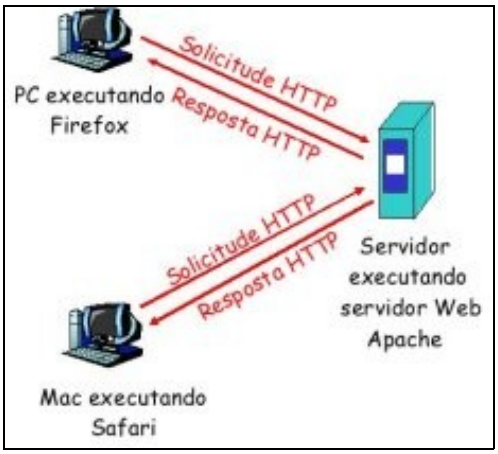
Este protocolo está descrito no RFC 1945 e foi ampliado e modificado po outros, como o RFC 2068, RFC 2069, RFC 2109, RFC 2145, RFC 2169, RFC 2227, RFC 2295, RFC 2296, RFC 2518 e RFC 2585 (entre outros).

O HTTP naceu no 1989 no CERN (Centro Europeo de Investigación Nuclear). A Web xurdiu pola necesidade de lograr que os equipos de investigadores dispersos internacionalmente colaborasen, usando un conxunto sempre cambiante de informes, planos, debuxos, fotos e outros documentos.

A proposta inicial dunha rede (web) de documentos vinculados xurdiu do físico do CERN **Tim Berners-Le**. O primeiro prototipo estaba baseado só en texto. En decembro de 1991 fíxose unha demostración pública e o desenvolvemento continuou durante o seguinte ano, culminando coa liberación dunha interface gráfica, Mosaic, no ano 1993. Mosaic tivo tanto éxito que, un ano despois, o seu autor, Marc Andreeseen formou a compañía, Netscape Communications Corp.

En 1994, o CERN e o M.I.T. asinaron un acordo para establecer o World Wide Web Consortium, unha organización adicada ao desenvolvemento da Web, a estandarización de protocolos e o fomento da interoperabilidade entre as instalacións. Berners-Le converteuse no director.

Funcionamento



O HTTP é un protocolo do nivel de aplicación para a Web. Segue o modelo ou **arquitectura cliente/servidor**, tal e como se ve na figura. O cliente é un navegador (*browser*) que solicita, recibe, e mostra obxectos Web. O servidor Web envía obxectos como resposta ás peticións do cliente. Hai dúas versións do protocolo, o HTTP 1.0, especificada no RFC 1945, e o HTTP 1.1, especificado no RFC 2068.

HTTP **utiliza TCP como protocolo de transporte**. Os clientes inician conexións TCP (crean un socket) co servidor, no porto 80. Os servidores aceptan conexións TCP dos clientes. Unha vez establecida a conexión intercámbianse mensaxes HTTP (mensaxes do protocolo do nivel de aplicación) entre o cliente (cliente HTTP) e o servidor Web (servidor HTTP). Finalmente, péchase a conexión TCP.

HTTP non ten **?estados?** (*stateless*). Isto quere dicir que o servidor non mantén información sobre solicitudes anteriores dos clientes. Deseñouse así porque os protocolos que manteñen o estado son complexos, xa que hai que manter un histórico do que acontece (estado). Se o cliente ou o servidor fallan pode haber inconsistencias e deberán volver ao estado anterior, co correspondente consumo de recursos.

Os *Uniform Resource Locators*

As **páxinas Web** conteñen **obxectos**. Os obxectos poden ser ficheiros HTML, imaxes, ficheiros de audio, vídeo, etc. Unha páxina web é un **ficheiro HTML base** que inclúe varias referencias aos obxectos. A cada obxecto accédese mediante un **URL** (*Uniform Resource Locator*) que ten a seguinte forma:

Diante do nome da máquina, normalmente inclúese o protocolo que permite o acceso ao recurso, como por exemplo, `http://`, `ftp://`, `smb://`, etc.

Conexións simples ou múltiples

As conexións HTTP poden ser de dous tipos:

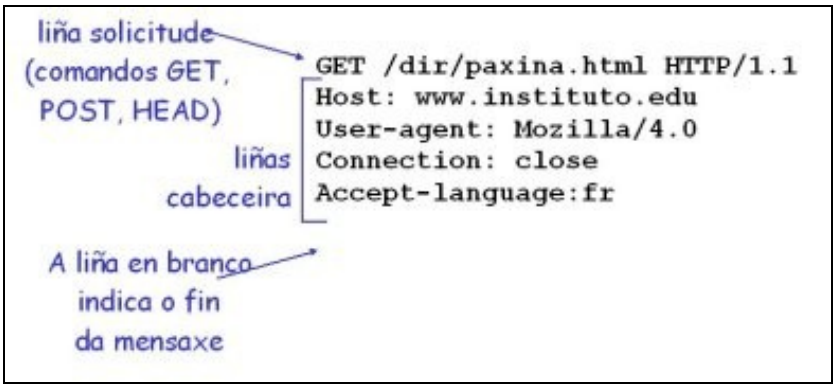
- **Conexión non persistentes**. Nelas, envíase un obxecto como máximo en cada conexión TCP. Para cada nova petición do cliente precísase unha nova conexión TCP. HTTP/1.0 utiliza HTTP non persistente. Por exemplo, se unha páxina Web ten referencias a 10 obxectos, abriranse 10 conexións TCP.
- **Conexións persistentes**. Envíanse varios obxectos nunha única conexión TCP entre o cliente e o servidor. HTTP/1.1 utiliza conexións persistentes por defecto. As conexións persistentes supoñen menos sobrecarga para o SO xa que os navegadores non abren varias conexións TCP paralelas para conseguir os obxectos referenciados. O servidor deixa a conexión TCP aberta despois de enviar unha resposta, xa que logo, as mensaxes HTTP entre o mesmo cliente e servidor usan a mesma conexión TCP.

Protocolo HTTP

En HTTP hai mensaxes de **solicitude** e mensaxes **resposta**. Ambas as dúas mensaxes están en formato ASCII, polo que poden verse e lerse se capturamos unha sesión HTTP cun analizador de rede como, por exemplo, o Wireshark.

Solicitudes

Unha mensaxe HTTP de solicitude ten o seguinte formato:



Se queremos facer unha solicitude de envío de datos (por exemplo, cando facemos unha consulta nun buscador enviámoslle ao servidor a palabra a buscar, ou cando enviamos un formulario) podemos usar dous métodos:

- **Método POST**, no que os datos se envían ao servidor dentro da mensaxe de solicitude HTTP.
- **Método URL**, tamén chamado método GET, no que os datos se envía na URL da liña de solicitude, por exemplo

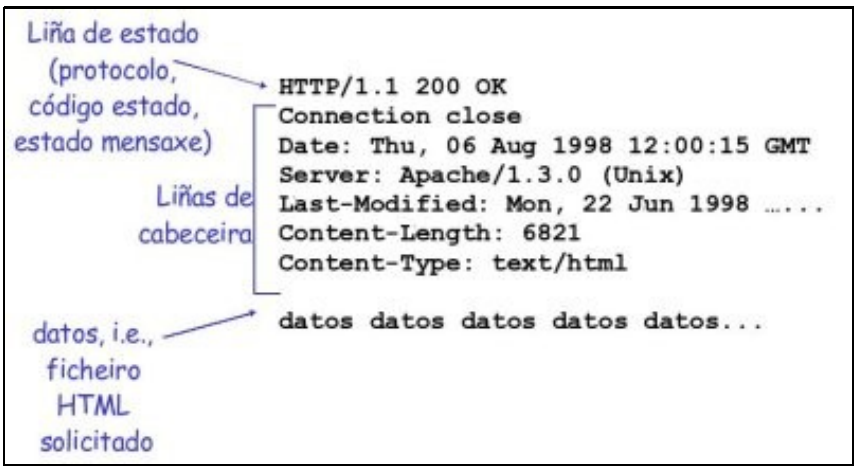
```
www.unsite.com/buscaAnimal?mono&banana
```

Dependendo da versión do protocolo que se use están soportadas distintas mensaxes HTTP (tamén coñecidas como métodos):

- **Para HTTP 1.0**: están soportadas as mensaxes **GET** e **POST** xa vistas. Tamén está soportada a mensaxe **HEAD** que só devolve a cabeceira, non o obxecto referenciado
- **Para HTTP 1.1**: están soportadas as tres mensaxes anteriores e tamén **PUT**, que permite subir o ficheiro especificado no URL ao servidor, e **DELETE**, que permite borrar o ficheiro especificado no URL.

Respostas

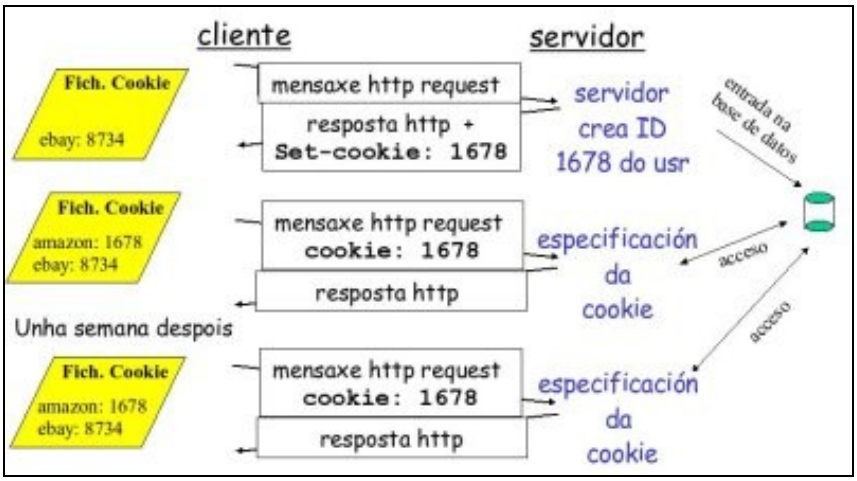
As mensaxes HTTP de resposta teñen un formato similar ás de solicitude:



As mensaxes de resposta inclúen un código dependendo do significado da mesma. Dito código vai na primeira liña da mensaxe (liña de estado). Algúns exemplos son os seguintes:

```
200 OK
  Solicitude correcta, envíarase o obxecto solicitado nesta mensaxe

301 Moved Permanently
  Obxecto solicitado noutro URL, especifícase a nova localización nesta mensaxe (Location:)

400 Bad Request
  Mensaxe non entendida polo servidor

404 Not Found
  Obxecto solicitado non atopado no servidor

505 HTTP Version Not Supported
```

As cookies



Dixemos que HTTP é un protocolo sen estado. Isto, ás veces, pode non ser o desexado. Por exemplo, supoñamos que temos unha web na que as súas páxinas están restrinxidas a determinados usuarios. Se non mantemos o estado do cliente teremos que estar solicitándolle para cada páxina que queira ver o seu nome de usuario e clave. HTTP resolve este escenario coas cookies.

As cookies están definidas no RFC 2109 e permiten monitorizar a navegación do usuario. Moitos sites usan cookies. Son útiles, por exemplo, para validar un usuario, escoller opcións personalizadas (idioma, etc.), rexistrar hábitos de navegación, etc. Están formadas por catro compoñentes:

1. Liña de cabeceira da cookie na resposta HTTP.
2. Liña de cabeceira da cookie na solicitude HTTP.
3. Ficheiro que se mantén no computador do cliente, xestionado polo navegador.
4. Base de datos no servidor.

Un exemplo de funcionamento das cookies podémolo ver na figura.

Instalación e configuración do cliente HTTP

Instalación e configuración do servidor Web

Apache, dispoñible tanto para Linux como para Windows, é o servidor web mais utilizado hoxe en día. Actualmente, o servidor Apache está na versión 2.0 (paquete apache2) . En GNU/Linux, o servidor apache correspóndese co servizo httpd. Está deseñado cunha estrutura baseada en módulos, de tal xeito que ademais do servidor existen diferentes módulos que lle engaden diversas capacidades.

Instalación

Para instalar Apache hai que teclear o seguinte (hai que ter configurada correctamente a rede para poder acceder aos repositorios):

```
sudo apt-get install apache2
```

Para verificar que o servizo se está executando (de momento, coas opcións por defecto) pódese teclear:

```
netstat -tl
```

Ou ben:

```
ps aux | grep apache
```

Tamén se pode abrir un navegador e teclear na barra de direccións `http://localhost` para ver a páxina de proba do servidor. As páxinas que se mostran no servidor por defecto atópanse no directorio `/var/www/`.

Configuración xeral

Igual que o vsFTPD crea un usuario FTP, Apache crea un usuario por defecto que é o www-data.

O ficheiro principal de configuración do servidor Apache é `/etc/apache2/apache2.conf` e contén moitas directivas que determinan o seu comportamento. A información de cada unha está explicada no propio ficheiro de configuración. Se non se entende algún parámetro existe abundante documentación en liña. Outros ficheiros importantes de configuración son:

- /etc/apache2/httpd.conf
- /etc/apache2/sites-available/default (permite crear hosts virtuais)
- /etc/apache2/ports.conf (permite cambiar os portos de escoita do Apache)

Antes de modificar un ficheiro de configuración convén facer unha copia de seguridade. Para que cada un dos cambios que se fagan nos parámetros hai que reiniciar o servidor, ou mellor aínda, en lugar de reinicialo, simplemente indicarlle que relea a nova configuración mediante o seguinte comando:

```
sudo /etc/init.d/apache2 force-reload
```

Hosts virtuais

Tal e como está configurado por defecto o Apache servirá un único sitio Web de xeito anónimo. Isto non é o habitual, xa que teríamos que ter un servidor para cada Web que quixeramos servir. Existen dúas formas de configurar Apache coa finalidade de servir diferentes sitios Web:

- Hosts virtuais baseados en direccións IP
- Hosts virtuais baseados en nomes

Hosts virtuais baseados en direccións IP

A directiva `<VirtualHost></VirtualHost>`, no ficheiro `/etc/apache2/sites-available` fai crer ao usuario que se conecta a unha Web mediante unha dirección IP (ou un nome de dominio) co navegador que accede a diferentes servidores pero realmente están todos nunha única máquina. Por exemplo, para incluír tres sitios Web servidos a través de tres direccións IP distintas incluiríamos no ficheiro anterior a seguinte información:

```
# www.web1.gal
<VirtualHost 192.168.1.100>
        DocumentRoot /var/www/web1/
</VirtualHost>

# www.web2.gal
<VirtualHost 192.168.1.101>
        DocumentRoot /var/www/web2/
</VirtualHost>

# www.web3.gal
<VirtualHost 192.168.1.102>
        DocumentRoot /var/www/web3/
</VirtualHost>
```

Así teriamos os tres sitios Web executándose no porto 80 pero se o que se quere é que a primeira das Webs estea accesible nos portos 80 e 2000, a segunda nos portos 80 e 2001 e a terceira nos portos 80 e 2002, habería que modificar o ficheiro `/etc/apache2/ports.conf` para que a directiva `listen` teña os seguintes valores:

```
Listen 192.168.1.100:80
Listen 192.168.1.100:2000

Listen 192.168.1.101:80
Listen 192.168.1.101:2001

Listen 192.168.1.102:80
Listen 192.168.1.102:2002
```

Hosts virtuais baseados en nomes

Unha alternativa á práctica anterior é a configuración de hosts virtuais baseados en nome. Esta opción permite a Apache servir diferentes sitios Web cunha única dirección IP, distinguindo entre eles mediante un nome de dominio diferente. Para iso, úsase a directiva `NameVirtualHost dirección IP`

para facer crer ao usuario que teclea un nome de dominio no navegador que accede a diferentes servidores, pero realmente están todos nunha única IP.

Para incluír a directiva hai que editar o ficheiro de configuración `/etc/apache2/sites-available/default`. O seguinte é un exemplo para tres sitios web:

```
#Páxina de inicio por defecto
DirectoryIndex  inicio.html

#Dirección IP do host virtual (virtualización por nome)
NameVirtualHost 192.168.1.200

#www.meusitioweb1.gal
<VirtualHost 192.168.1.200>
        DocumentRoot    /var/www/meusitioweb1/
        ServerName      www.meusitioweb1.gal
</VirtualHost>

#www.meusitioweb2.gal
<VirtualHost 192.168.1.200>
        DocumentRoot    /var/www/meusitioweb2/
        ServerName      www.meusitioweb2.gal
</VirtualHost>

#www.meusitioweb3.gal
<VirtualHost 192.168.1.200>
        DocumentRoot /var/www/meusitioweb3/
        ServerName      www.meusitioweb3.gal
</VirtualHost>
```

Acceso mediante clave

En ocasións pode interesarnos restrinxir o acceso a un sitio Web (ou a unha determinada carpeta do sitio Web).

Hai que ter en conta que a funcionalidade do servidor Apache depende dos módulos que teña cargados. Os módulos básicos relacionados co control de acceso son `mod_auth` e `mod_access`, ambos cargados por defecto co servidor.

Restrición de acceso a usuarios

As restricións de acceso se realizan sobre o contido de directorios. Por iso, existe unha directiva chamada `Directory` que conterá outras directivas que afectarán ao acceso ao contido dese directorio. A sintaxe da directiva é a seguinte:

```
<Directory ruta do directorio afectado>
  #Directivas de configuración que afectan ao directorio especificado
  AuthType        Basic                 #Tipo autenticación
  AuthName        "Acceso restrinxido"  #Mensaxe informativa
  AuthUserFile    seguridade/usuarios   #Ruta relativa ao ficheiro
  Require user    alum1 alum2 alum3     #Lista usuarios permitidos
</Directory>
```

Para poder facer referencia aos usuarios alum1, alum2 e alum3, do exemplo anterior primeiro hai que crealos e asignarlles unha clave co comando `htpasswd`. Para iso pódese teclear o seguinte:

```
sudo htpasswd -c /etc/apache2/seguridade/usuarios alum1
```

O parámetro `-c` crea o ficheiro `AuthUserFile` onde se almacenan os usuarios autorizados. Polo tanto, só é necesario utilizar ese parámetro a primeira vez. Para crear o resto de usuarios habería que teclear o seguinte:

```
sudo htpasswd /etc/apache2/seguridade/usuarios alum2
sudo htpasswd /etc/apache2/seguridade/usuarios alum3
```

A ruta relativa ao ficheiro `seguridade/usuarios` ten en conta a directiva `ServerRoot` do ficheiro de configuración do servidor.

Apache tamén ten módulos para validar os usuarios contra unha base de datos ou mediante un servizo de directorio como LDAP, pero hai que ter os módulos correspondentes cargados.

Restrición de acceso a máquinas

Se se quere restrinxir o acceso a unha máquina (ou conxunto de máquinas) utilizaranse as directivas allow e deny. Por exemplo:

```
Allow from 192.168.1.1
```

Tamén se poden usar nomes de dominio:

```
Allow from iessanclemente.net
```

Ou unha subrede completa:

```
Allow from 192.168.1
```

A sintaxe de deny é análoga.

A directiva Order permite combinar as dúas directivas anteriores para que non se interfiran, por exemplo:

```
Order allow deny
Allow from 192.168.1.100 192.168.1.200
Deny from all
```

HTTP seguro

Ata agora, a interacción entre o cliente e o servidor faise en texto claro ou plano. Xa que logo, calquera que analice a rede pode roubar eses datos e entrar no sitio web. Para evitalo podemos establecer comunicacións seguras entre o cliente e o servidor seguindo tres pasos:

1. Habilitar o módulo mod_ssl
2. Crear un certificado para o servidor
3. Configurar o HTTPS para que soporte o certificado

Habilitar o módulo mod_ssl

O módulo mod_ssl de Apache2 (vén instalado por defecto pero hai que habilitalo) proporciona a capacidade de encriptar os datos mediante SSL (*Secure Socket Layer*). Cando nos queremos conectar a un sitio web con estas características hai que usar o prefixo https:// ao principio do URL, na barra de direccións do navegador. SSL está baseado en criptografía de clave pública ou asimétrica.

En xeral, para habilitar un módulo úsase o comando `a2enmod` (*Apache2 Enable Module*). Polo tanto, hai que teclear o seguinte para habilitar o módulo mod_ssl:

```
sudo a2enmod ssl
```

Para que os cambios se apliquen hai que recargar a configuración do Apache2 ou reinicialo.

Nota.- Para instalar calquera módulo de Apache2 úsase a ferramenta apt-get.

Creación dun certificado

Para que HTTPS funcione hai que crear un certificado no servidor. Seguindo os seguinte pasos:

- Xeneración dunha clave privada para o servidor:

```
sudo openssl genrsa -des3 -out server.key 1024
```

Obterase unha saída similar á seguinte e pedirá unha clave:

```
Generating RSA private key, 1024 bit long modulus
.++++++
.....................++++++
```

```
e is 65537 (0x10001)

Enter pass phrase for server.key:

Verifying - Enter pass phrase for server.key:
```

A clave que almacénase no ficheiro server.key.

- Xeneración do ficheiro *Certificate Signing Request*, tecleando:

```
sudo openssl req -new -key server.key -out server.csr
```

O ficheiro CSR contén información sobre a entidade que posúe o certificado. Hai que cubrir os datos necesarios. Se o certificado está asinado por unha entidade de certificación (CA) envíase a esta para que comprobe que os datos son correctos. Neste caso isto non sucederá porque o certificado está asinado por nós mesmos.

- Xeneración do certificado tecleando o seguinte:

```
sudo openssl x509 -req -days 365 -in server.csr -signkey server.key -out server.crt
```

O certificado almacenarase no ficheiro server.crt.

Configuración do HTTPS

Hai que seguir os seguinte pasos:

1. Crear un cartafol, por exemplo, /etc/apache2/ssl, para almacenar o certificado e a clave.
2. Copiar os ficheiros server.key e server.crt que se xeneraron anteriormente nese cartafol.
3. Engadir as seguintes liñas ao ficheiro /etc/apache2/sites-available/default. Teñen que estar incluídas na sercción VirtualHost de www.esi2.gal, xusto baixo a directiva DocumentRoot.

```
SSLEngine on

SSLOptions +StrictRequire

SSLCertificateFile /etc/apache2/ssl/server.crt
SSLCertificateKeyFile /etc/apache2/ssl/server.key
```

Outras configuracións

Soporte para PHP

PHP (*PHP Hypertext Pre-processor*) é unha linguaxe de programación interpretada, deseñada para a creación de páxinas web dinámicas. Para que Apache sexa capaz de xestionar páxinas web creadas mediante PHP hai que instalar o módulo correspondente tecleando o seguinte:

```
sudo apt-get install php5 libapache2-mod-php5
```

O módulo PHP5 habilítase automaticamente no Apache2 ao instalalo. Pódese ver comprobando que existen os ficheiros /etc/apache2/mods-enabled/php5.conf e /etc/apache2/mods-enabled/php5.load. Se non fose hai que habilitar o módulo co comando:

```
sudo a2enmod php5
```

Para comprobar a configuración pódese crear un script index.php como o seguinte e configurar Apache2 para que o lea:

```
<?php
print_r (phpinfo());
?>
```

Listado do contido dun directorio

Cando se tenta acceder a un directorio, en ausencia dun ficheiro de índice, Apache mostra por defecto o contido do directorio. Se queremos impedir ao usuario ver o contido dun directorio, (bastaría con crear un ficheiro de índice, por exemplo index.html) hai que crear un ficheiro .htaccess no raíz do sitio

web que conteña a liña:

```
        Options -Indexes
```

Tamén poderiamos utilizar a directiva DirectoryIndex, que especifica que ficheiros actúan como índice por defecto, e que ficheiro mostrar no caso de que estes non se atopen. Por exemplo a liña:

DirectoryIndex index.php index.html index.htm /prohibido.php

Indicaría ao servidor que debe buscar os ficheiros index.php, index.html ou index.htm, nesa orde, e no caso de non atopar ningún deles, cargar o arquivo prohibido.php.

Personalizar as mensaxes de Apache

Pódense personalizar as mensaxes do Apache2, xa que están no ficheiro /etc/apache2/apache2.conf. Só hai que descomentar a liña correspondete, por exemplo, a liña ErrorDocument 404 /missing.html para que poña ErrorDocument 404 ?Erro: Páxina non localizada neste servidor?.

XAMMP

É un paquete libre que permite instalar o Apache, mysql, phpmyadmin e proftpd de forma fácil. Convén desinstalar os servizos que teñamos dispoñibles antes de instalalo. Máis información no seu sitio web: http://www.apachefriends.org/

-- *Arribi, novembro 2008*