O servizo web: Servidor Apache en Linux

Configuración básica de Apache en Linux

Índice

1. Configuración básica do servizo web en Linux

Neste apartado veremos os pasos para a instalación e configuración básica do servidor web Apache, que é o servidor web máis utilizado, sobre Debian.

En primeiro lugar, instalaremos o paquete do apache:

```
apt-get install apache2
```

Estrutura dos ficheiros de configuración de Apache

A partires da versión 2 de Apache, deixouse de establecer todas as directivas de configuración de Apache nun so ficheiro e pasouse a facelo en varios que se inclúen uns a outros, ademais de establecer toda a configuración baseada no uso de sitios virtuais.

En Debian, todos os ficheiros de configuración están dentro do directorio /etc/apache2.

O ficheiro de configuración é /etc/apache2/apache2.conf. Ten unha estrutura de ficheiro autoexplicativo, e nel atopamos un gráfico en modo texto, que nos explica como se inclúen os ficheiros de configuración:

```
# /etc/apache2/
# |-- apache2.conf
# | `-- ports.conf
# |-- mods-enabled
# | |-- *.load
# | `-- *.conf
# |-- conf-enabled
# | `-- *.conf
# `-- sites-enabled
```

```
# `-- *.conf
```

A explicación de cada ficheiro incluído no directorio /etc/apache2 é a seguinte:

- **apache2.conf**: O ficheiro principal de configuración de Apache2. Conten toda a configuración global.
- **conf-available**: este directorio conten ficheiros de configuración dispoñibles . Todos os ficheiros que en versións anteriores estaban en /etc/apache2/conf.d deberían moverse a /etc/apache2/conf-available.
- **conf-enabled**: mantén ligazóns simbólicas aos ficheiros en /etc/apache2/conf-available. Cando un ficheiro de configuración é enlazado mediante unha ligazón simbólica estará activo a seguinte vez que se reinicie Apache.
- **envvars**: ficheiro con variables de contorna e os seus valores.
- **mods-available**: este directorio contén ficheiros de configuración para cargar módulos e configuralos. Sen embargo, non todos os módulos teñen ficheiros de configuración específicos.
- **mods-enabled**: mantén ligazóns simbólicas aos ficheiros en /etc/apache2/mods-available. Cando un ficheiro de configuración de módulo é enlazado mediante un ligazón simbólica será activado despois do seguinte reinicio de Apache.
- **ports.conf**: Aloxa as directivas que determinan os portos TCP nos que Apache escoita e atende peticións.
- **sites-available**: Este directorio ten os ficheiros de configuración dos Host Virtuais de Apache. Os Host virtuais permiten que Apache sexa configurado para múltiples sitios con configuracións separadas.

- **sites-enabled**: Ao igual que mods-enabled, sites-enabled contén ligazóns simbólicas a ficheiros dentro de /etc/apache2/sites-available. De xeito similar, cando se crea unha ligazón simbólica o sitio virtual estará activo unha vez se reinicie Apache.
- **magic**: instrucións para determinar os tipos MIME baseándose nos primeiros bytes de cada ficheiro.

Para estes tres directorios nos que habitualmente se crean ligazóns simbólicas aos ficheiros que existen no directorio *available* correspondente, hai certas aplicacións utilidade para executar desde a liña de comandos. Estas utilidades son a2ensite, a2dissite, a2enmod, a2dismod, a2enconf e a2disconf para habilitar e deshabilitar sitios virtuais, módulos e fragmentos de configuración.

A maiores, outros ficheiros de configuración poden ser engadidos usando a directiva Include, que tamén poden ser incluídos empregando comodíns para incluír varios nunha soa directiva. Calquera directiva pode ser introducida nestes ficheiros de configuración. Os cambios so serán recoñecidos unha vez que se reinicie o servidor Apache.

Pódese comprobar a sintaxe dos ficheiros de configuración antes de iniciar o servidor mediante a execución do comando apachectl configtest ou coa opción -t.

2.1. Directivas básicas

Neste apartado imos recoller as directivas máis relevantes do servidor web Apache. O número total de directivas permitidas é moi grande; neste enlace pódese ver a documentación de todas as posibles directivas.

Listen

A directiva **Listen** dille ao servidor que acepte peticións so en algún(s) porto(s) específico(s) ou combinacións de enderezos e portos. Se unicamente se especifica unha directiva Listen o servidor escoita peticións nese porto en todos os seus enderezos. Se se especifica un enderezo IP xunto co porto, atenderá peticións nese porto e interface. Pódense poñer múltiples directivas Listen xunto con portos e enderezos. O servidor atenderá peticións en calquera deles.

Por exemplo, para facer que o servidor acepte peticións nos portos 80 e 8000 en todos os seus interfaces de rede usaremos:

```
Listen 80
Listen 8000
```

Para facer que o servidor acepte conexións no porto 80 para unha interface e o 8000 para outro, empregaremos

```
Listen 192.0.2.1:80
Listen 192.0.2.5:8000
```

Os enderezos IPv6 débense poñer encerrados entre corchetes coma no seguinte exemplo:

```
Listen [2001:db8::a00:20ff:fea7:ccea]:80
```

Se se emprega cifrado SSL e o módulo ssl está activo, debermos tamén activar o porto 443:

```
<IfModule ssl_module>
     Listen 443
</IfModule>
```

Nas versións modernas de apache, esta directiva aparece no ficheiro "ports.conf".

VirtualHosts

O termo *Virtual Host* refírese a práctica de executar máis dun sito (coma company1.exemplo.com e company2.exemplo.com) nunha mesma máquina. Os Virtual Hosts poden ser **baseados en IP**, querendo dicir que temos enderezos IP diferentes para sitio, **baseados en portos**, o que quere dicir que temos un porto para cada sitio, ou **baseados en nome**, que quere dicir que temos múltiples sitios web no mesmo enderezo e porto, simplemente con distintos nomes de DNS. O feito de que se executen na mesma máquina física non é apreciable polo usuario final.

Nos hosts virtuais baseados en nomes, o servidor analiza o que o cliente indica como nome de host nas cabeceiras das peticións HTTP. Para configuralos, definiremos un bloque <VirtualHost> para cada host diferente que se quere servir. Dentro de cada bloque <VirtualHost> necesítase polo menos unha directiva ServerName para designar que host virtual se sirve, e tamén unha directiva DocumentRoot que indique en que lugar do sistema de ficheiros reside o contido para ese virtual host. Tamén se pode incluír a directiva ServerAlias para configurar nomes alternativos. Exemplo:

```
<VirtualHost *:80>
   ServerName www.exemplo1.com
   ServerAlias exemplo1.com
   DocumentRoot "/www/exemplo1"
 </VirtualHost>
<VirtualHost *:80>
   ServerName www.exemplo2.com
   ServerAlias exemplo2.com
   DocumentRoot "/www/exemplo2"
 </VirtualHost>
```

O virtual hosting baseado en enderezos IP é un método para aplicar diferentes directivas baseadas no enderezo IP e porto no que se recibe a petición. Desta maneira sérvense diferentes sitios e en diferentes portos e interfaces de rede. Exemplo:

```
<VirtualHost 172.20.30.40:80>
   ServerAdmin webmaster@www1.exemplo.com
   DocumentRoot "/www/vhosts/www1"
 </VirtualHost>
<VirtualHost 172.20.30.50:80>
   ServerAdmin webmaster@www2.exemplo.org
   DocumentRoot "/www/vhosts/www2"
 </VirtualHost>
```

Apache2 ven de serie con unha configuración de host virtual por defecto amigable. Isto é, configurado cun único host virtual por defecto (empregando a directiva VirtualHost), que pode ser modificada ou usada como se houbese so un único sitio, ou tamén como modelo para host virtuais adicionais se se desexan configurar. Se se deixa so, este host virtual, funcionará coma sitio por defecto, ou tamén se a URL que introduce o usuario non encaixa con ningunha directiva ServerName de ningún outro host virtual. Para modificar o host virtual, edítase o ficheiro /etc/apache2/sites-available/000-default.conf.

As directivas establecidas para un host virtual, so se aplican a ese host virtual en particular. Se a directiva é de ámbito de servidor e non se define dentro do host virtual, úsase un valor por defecto. Por exemplo, pódese definir o enderezo de correo electrónico do Webmaster no ámbito do servidor, e non definilo dentro de cada host virtual de forma individual.

Se se quere configurar un novo sitio ou host virtual pódese copiar ese ficheiro dentro doe mesmo directorio cun nome a escoller por parte do usuario.

Alias

A directiva Alias permite acceder a documentos almacenados no sistema de ficheiros local nalgún lugar diferente a DocumentRoot. Cando un cliente solicite unha URL, aquelas que comecen na parte de ruta coa ruta que coincida cun alias serán mapeadas a ficheiros locais que comecen coa ruta de directorio indicada no alias. A ruta de URL é sensible a maiúsculas, incluso en sistemas de ficheiros non sensibles a elas.

```
Alias /image /ftp/pub/image
```

DirectoryIndex

A directiva DirectoryIndex establece a lista de recursos a atopar cando, cando o cliente solicita un índice do directorio especificando unha / ao final do nome do directorio. Exemplo:

```
DirectoryIndex index.html
```

Options

A directiva Options controla que características do servidor están activas nun directorio en particular. Pode tomar o valor de None, o cal indica que ningunha opción está habilitada ou unha ou máis das seguintes opcións:

- **All**: Todas as opcións excepto MultiViews. Este é o valor por defecto.
- **FollowSymLinks**: O servidor seguirá todos as ligazóns simbólicas deste directorio. Anque o servidor siga a ligazón simbólica non se cambiará o nome da ruta emparellada dentro da sección <Directory>.
- **Includes**: Permítense as inclusións do lado do servidor fornecidas polo módulo mod_include.

- **Indexes:** Se se solicita unha URL que se mapea a un directorio que non existe ningún dos ficheiros listados coa directiva DirectoryIndex (por exemplo, index.html) nese directorio, entón o módulo mod_autoindex devolverá un listado formatado do contido do directorio.
- **SymLinksIfOwnerMatch:** O servidor so seguirá a ligazón simbólica se os propietarios do enlace e do destino coinciden.

Normalmente, se se aplican múltiples Options a un directorio, entón so son de aplicación as máis específicas e as outras son ignoradas e non se mesturan. Sen embargo, se todas as opcións expresadas na directiva Options están precedidas por un símbolo + ou -, entón mestúranse as opcións. Toda opción precedida por un + engádense a lista de opcións actuais, e calqueras precedidas por un – réstase da lista. Non se permite mesturar Options con un + ou un – con outras sen símbolos, e é causa de resultados impredecibles.

2.2. Seccións de configuración e módulos

Seccións de configuración

O bloque <VirtualHost> dálle aos administradores a capacidade de modificar o comportamento do servidor web a nivel de host ou de dominio. Calquera opción especificada nun <VirtualHost> apícase ao sitio enteiro. Sen embargo, non prové a capacidade de especificar opcións a nivel de directorio. Afortunadamente, Apache dispón de outras posibilidades para especificar configuración.

Os contedores máis empregados son aqueles que permiten cambiar algunha configuración en determinados lugares do sistema de ficheiros (a vista dos discos locais vistos polo sistema operativo) ou do espazo web (a vista do sitio ofrecida polo servidor web tal e como o ve o cliente).

As directivas **<Directory>** e **<Files>**, xunto cos seus correspondentes parámetros con expresións regulares, aplícanse directamente a partes do sistemas de ficheiros.

- As directivas incluidas nunha sección <Directory> aplícase á porción de sistema de ficheiros correspondente ao directorio especificado, e a todos os seus subdirectorios dentro dese directorio (así coma tamén a todos os ficheiros neses directorios). Por exemplo, na seguinte configuración habilítase a opción Indexes para o directorio /var/web/dir1 e todos os seus subdirectorios.

```
<Directory "/var/web/dir1">
   Options +Indexes
</Directory>
```

- As directivas englobadas nunha sección <Files> aplícanse a calquera ficheiro cun nome especificado. Así, por exemplo, as seguintes directivas de configuración ubicadas no ficheiro de configuración global, denegan o acceso a calquera ficheiro chamado private.html independentemente de onde se atope.

```
<Files "private.html">
   Require all denied
</Files>
```

Para referirse a ficheiros que se atopan en lugares particulares do sistema de ficheiros, as seccións <Files> e <Directory> poden estar combinadas. Por exemplo, a seguinte configuración denegará o acceso aos ficheiros /var/web/dir1/private.html, /var/web/dir1/subdir2/private.html, /var/web/dir1/subdir3/private.html, e calquera outra instancia de private.html que se atope debaixo do directorio /var/web/dir1/.

```
<Directory "/var/web/dir1">
   <Files "private.html">
     Require all denied
```

```
   </Files>
</Directory>
```

Se aínda así se necesita un emparellamento máis flexible, **<DirectoryMatch>** e **<FilesMatch>** permiten o uso de expresións regulares perl-compatibles. Empregado expresións regulares, podemos denegar o acceso a moitos tipos de imaxe da seguinte maneira:

```
<FilesMatch "\.(bmp|gif|jpe?g|png)$">
   Require all denied
</FilesMatch>
```

A directiva **<Location>** e a súa correspondente que usa expresións regulares **<LocationMatch>**, cambian a configuración para contido situado no espazo web. Por exemplo, a seguinte configuración prevén o acceso a calquera ruta de URL que comece con /private, e aplícase a peticións tales coma http://yoursite.exemplo.com/private, http://yoursite.exemplo.com/private123, e http://yoursite.exemplo.com/private/dir/file.html así coma a calquera outra petición que comece coa /private.

```
<LocationMatch "^/private">
   Require all denied
</LocationMatch>
```

Ficheiros .htaccess

Os ficheiros **.htaccess** (ou "ficheiros de configuración distribuídos") proven unha vía para facer cambios na configuración a nivel de directorio. Un ficheiro que conten unha ou máis directivas de configuración colócase nun directorio en particular, e as directivas aplícanse a ese directorio e a todos os seus subdirectorios.

Así descentralizase a administración do servidor web. Calquera opción especificada unha sección <Directory> pode especificarse, como norma xeral, dentro dun ficheiro .htacces. Os ficheiros

.htaccess son moi útiles en casos no que o operador do sitio web ten acceso para editar ficheiros no directorio publico do sitio web, pero non nos ficheiros de configuración de Apache, o que é moi habitual se temos o noso sitio web sobre un servizo de aloxamento.

O que poñemos neses ficheiros ven determinado polo valor da directiva AllowOverride. Esta directiva especifica, en categorías, que directivas se lles permite aparecer nos ficheiros .htaccess. Cando o servidor atopa un ficheiro .htaccess, necesita saber que directivas declaradas nese ficheiro poden anular ou cambiar outras declaradas previamente. A directiva AllowOverride so é valida en seccións <Directory> especificadas sen expresións regulares, e non están permitidas en seccións <Location>, <DirectoryMatch> ou <Files>.

Cando esta directiva ten o valor None e a directiva AllowOverrideList tamén ten o valor None, os ficheiros .htaccess ignóranse completamente. Nese caso, o servidor nin sequera os intenta ler do sistema de ficheiros. Cando toma o valor All, entón permítese calquera directiva válida no contexto .htaccess nos ficheiros .htaccess.

Módulos

A inclusión de módulos para ampliar a funcionalidade de Apache, faise coa directiva LoadModule seguida do nome do módulo e o nome de ficheiro (ou libraría) que o contén. Serán necesarias tantas directivas LoadModule coma módulos que vaian a ser empregados.

En Debian/Ubuntu os módulos poden ser habilitados ou deshabilitados cos comandos a2enmod e a2dismod. O nome do módulo encaixa co nome do ficheiro que o contén (sen a extensión .conf). Exemplo:

```
a2enmod usertrack
a2dismod usertrack
```

Sempre é obrigatorio, ao habilitar ou deshabilitar módulos reiniciar o servizo Apache2.

2.3. Control de acceso por equipos

O control de acceso pode facerse a través de varios módulos. Os máis importantes deles son mod_authz_core e mod_authz_host. Se se quere restrinxir o acceso a porcións dun sitio web baseándose no enderezo IP do visitante, a maneira máis sinxela é empregar o módulo mod_authz_host.

A directiva Require ten unha variedade grande de maneiras de permitir ou denegar acceso a recursos. En conxunción coas directivas de agrupamento RequireAll, RequireAny, e RequireNone, estes requirimentos pódense combinar de innumerables maneiras, para conseguir a política de acceso que se desexa. A maneira de usar estas directivas é a seguinte:

```
Require host address
Require ip ip.address
```

Na primeira delas, o enderezo é un nome totalmente cualificado (FQDN) ou un nome parcial. Tamén se poden poñer múltiples nomes ou enderezos, se é necesario. Na segunda maneira, ip.address é un enderezo IP, un enderezo IP parcial, unha parella enderezo de rede/máscara ou un bloque CIDR. Tamén se poden empregar bloques CIDR. Exemplos:

```
Require ip 10.1.2.3
Require ip 192.168.1.104 192.168.1.205
Require ip 10.1
Require ip 10.1.0.0/255.255.0.0
Require ip 10.1.0.0/16
Require host exemplo.org
Require host .net exemplo.edu
```

Pódese inserir a palabra not para negar un requirimento particular. Nese caso, como not é a negación dun valor, por si so non permite ou denega unha petición xa que non verdadeiro, non implica falso. Dese xeito, para negarlle a visita usando unha negación, o bloque

debe ter polo menos *un elemento que se avalíe a verdadeiro ou falso.* Por exemplo, se temos alguén atacando ao servidor, e queremos denegarlle o acceso, podemos facelo da seguinte maneira:

```
<RequireAll>
   Require all granted
   Require not ip 10.252.46.165
</RequireAll>
```

Contedores de autorización

A directiva Require comproba se un determinado usuario pode acceder de acordo cun provedor de autorización e unhas restricións especificadas. O módulo mod_authz_core proporciona varios provedores de autorización dos máis empregados son

- **Require all granted**: Acceso incondicional permitido.
- **Require all denied**: Acceso incondicional denegado.

Os dous habitualmente están dentro de seccións <Directory>.

- **<RequireAll>** e **</RequireAll>** tamén se usan para englobar un grupo de directivas de autorización das que ningunha pode fallar e polo menos unha debe de cumprirse para que o grupo se avalíe como exitoso. Se ningunha das directivas dentro de <RequireAll> falla, e polo menos unha se avalía de forma afirmativa, entón o bloque avaliase coma exitoso. Se ningunha se avalía de forma positiva e ningunha falla, o bloque avaliase coma neutral. En todos os demais casos coma non exitoso.
- **<RequireAny>** e **</RequireAny>** empréganse para englobar un grupo de directivas de autorización das que polo menos unha ten que avaliarse de xeito afirmativo para que a directiva <RequireAny> sexa exitosa. Se unha ou máis directivas contidas dentro de <RequireAny> se avalía de xeito afirmativo, entón, a directiva <RequireAny> é exitosa. Se ningunha é exitosa pero ningunha falla, entón devolvese un resultado neutral. Nos outros casos

a avaliación é negativa. Se se poñen varias directivas Require sen ningún tipo de agrupamento, suponse que o agrupamento que as engloba <RequireAny>.

- **<RequireNone>** e **</RequireNone>** úsanse para englobar grupos de directivas de autorización das cales ningunha debe avaliarse de forma positiva para que a directiva <RequireNone> non falle. Se unha ou máis directivas contidas dentro de <RequireNone> ten éxito, a directiva <RequireNone> falla. En todos os demais casos o resultado é neutral. Así do mesmo xeito que a outra directiva para negar autorización Require not, non pode autorizar unha petición de forma independente, xa que nunca devolve un resultado exitoso. Sen embargo, si que se pode empregar para restrinxir un conxunto de usuarios que si terían acceso a un recurso.

Os contedores de autorización <RequireAll>, <RequireAny> e <RequireNone> pódense combinar uns con outros e coa directiva Require para construír unha lóxica de autorización complexa.

2.4. Control de acceso por usuarios

Hai tres tipos de módulos involucrados nos procesos de autenticación e autorización. É necesario escoller polo menos un módulo de cada grupo:

- **Tipos de Autenticación**: mod_auth_basic e mod_auth_digest
- **Provedor de autenticación**: mod_authn_anon, mod_authn_dbd, mod_authn_dbm, mod_authn_file, mod_authnz_ldap e mod_authn_socache
- **Autorización**: mod_authnz_ldap, mod_authz_dbd, mod_authz_dbm, mod_authz_groupfile, mod_authz_host, mod_authz_owne and mod_authz_user

De xeito adicional a estes módulos, tamén están os módulos mod_authn_core e mod_authz_core, que implementan directivas propias comúns a todos os módulos de autenticación. O módulo mod_authz_host ofrece autorización e control de acceso baseado no nome do equipo,

enderezo IP ou características da petición, pero non é parte do sistema de provedores de autenticación.

As directivas tratadas aquí necesitan estar dentro dalgunha sección de configuración do servidor principal, tipicamente nunha sección <Directory> ou nun ficheiro de configuración de directorio .htaccess. Se se pretende empregar ficheiros .htaccess hai que poñer na directiva AllowOverride un valor que permita directivas de autenticación neses ficheiros. Os valores deben ser AuthConfig ou All.

Finalmente, a directiva Require de forma illada ou agrupada con <RequireAll>, <RequireAny> ou <RequireNone> indican a que usuarios se lles permite ver o contido do directorio.

Autenticación Basic

O módulo mod_auth_basic permite o uso da autenticación HTTP Basic para restrinxir o acceso comprobando os usuarios no provedor indicado. Este módulo debe ser combinado cun módulo de autenticación, como pode mod_authn_file e un modulo de autorización como pode ser mod_authz_user. Hai que ter en conta que o contrasinal envíase ao servidor codificado co algoritmo *base64*, o cal ten un algoritmo inverso que permite unha descodificación sinxela.

Para escoller este módulo é necesario poñer o valor Basic na directiva AuthType. Esta directiva establece o nome do dominio (*realm*) de autorización para un directorio. Este nome de dominio amósaselle ao cliente para que saiba que nome de usuario e contrasinal debe introducir. Se este nome de dominio conten espazos debe ser encerrado entre comiñas dobres. A cadea introducida na directiva AuthName coincide coa mensaxe que lle aparece no cadro de diálogo de autenticación en moitos navegadores.

Así, por exemplo, se un usuario se autenticou no nome de dominio "Área restrinxida", automaticamente empregarase o mesmo nome de

usuario e contrasinal sempre que se indique ese mesmo dominio. Así impídese que se lle pregunte o nome de usuario e contrasinal ao usuario máis dunha vez no mesmo dominio. Por suposto, se cambia o nome de equipo do cliente ou o seu enderezo IP, debese introducir de novo a información de login.

Segundo, necesítase un provedor de Autenticación. Se se usa mod_auth_basic como tipo de Autenticación, necesítase poñer na directiva AuthBasicProvider un dos seguintes valores: file para usar o módulo mod_authn_file, dbm para usar o módulo mod_authn_dbm ou dbd para o módulo mod_authn_dbd.

Un dos valores máis típicos é file. Nese caso necesítase un ficheiro de texto cos nomes de usuario e contrasinais que sexa lexible polo demo de execución de Apache2. O ficheiro crease co comando htpasswd que ven de serie coa instalación de Apache. Para establecer cal é ese ficheiro empregase a directiva AuthUserFile. Exemplo:

```
#First time -c option to create the file.
htpasswd -c /usr/local/apache/passwd/passwords rbowen
#Next time we add new users to file.
htpasswd /usr/local/apache/passwd/passwords joesmith
```

Finalmente, a directiva Require soa ou agrupada con <RequireAll>, <RequireAny> ou <RequireNone> indica que usuarios se lles permite ver o contido do directorio. No directorio en cuestión, a configuración pode ser tal que así:

```
AuthType Basic
AuthName "Restricted Files"
AuthBasicProvider file
AuthUserFile "/usr/local/apache/passwd/passwords"
Require user rbowen
```

Autenticación Digest

O módulo mod_auth_digest implementa a Autenticación HTTP co algoritmo *MD5* para codificar passwords, e é unha alternativa a mod_auth_basic no que o contrasinal é transmitido en texto plano. Sen embargo, isto non é ningunha mellora de seguridade sobre a autenticación basic xa que o algoritmo MD5 está roto dende hai anos. Por outro lado, o almacenamento do contrasinal no servidor é menos seguro con autenticación digest que con autenticación basic. Desa maneira é moito mellor alternativa empregar autenticación basic xunto co módulo mod_ssl.

Para escoller este módulo é necesario escoller o valor Digest na directiva AuthType. Do mesmo xeito que mod_auth_basic, a directiva AuthName establece o nome do dominio de autenticación (*realm*) dun directorio. Este nome de dominio é o que se lle da ao cliente para que o usuario saiba que nome de usuario e contrasinal debe poñer. Pero a maiores, tamén se emprega co comando htdigest para crear os contrasinais dos usuarios.

De maneira similar ao módulo mod_auth_basic é necesario un provedor de autenticación. A directiva AuthDigestProvider debe especificar un dos tres seguintes valores file, dbm ou dbd dependendo de que módulo se vai empregar.

Se se escolle o valor file, coa utilidade htdigest pódese crear a lista de usuarios. A forma de usala é similar a htpasswd vista anteriormente, pero a maiores debe ser especificado o nome de dominio no momento da creación, e debe coincidir co introducido na directiva AuthName. Vémolo nun exemplo:

```
#First time -c option to create the file.
htdigest -c /usr/local/apache/passwd/passwords myrealm rbowen
#Next time we add new users to file.
htdigest /usr/local/apache/passwd/passwords myrealm joesmith
```

Finalmente a directiva Require soa ou agrupada con <RequireAll>, <RequireAny> ou <RequireNone> indica que usuarios teñen permiso para ver o contido do directorio. Un exemplo de configuración pode ser este:

```
AuthType Digest
AuthName myrealm
AuthDigestProvider file
AuthUserFile "/usr/local/apache/passwd/passwords"
Require user rbowen
```

Autorización por grupos de usuarios

As directivas anteriores so lle permiten a unha persoa (ou máis de unha se especificamos unha lista de usuarios) acceder ao directorio. En moitos casos, queremos permitir que poida acceder máis dunha persoa. Aquí e onde entra en xogo a directiva AuthGroupFile (teremos que ter activado o módulo authz_groupfile para poder usala). Se queremos que máis dunha persoa poida acceder, necesitarase crear un ficheiro con grupos de usuarios que asocia nomes de grupos con listas de usuarios de cada grupo. O formato dese ficheiro é ben simple, e pódese crear con calquera editor. O contido pode ser coma este:

```
GroupName: rbowen dpitts sungo rshersey
```

Consiste nunha lista de usuarios separada por espazos, e así soamente nos queda modificar os ficheiros .htaccess ou os bloques <Directory> de xeito similar ao seguinte:

```
AuthType Basic
AuthName "By Invitation Only"
AuthBasicProvider file
AuthUserFile "/usr/local/apache/passwd/passwords"
AuthGroupFile "/usr/local/apache/passwd/groups"
Require group GroupName
```

Agora, calquera que apareza no listado GroupName, e apareza no ficheiro de password, poderá acceder se introduce o contrasinal correcto.

Tamén hai outra maneira de permitir que múltiples usuarios poidan acceder, aínda que é menos específica. En vez de crear un grupo de usuarios, podemos usar a seguinte directiva:

```
Require valid-user
```

Usando isto, calquera que apareza no ficheiro de contrasinais e introduza o contrasinal correcto, poderá acceder.

2.5. Ficheiros de log

Para manexar de xeito efectivo un servidor web, é necesario ter feedback sobre a actividade e o rendemento do servidor, así coma dos problemas que poden ocorrer. O servidor HTTP Apache, fornece unha variedade de diferentes mecanismos para rexistrar todo o que ocorre no servidor, desde a petición inicial pasando polo mapeo da URL ata a resolución final da conexión incluíndo os erros que poden ocorrer no proceso. Adicionalmente módulos de terceiros, poden fornecer capacidade de rexistro, ou inxectar entidades nos ficheiros de log, e tamén aplicacións coma programas CGI, scripts PHP ou outros poden mandar mensaxes ao log do servidor.

Log de erros

O rexistro de log de erros do servidor, cuxo nome e localización se establece coa directiva ErrorLog é o ficheiro de log máis importante. Aquí é onde Apache manda información de diagnóstico e rexistra calquera erro que se atope no procesamento das peticións. É o primeiro lugar onde se debe mirar cando se produce un erro co inicio do servidor, ou coa operación do servidor, xa que frecuentemente contén detalles do que está mal e como poder solucionalo.

O log de erros, habitualmente escríbese nun ficheiro (tipicamente *error.log* en sistemas Unix/Linux). En Debian e as súas derivadas o ficheiro máis típico é /var/log/apache2/error.log.

ErrorLog /var/log/apache2/myvirtualhost/error.log

Se a ruta ao ficheiro non é absoluta, entón asúmese que é relativa ao valor da directiva ServerRoot. O formato do log de erros defínese coa directiva ErrorLogFormat, que nos permite personalizar que valores se rexistran. Un formato por defecto defínese se non se establece ningún.

ErrorLogFormat "[%t] [%l] [pid %P] %F: %E: [client %a] %M"

Unha mensaxe típica é esta:

[Fri Sep 09 10:42:29.902022 2011] [core:error] [pid 35708:tid 4328636416] [client 72.15.99.187] File does not exist: /usr/local/apache2/htdocs/favicon.ico

O primeiro elemento na entrada do log é a data e hora da mensaxe. A seguinte, o módulo que produciu a mensaxe (core, neste caso) e o nivel de severidade da mensaxe. A continuación sígueo o ID do proceso, se é aplicable o ID do fío que provocou a condición. Despois temos o enderezo que fixo a petición, e finalmente a mensaxe detallada de erro, que neste case indica que hai un ficheiro que non existe.

Poden aparecer unha gran variedade de mensaxes diferentes no log de erros. Moitos deles son similares ao exemplo anterior. O log de erros tamén pode ter información de depuración de scripts CGI. Calquera información escrita no stderr por un script CGI copiase directamente ao ficheiro de log de erros.

Durante as probas, e común monitorizar de xeito continuo o log de erros para buscar problemas. En sistemas Linux, pode facerse empregando:

tail -f /var/log/apache2/error.log

Logs de accesos

O ficheiro de log de accesos rexistra todas as peticións procesadas polo servidor. A localización e contido do ficheiro de log de acceso, contrólase coa directiva CustomLog. A directiva LogFormat pode ser usada para simplificar a selección de contidos dos logs. Aquí descríbese como como configurar o servidor para gravar información no log de accesos.

A directiva CustomLog, ademais do ficheiro onde se garda a información de log, pode tamén especificar un formato explícito ou un alcume ou *nickname* definido por unha directiva LogFormat anterior. Exemplo:

```
# CustomLog with format nickname
LogFormat "%h %l %u %t \"%r\" %>s %b" common
CustomLog "logs/access_log" common
# CustomLog with explicit format string
CustomLog "logs/access_log" "%h %l %u %t \"%r\" %>s %b"
```

O formato dos logs de acceso é altamente configurable. Especifícase cunha cadea que se parece ao formado expresado na sentencia printf() da linguaxe de programación C.

Poden crearse múltiples logs de acceso simplemente especificando múltiples CustomLog nos ficheiros de configuración. Por exemplo, as seguintes directivas crean tres logs de acceso. O primeiro contén información información básica do formato de logs, mentres que o segundo e terceiro rexistran a páxina desde a que se chegou aquí, e tamén información do navegador do usuario.

```
LogFormat "%h %l %u %t \"%r\" %>s %b" common
CustomLog logs/access_log common
CustomLog logs/referer_log "%{Referer}i -> %U"
CustomLog logs/agent_log "%{User-agent}i"
```

Virtual Hosts

Cando se executa un servidor con varios host virtuais, hai varias opcións para manexar ficheiros de log. Primeiro é posible empregar logs do mesmo xeito que un servidor individual. Simplemente poñendo as directivas de log fora das seccións <VirtualHost> no contexto do servidor principal, é posible facer un rexistro de todas as peticións no mesmo log de acceso. Esta técnica non permite facer unha colección de estatísticas por servidor virtual sinxela.

Se as directivas CustomLog ou ErrorLog se establecen dentro de bloques <VirtualHost> todas as peticións ou erros dese host virtual en particular rexístranse un ficheiro específico dese host virtual. Calquera host virtual que non teña directivas de log, terá sempre o rexistro de peticións do servidor principal. Outra opción é engadir información no host virtual á cadea do formato de logs é posible facer un log de todos os host no mesmo ficheiro de log, e posteriormente dividilos en ficheiros individuais se se precisa. Por exemplo, considerando as seguintes directivas:

```
LogFormat "%v %l %u %t \"%r\" %>s %b" comonvhost
CustomLog logs/access_log comonvhost
```

O %v emprégase para rexistrar o nome do virtual host que se está servindo a unha petición.

2.6. Mensaxes de erro personalizadas

Aínda que o servidor HTTP Apache dispón de respostas xenéricas aos eventos dos códigos de estado HTTP 4xx e 5xx, estas respostas son as veces un pouco concisas, pouco informativas, e que poden intimidar aos usuarios do sitio. Pódense fornecer respostas personalizadas, que son máis amigables, noutro idioma diferente ao inglés ou máis acorde co deseñado sitio web. As respostas personalizadas pódense definir para cada código de estado HTTP

designado coma unha condición de erro, é dicir, para códigos de estado 4xx e 5xx.

Tamén se dispón dun conxunto de valores para que os documentos coas mensaxes de erro poidan ser personalizados baseándose nos valores desas variables, empregando inclusións do lado do servidor (Server Side Includes). Ou tamén podemos ter condicións de erro manexada por un programa cgi ou calquera outro manexador (PHP, mod_perl, etc) que faga uso desas variables.

Os documentos de erros configúranse coa ErrorDocument, que se pode empregar en contexto global, virtualhost, directorio, ou tamén en ficheiros .htaccess se a directiva AllowOverride ten o valor FileInfo.

```
ErrorDocument 500 "Sorry, our script crashed. Oh dear"
ErrorDocument 500 /cgi-bin/crash-recover
ErrorDocument 500 http://error.exemplo.com/server_error.html
ErrorDocument 404 /errors/not_found.html
ErrorDocument 401 /subscription/how_to_subscribe.html
```

A sintaxe da directiva ErrorDocument é:

```
ErrorDocument <3-digit-code> <action>
```

onde se trata a acción coma:

- Unha URL local á que redireccionarse (se a acción comeza con unha "/").
- Unha URL externa á que redireccionarse (se a acción é unha URL válida).
- Texto a amosar (en calquera dos outros casos). O texto debe estar entre comiñas dobres (") se consiste en máis dunha palabra.