El servidor HTTP Apache

El servidor "web" Apache (http://httpd.apache.org) és el més utilitzat en el món, tal com es pot veure a les "Web Server Survey" mensuals disponibles a http://www.netcraft.com (on es pot anar seguint la popularitat dels diferents servidors més importants). A més a més, és lliure, multiplataforma i gratuït.

Aquest document explicarà les característiques, configuracions i usos més habituals d'aquest programa, però si és necessària qualsevol ampliació d'informació, tot el que necessitareu serà consultar la excel.lent documentació oficial, disponible a http://httpd.apache.org/docs (i més concretament, la corresponent a la versió de l'Apache amb la què treballarem: http://httpd.apache.org/docs/2.4).

Instal.lació de l'Apache i posada en marxa (Ubuntu 17.10)

Un cop està tot el sistema actualitzat (`apt update && apt upgrade`) procedirem a instal.lar l'Apache simplement executant: `apt install apache2` .

> *NOTA:Si anéssim a l'apartat de descàrregues de la web de l'Apache veuríem que allà hi ha una versió més moderna que la que es s'instal.la des dels repositoris de l'Ubuntu, però només està disponible en forma de codi font (és a dir, l'hauríem de compilar per tenir l'executable) i el procés, tot i que no és complicat, no ens val la pena ja que els canvis entre la versió més moderna i la que ofereix Ubuntu són mínims*

Un cop instal.lat, el servidor es posarà en marxa automàticament (escoltant al port 80) sense haver de fer res especial. Per comprovar-ho un mateix, es pot obrir un navegador qualsevol i accedir a la direcció http://127.0.0.1 (o bé http://nom , on "nom" és el nom associat a la ip 127.0.0.1 dins del nostre arxiu "/etc/hosts" -normalment aquest nom ja hi és i és "localhost"-) : si tot ha anat bé s'hauria de veure una pàgina web explicativa del programa. Si hom volgués conectar-se al servidor Apache d'un company, llavors al navegador hauria d'escriure http://IP.lan.del.company (o bé http://nom, on "nom" és el nom associat a la IP del servidor del company dins del nostre arxiu "/etc/hosts").

Per iniciar, aturar o reiniciar manualment el servei s'ha de procedir com qualsevol altre servei Systemd del sistema, respectivament: `systemctl start apache2` ; `systemctl stop apache2` ; `systemctl restart apache2` També està la comanda `systemctl status apache2` per veure si el servei està encés o no. Per fer que s'inicïi sempre automàticament en arrencar el sistema, s'ha d'executar `systemctl enable apache2` (i per deshabilitar-lo permanentment -fins que no es torni a habilitar-, `systemctl disable apache2`).

Estructura global de carpetes i fitxers de configuració (Ubuntu 17.10)

Dins de la carpeta "/etc/apache2" es troben la majoria de subcarpetes i fitxers de configuració de l'Apache (que no deixen de ser simples fitxers de text pla amb un determinat contingut). Concretament, els fitxers més importants són:

*"**apache2.conf"** : Fitxer de configuració principal del programa. Quasi tota la configuració general es pot establir dins d'aquest fitxer, encara que es recomana (per simplicitat) utilitzar fitxers separats específics (ubicats a la subcarpeta "conf-available/"). En qualsevol cas, aquest fitxer configura els valors per defecte, els quals la majoria de vegades no caldrà canviar.

*"**ports.conf**" : Fitxer on s'especifica els ports per on escoltaran els diferents "virtual hosts" que haguem definit. Un "virtual host" representa un servidor web que funciona (i està configurat) de forma totalment independent respecte a un altre servidor web -és a dir, un altre "virtual host"-, però que està definit juntament amb la resta de "virtual hosts" dins una única màquina: la que executa el programa Apache (és a dir, l'únic servidor "real", d'aquí el nom). Aquest fitxer és important tenir-lo en compte sobre tot a l'hora de configurar servidors HTTPS (és a dir, servidors HTTP combinats amb la capa de seguretat i encriptació TLS).

*"**envvars**" : Fitxer on es defineixen les variables d'entorn que es necessiten tenir establertes per a què l'Apache pugui funciona correctament. Per exemple, APACHE_RUN_USER i APACHE_RUN_GROUP (que serveixen per establir l'usuari i grup sota els que s'executarà el servidor -i per tant, els que definiran els permisos que té el programa-; ambdues variables valen "www-data", un usuari "virtual" específic que es crea en instal.lar l'Apache) o APACHE_PID_FILE (que serveix per establir la ruta del fitxer temporal que conté el PID del procés Apache en aquest moment; val "/var/run/apache2/apache2.pid") o APACHE_LOG_FILE (que serveix per establir la ruta de la carpeta on s'emmagatzemaran els diferents arxius de registres -els "logs"- del servidor; val "/var/log/apache2"), entre altres. Nosaltres no necessitarem modificar aquest fitxer mai.

I les subcarpetes més importants són:

*"**sites-available/**" : Subcarpeta on s'ubiquen els fitxers de configuració dels diferents "virtual hosts" definits, els quals estableixen el contingut que servit pel "virtual host" respectiu depenent del tipus de peticions rebudes. Per defecte ja hi ha dos "virtual hosts" llestos per fer servir: un de tipus HTTP (arxiu "000-default.conf") i un altre de tipus HTTPS (arxiu "default-ssl.conf").
NOTA: La raó de què el nom del fitxer .conf del "virtual host" HTTP comenci amb "000" és perquè la lectura dels diferents fitxers .conf ubicats en aquesta subcarpeta es realitza en ordre alfabètic: d'aquesta manera, "000-default.conf" sempre serà el primer en ser llegit.

*"**sites-enabled/**" : Subcarpeta que conté enllaços directes apuntant a fitxers ubicats dins de la subcarpeta "sites-available/". La existència d'un enllaç concret implica l'activació del "virtual host" associat (o dit d'una altra manera, per desactivar un "virtual host" és suficient amb esborrar l'enllaç d'aquesta carpeta i per tornar-lo a activar només cal refer l'enllaç amb una simple comanda *ln -s*). Per defecte només el "virtual host" HTTP està activat.

*"**mods-available/**" : Subcarpeta on s'ubiquen els fitxers corresponents als diferents mòduls ("plugins") que el servidor Apache té disponibles per utilitzar. Els mòduls de l'Apache són "trossos" de programa que, en general, augmenten la funcionalitat del programa base (encara que entre sí són força diferents en termes d'objectius i funcionament). Dins d'aquesta carpeta cada mòdul sol tenir associat un arxiu amb extensió .load (de càrrega, imprescindible) i un altre amb extensió .conf (de configuració, opcional), ambdós amb el mateix nom (el del propi mòdul). Els arxius .load contenen sols una línia de text com *LoadModule nomModul_module /usr/lib/apache2/modules/nomFitxer.so* , la qual s'encarrega de carregar efectivament el mòdul real (que és el fitxer binari amb extensió .so -concretament, es tracta d'una llibreria dinàmica- a la memòria RAM de la màquina. Els arxius .conf contenen, tal com ja s'ha dit, la configuració del mòdul pertinent, la qual pot ser molt diferent de mòdul a mòdul.

*"**mods-enabled/**" : Subcarpeta que conté enllaços directes apuntant a fitxers ubicats dins de la subcarpeta "mods-available/". La existència d'un enllaç concret implica l'activació del mòdul associat i, per tant, la possibilitat d'utilitzar la funcionalitat que aquest mòdul aporta (o dit d'una altra manera, per evitar que es pugui fer servir aquesta funcionalitat extra -és a dir, per "desactivar" el mòdul- és suficient amb esborrar l'enllaç d'aquesta carpeta, i per tornar-la a activar només cal refer l'enllaç amb una simple comanda *ln -s*).

*"**conf-available/**" : Subcarpeta on s'ubiquen fitxers amb trossos de configuració (generalment relatius a aspectes molt específics) que són interpretats com si fossin part de l'"apache2.conf". La gràcia està en què si la configuració es troba en algun tros dins d'aquesta carpeta no cal tocar el fitxer de configuració principal.

*"**conf-enabled/**" : Subcarpeta que conté enllaços directes apuntant a fitxers ubicats dins de la subcarpeta "conf-available/". La existència d'un enllaç concret implica l'activació de la configuració associada (o dit d'una altra manera, per no tenir en compte aquest tros de configuració -és a dir, per "desactivar-lo"- és suficient amb esborrar l'enllaç d'aquesta carpeta, i per tornar-lo a tenir en compte només cal refer l'enllaç amb una simple comanda *ln -s*).

Cal tenir present que qualsevol canvi que es guardi en algun dels fitxers i subcarpetes anteriors no serà efectiu fins reiniciar el servidor (o bé, de forma alternativa, executar la comanda `systemclt reload apache2`, la qual obliga al servidor a llegir de nou tots els arxius de configuració sense que calgui reiniciar).

En comptes d'haver d'utilitzar manualment la comanda *ln -s* cada cop que volguem activar o desactivar mòduls i/o "virtual hosts" podem aconseguir el mateix (és a dir, crear o destruir els enllaços directes pertinents) fent servir el següent conjunt de comandes (instal.lades automàticament amb el servidor) molt senzilles:

`a2enmod nomModul` : activa el mòdul indicat
`a2dismod nomModul` : desactiva el mòdul indicat
`a2ensite nomarxiuVH` : activa el "virtual host" indicat
`a2dissite nomarxiuVH` : desactiva el "virtual host indicat

Després d'executar qualsevol d'aquestes comandes, caldrà reiniciar (o "recarregar") el servidor per a què els canvis es tinguin en compte.

A sistemes Fedora/CentOS l'estructura de fitxers i carpetes de configuració és diferent. L'arxiu de configuració principal passa a anomenar-se "httpd.conf" i es troba dins de la carpeta "/etc/httpd/conf". Aquest arxiu inclou el que seria l'"apache2.conf", el "ports.conf" i el "000-default.conf" si es volguessin afegir més "virtual hosts", allà mateix es podria fer). El contingut de la carpeta "/etc/httpd/conf.d" seria equivalent al de "/etc/apache2/conf.d" més els arxius .conf de "/etc/apache2/mods-enabled"; el contingut de la carpeta "/etc/httpd/conf.modules.d" és equivalent al dels arxius .load de "/etc/apache2/mods-enabled". La carpeta "/usr/lib64/httpd/modules" conté els mòduls pròpiament dits. El nom del servei (per aturar-lo, iniciar-lo, etc.) s'anomena `httpd` en comptes d'`apache2`.

Repàs al contingut dels arxius de configuració més importants

El fitxer "ports.conf" conté la directiva:

Listen [direccioIP:]port : Estableix la tarja de xarxa (amb la IP indicada) i port pel qual s'escoltaran peticions. Es poden escriure vàries línies *Listen*, però llavors caldrà tenir en compte la configuració dels diferents "virtual hosts" (veure més avall). Fixeu-vos que també poden aparèixer línies *Listen* dins de condicionals que comproven si un determinat mòdul està activat o no: això permet obrir un port només si el mòdul adequat (normalment relacionat amb les conexions segures TLS) està activat.

Als arxius que hi ha dins de la subcarpeta "conf-available" també podem trobar algunes directives interessants, com ara:

AddDefaultCharset UTF-8 : Estableix la codificació predefinida de les pàgines web a servir a UTF-8 (una evolució ampliada i millorada del vell estàndar ASCII) sobrescrivint llavors la que estigui especificada dins d'elles (amb l'etiqueta *<meta charset...>*, per exemple). Si aquesta línia està comentada, llavors prevaldrà la codificació indicada dins de la pròpia pàgina web.

ErrorDocument nºerror {"text error" | /etc/apache2/fitxer | http://ruta/remota/resultat/dun/script } : Estableix el missatge de text que s'enviarà al client si el servidor detecta un error HTTP indicat. En comptes del missatge de text també es pot establir la ruta d'un fitxer (llavors s'enviaria al client el seu contingut) o la URL d'un script -PHP per exemple- (llavors s'enviaria al client el resultat d'haver executat aquest script). Per exemple, si el nº d'error fos el 404, el document a retornar hauria de ser la pàgina mostrant "pàgina no trobada".

El que ha de quedar molt clar és que hi ha moltes directives que poden aparèixer a diferents arxius: a l'arxiu "apache2.conf", a algun arxiu dins de "conf-available", a algun arxiu de configuració d'un mòdul concret (és a dir, que estigui dins de "mods-available"), a algun arxiu de configuració d'un "virtual host" (és a dir, que estigui dins de "sites-available") o fins i tot dins d'una secció <Directory> ó <File> específica (de seguida les veurem). Depenent del lloc on aparegui, aquesta directiva afectarà a més o menys elements: si

apareix a "apache2.conf", és una directiva que afecta a tots els "virtual hosts"; a l'igual passa si apareix a l'arxiu de configuració d'un mòdul (ja que aquest és carregat per tots els "virtual hosts" sense distinció). En canvi, si la directiva es troba només en un arxiu de configuració d'un "virtual host", només afectarà a ell, i si només es troba dins d'una secció <Directory> només afectarà a aquella carpeta concreta d'aquell "virtual host" (o si està dins de la secció <File>, només afectarà a aquell fitxer en concret).

Si es donés el cas que la mateixa directiva amb es troba en diferents llocs amb valors diferents, sempre guanya l'última (la més concreta); és a dir, que si trobem una directiva a l'"apache2.conf" i després a una secció <Directory>, s'aplica aquesta última. Cal tenir present aquest fet, per exemple, a l'hora d'establir directives de registre (*ErrorLog*, *CustomLog*, ...) , entre altres casos.

Per trobar d'un cop totes les línies que continguin una directiva concreta al llarg dels diferents arxius on poden aparèixer es pot executar la comanda: `grep -ri "nomDirectiva" /etc/apache2/*`

Configuració de "Virtual hosts"

La configuració d'un "virtual host" ha d'estar envoltada de l'etiqueta *<VirtualHost ip:port>* al principi (on "ip:port" indiquen, respectivament la direcció IP concreta per on estarà accessible aquest "virtual host" -o bé el símbol "*" si volem que ho estigui a través de qualsevol IP que tingui el nostre servidor- i el port per on estarà escoltant, el qual ser el nº80) i una etiqueta *</VirtualHost>* al final. Aquesta configuració es pot escriure directament dins de "apache2.conf" o, més habitualment, dins d'un arxiu (anomenat com volguem però amb extensió .conf) dins de la carpeta "/etc/apache2/sites-available".

Dins d'aquestes etiquetes les directives imprescindibles que cal escriure per definir un "virtual host" són:

ServerName nom : Indica el nom DNS del "virtual host". Aquesta dada és la que distingueix un "virtual host" d'un altre que hi estigui definit dins del mateix servidor Apache.

NOTA: Si no disposem d'un servidor DNS a la nostra infrastructura de xarxa (o bé no tenim contractat cap servidor DNS extern que apunti a la IP de la màquina on s'està executant l'Apache) no podrem tenir accessible el nostre servidor Apache a Internet. No obstant, sí que el podrem fer servir dins de la nostra LAN: per això haurem de configurar convenientment l'arxiu "/etc/hosts" a cadascun dels ordinadors (clients i servidors) presents a la nostra xarxa de forma que reconeguin mútuament els seus respectius noms.

ServerAlias unNomDiferent unAltre...: Indica un nom o noms alternatiu/s a l'indicat a *ServerName* . D'aquesta manera, els clients podran accedir al "virtual host" fent servir diferents noms. És molt típic trobar que *ServerName* té el valor *www.domini.com* i *ServerAlias* té el valor *domini.com*, per ex.

DocumentRoot /ruta/carpeta : Indica la ruta de la carpeta arrel des d'on pengen les pàgines web (i la resta de documents) que serveix aquest "virtual host". Al "virtual host" definit per defecte en una instal.lació d'Apache (és a dir, a "000-default.conf" en Ubuntu) aquesta carpeta és "/var/www/html". Això vol dir que la pàgina web que apareix quan accedim a http://127.0.0.1 (o http://ip.dun.altre.servidor) només havent instal.lat l'Apache es troba allà (concretament és la pàgina anomenada "index.html"). El mateix passaria si escrivíssim http://127.0.0.1/index.html (o http://ip.dun.altre.servidor/index.html) ja que "index.html" és el nom de pàgina web a mostrar que Apache pren per defecte si a la petició no s'especifica cap. Si volguéssim publicar qualsevol altre pàgina o fitxer (per exemple, un document anomenat "hola.pdf") només caldria copiar-lo dins d'aquesta carpeta i ja està: anant llavors a http://ip.del.servidor.web/hola.pdf ja el tindríem disponible.

NOTA: Dins de la carpeta arrel es poden crear tantes subcarpetes com es vulgui, les quals també s'hauran d'indicar convenientment a la URL demanada; per exemple, si creem una carpeta "/var/www/html/fotos" i allà copiem l'arxiu "platja1.png", per accedir a aquesta foto via navegador hauríem d'escriure a la seva barra de direccions la següent URL: http://ip.del.servidor.web/fotos/platja1.png". Com es pot comprovar, tot el que queda fora de "/var/www/html" és invisible al visitant: per això s'anomena "carpeta arrel".

NOTA:Per a què tot funcioni correctament, el propietari de la carpeta arrel i el de les seves subcarpetes ha de ser l'usuari propi de l'Apache (www-data) i els seus permisos han de permetre l'accés a tot el món (és a dir, tenir els permisos d'execució -"x"- per l'usuari, el grup i la resta). Els fitxers dins de la carpeta arrel i els que hi hagi dins de les seves subcarpetes, en canvi, el que han de tenir per tothom (usuari, grup, resta) és el permís de lectura -"r"-. Aquestes configuracions ja estan ben establertes per defecte, però, en tot cas, recomano repassar les comandes chmod i chown per si fos necessari el seu ús en algun altre moment.

Altres directives molt comunes definides per cada "virtual host" són les relacionades amb els missatges de registre:

ErrorLog /ruta/error.log : Indica la ruta de l'arxiu que guardarà els missatges dels errors que puguin ocórrer. La configuració per defecte fa servir el fitxer "/var/log/apache2/error.log". També es podria indicar, en comptes de la ruta d'un fitxer, la paraula "journald" (d'aquesta manera, es delega al sistema de logs de Systemd la recopilació i gestió dels missatges d'error) però això només és possible si s'usa el mòdul "mod_journald" , disponible només a partir de la versió 2.5 d'Apache (http://httpd.apache.org/docs/trunk/mod/mod_journald.html)

LogLevel nivell : Indica el nivell mínim d'importància que els errors han de tenir per ser guardats a l'arxiu especificat a *ErrorLog*. Els nivells són (de menys a més importants): "debug", "info", "notice", "warn", "error", "crit", "alert" i "emerg". També es pot especificar un nivell mínim només per errors relacionats amb un determinat mòdul (o més) diferents del nivell mínim per defecte; això es fa indicant el nom del mòdul seguit de dos punts i el nivell, així per exemple: *LogLevel notice rewrite:info* (on s'estableix el nivell "notice" per tots els errors excepte els relacionats amb el mòdul "rewrite" -que ja estudiarem-, els quals han de tenir el nivell "info").

ErrorLogFormat "format" : Aquesta línia defineix quina informació es vol guardar a l'arxiu indicat a *ErrorLog* seguint el format especificat. Aquest format està documentat a http://httpd.apache.org/docs/2.4/mod/core.html#errorlogformat però no se sol utilitzar gaire (de fet, per defecte no apareix escrita); això és perquè l'Apache ja incorpora un format predefinit pels missatges d'error que sol ser adequat a la majoria d'ocasions; concretament, consta dels següents camps delimitats per claudàtors: la data i hora del missatge, el nom del mòdul que produeix el missatge i el seu nivell d'importància (veure *LogLevel*), el PID del procés que ha experimentat l'error, la direcció del client que ha fet la petició i una frase detallada explicant l'error.

I també:

LogFormat "format" nomFormat : Defineix un format i li assigna un nom per poder-lo referenciar més endavant en una línia *CustomLog* (que és la que realment s'encarrega de guardar la informació en un arxiu de registre personalitzat, amb el format referenciat). Aquest format està documentat a http://httpd.apache.org/docs/2.4/mod/mod_log_config.html#formats (i podem trobar exemples a http://httpd.apache.org/docs/2.2/logs.html#accesslog) però tot seguit mostrem els valors comuns:

%a : direcció Ip del client
%h : nom del client
%H : el protocol de la petició (HTTP, HTTPS,...)
%m : el mètode de la petició (GET, POST,...)
%U : la url sencera demanada, sense querystring
%q : la querystring (el que va després del ?)
%r : primera línia de la petició (és a dir, la línia *<METODE><URI><VERSIO>*)
%u : usuari remot (si es requereix protecció per contrasenya; veure més endavant)
%{NomCapçaleraClient}i : valor de la capçalera de client (petició) indicada
%s : estat de la petició
%B : tamany del cos de la resposta. També pot servir %b
%{NomCapçaleraServidor}o : valor de la capçalera de servidor (resposta) indicada
%t : l´hora en la què es va rebre la petició (en format anglès)
%{format}t: l´hora en la què es va rebre la petició (en format strftime: http://strftime.org)
%D : temps trigat en servir la petició, en microsegons
%T : temps trigat en servir la petició, en segons

Un exemple podria ser: *LogFormat "%h %u %t |"%r|" %>s %b %{User-agent}i" unformat* . Poden existir múltiples línies *LogFormat*

NOTA: Els modificadors < i > s'usen en peticions que poden ser diferents entre l'original i la final; per exemple, %>s es pot usar per guardar l'estat final de la petició

CustomLog /ruta/fitxer "format" ó *CustomLog /ruta/fitxer nomFormat* : Registra informació escollida amb un determinat format (definit directament fent servir les mateixes opcions de la llista anterior -%r, %s, ...- o bé indicant el seu nom si ja ha sigut prèviament definit i batejat en alguna líniea *LogFormat* anterior). També es podria indicar, en comptes de la ruta d'un fitxer, la paraula "journald" (delegant, d'aquesta manera, al sistema de logs de Systemd la recopilació i gestió dels missatges). Pot haver-hi múltiples línies *CustomLog* . Per defecte l'Apache té definit una línia *CustomLog* que genera un arxiu anomenat "/var/log/apache2/access.log" amb la informació més típica que podem esperar d'un arxiu de registre de peticions i respostes HTTP; aquest fitxer serà el que més sovint consultem quan volguem saber l'activitat del nostre servidor quina és.

NOTA: Una manera ràpida de veure en temps real els nous missatges que es van emmagatzemant a un determinat arxiu log (això també val per l'arxiu error.log) és executar en un terminal la comanda *tail -f /ruta/fitxer*

HostnameLookups on/off : Indica si es volen guardar als logs els noms dels clients o sols la seva IP

Existeixen dos tipus de "virtual hosts": els basats en IP i els basats en nom. Els primers es distingeixen entre sí per tenir associada una direcció IP i un port d'escolta diferents, de manera que les peticions dels clients són redireccionades al "virtual host" adequat segon la IP i port a la què van dirigides (és a dir, serveixen per a què el servidor Apache amb una sola IP però escoltant en diferents ports o bé amb diferents IPs i/o ports associats cadascun a una tarja de xarxa diferent pugui oferir un lloc web diferent per cada combinació IP/port diferent). En la gran majoria de casos, però, els "virtual hosts" basats en nom són més convenients ja que un "virtual host" d'aquest tipus pot compartir una mateixa direcció IP i port amb altres "virtual hosts" del mateix tipus (és a dir, tots poden funcionar sense trepitjar-se rera el port 80 d'una sola tarja de xarxa) ja que el que els distingeix entre sí és, tal com ja hem comentat, el nom especificat a la directiva *ServerName*, que és el nom que les peticions dels clients hauran d'incloure a la seva capçalera *Host:* per poder ser redirigides al "virtual host" pertinent. Així doncs, quan arriba al servidor Apache la petició d´un document, aquest esbrina per la IP i port de destí (en el cas de "virtual hosts" basats en IP) o pel nom indicat (en el cas de "virtual hosts" basats en nom) quin *DocumentRoot* (entre altres configuracions) ha d'emprar.

La configuració de cadascun dels "virtual hosts" podria escriure's en un fitxer diferent dins de "/etc/apache2/sites-available" o també en un mateix fitxer (activant-lo, en qualsevol cas, amb *a2ensite*), o també dins del propi fitxer "apache2.conf". El més recomanable és desactivar amb *a2dissite* el fitxer "000-default.conf" i utilitzar-lo com a plantilla pel contingut/fitxer propi que creem .A continuació es mostren alguns dels exemples més habituals de configuració de "virtual hosts".

1.-Dos "virtual hosts" basats en nom. És a dir: el servidor Apache (amb una única IP i port) ofereix dos llocs webs, cadascun identificat per un nom DNS ("www.hola.com" i "www.hola.org") convenientment establerts prèviament per a què apuntin,en ambdós casos, a la mateixa màquina (IP):

```
# Això és un comentari: cal assegurar-se que existeixi una línia Listen 80 (a "ports.conf", normalment)
#En ser un "virtual host" basat en nom, l'asteric ("*") indica que tant se val la IP que tingui el servidor
<VirtualHost *:80>
        ServerName www.hola.com
        DocumentRoot "/var/www/html/web1"
        #Altres directives, com ara ErrorLog, CustomLog o seccions <Directory>, <Files>, etc
</VirtualHost>
<VirtualHost *:80>
        ServerName www.hola.org
```

```
    DocumentRoot "/var/www/html/web2"
    #Altres directives, com ara ErrorLog, CustomLog o seccions <Directory>, <Files>, etc
</VirtualHost>
```

2.-Dos "virtual hosts" basats en IPs diferents. És a dir: el servidor Apache disposa de dues targetes de xarxa, cadascuna amb una IP diferent (172.20.30.40 i 172.20.30.50) associades cadascuna a un nom DNS diferent (i oferint un lloc web diferent també)

```
# Cal assegurar-se que existeixi una línia Listen 80 (a "ports.conf", normalment)
<VirtualHost 172.20.30.40:80>
    ServerName www.hola.com
    DocumentRoot "/var/www/html/web1"
    #Altres directives, com ara ErrorLog, CustomLog o seccions <Directory>, <Files>, etc
</VirtualHost>
<VirtualHost 172.20.30.50:80>
    ServerName www.hola.org
    DocumentRoot "/var/www/html/web2"
    #Altres directives, com ara ErrorLog, CustomLog o seccions <Directory>, <Files>, etc
</VirtualHost>
```

3.-Dos "virtual hosts" basats en IPs iguals però ports diferents. És a dir: el servidor Apache disposa d'una tarja de xarxa només (amb una IP associada a un nom DNS) però pot atendre peticions per dos ports diferents, cadascun associat a un "virtual host" (i per tant, un lloc web) també diferent.

```
# Cal assegurar-se que existeixi una línia Listen 80 i una altra Listen 8080 (a "ports.conf", normalment)
<VirtualHost 172.20.30.40:80>
    ServerName www.hola.com
    DocumentRoot "/var/www/html/web1"
    #Altres directives, com ara ErrorLog, CustomLog o seccions <Directory>, <Files>, etc
</VirtualHost>
<VirtualHost 172.20.30.40:8080>
    ServerName www.hola.com
    DocumentRoot "/var/www/html/web2"
    #Altres directives, com ara ErrorLog, CustomLog o seccions <Directory>, <Files>, etc
</VirtualHost>
```

Aquesta configuració es pot combinar perfectament amb la mostrada al punt anterior.

Configuració d'accés a directoris

Dins de la configuració d'un "virtual host" (¡o dins de l'arxiu "apache2.conf" i/o els inclosos dins de la carpeta "conf-available"...afectant llavors a tots els "virtual hosts"!) es pot especificar la manera com volem que el servidor gestioni les diferents carpetes indicades. Això es fa amb diferents directives escrites dins d'una secció (per cada carpeta) que s'obre amb l'etiqueta *<Directory /ruta/carpeta>* i es tanca amb *</Directory>*. Aspectes a tenir en compte:

*Les directives allà indicades s'apliquen a la carpeta especificada (mitjançant la seva ruta absoluta dins del sistema de fitxers) i automàticament a totes les seves subcarpetes.

La ruta indicada pot contenir els comodins "" (equivalent a qualsevol número de caràcters -que no sigui "/"-) i "?" (equivalent a un caràcter qualsevol només)

*Si hi ha vàries seccions <Directory> vinculades a diferents carpetes que estan una dins d'una altra, les directives s'apliquen en l'ordre següent: primer les corresponents a la subcarpeta més interna, i així anar "cap amunt" fins a la carpeta "pare". L'ordre d'aplicació de les directives és molt important perquè sempre -i només- s'aplica l'última de les que hi hagi repetides.

AllowOverride

Una directiva que es pot escriure dins d'una secció <Directory> que cal explicar prèviament és la directiva *AllowOverride.* Aquesta directiva s'utilitza per decidir si dins de la carpeta (i subcarpetes) indicada a la secció <Directory> on està escrita es tindran en compte -i si sí, de quina manera- els "arxius d'accés" (habitualment anomenats ".htaccess"). Un "arxiu 'd'accés" és simplement un arxiu de text ubicat dins d'una carpeta que conté qualsevol de les directives que podriem trobar igualment dins d'una secció <Directory>. D'aquesta manera, si existís (a "apache2.conf", a l'arxiu d'un "virtual host", etc.) una secció <Directory> associada a una determinada carpeta i dins d'aquesta mateixa carpeta es trobés un arxiu d'accés, les directives escrites en aquest sobreescriurien les directives presents a la secció <Directory> corresponent. La utilitat dels arxius d'accés està en donar la possibilitat de poder configurar l'accés a una carpeta determinada per part d'usuaris que no poden modificar els arxius de configuració generals ("apache2.conf", els dels "virtual hosts",etc.); no obstant, si s'utilitzen (en principi estan desactivats via *AllowOverride*, com veurem més endavant) obliguen a l'Apache a buscar-los i llegir-los a cada subcarpeta que hi hagi sota la carpeta arrel, fet que penalitza molt el seu rendiment (per exemple, si un usuari demana la pàgina http://exemple.com/una/ruta/llarga/pagina.html , l'Apache llegirà (per ordre) els eventuals fitxers d'accés presents a "/var/www/html/una", "/var/www/html/una/ruta"i "/var/www/html/una/ruta/llarga", i això amb totes les peticions. Un altre tema a més és l'inconvenient de què les directives que hi hagi escrites puguin entrar en contradicció amb el que mana la configuració general...

A partir d'aquí, els valors més habituals que pot tenir la directiva *AllowOverride* per gestionar l'activació (parcial o total) dels arxius d'accés són:

AllowOverride AuthConfig : Permet que es puguin emprar només les directives *AuthName, AuthType, AuthUserFile i Require* (entre d'altres) a l'eventual arxiu d'accés existent dins de la carpeta associada a la secció <Directory> on es trobi aquesta línia. D'aquestes directives en parlarem en un apartat posterior relacionat amb la protecció de recursos per contrasenya

AllowOverride FileInfo : Permet que es puguin emprar només les directives (moltes d'elles estudiades més endavant) *ErrorDocument, SetHandler, AddHandler, AddType, Action, Header, RequestHeader, SetEnvIf, RewriteEngine, RewriteOptions, RewriteBase, RewriteCond, RewriteRule, Redirect* i família (entre d'altres) a l'eventual arxiu d'accés existent dins de la carpeta associada la secció <Directory> on es trobi aquesta línia

*AllowOverride Indexes :*Permet que es puguin emprar només les directives *AddDescription, AddIcon AddIconByType, AddIconByEncoding, DefaultIcon, DirectoryIndex, FancyIndexing, HeaderName, IndexIgnore, IndexOptions* i *ReadmeName* (totes elles pròpies del mòdul "dir", estudiat més endavant) a l'eventual arxiu d'accés existent dins de la carpeta associada a la secció <Directory> on es trobi aquesta línia.

AllowOverride Options : Permet que es puguin sobreescriure els valors de la directiva *Options* (llegir més avall). Si només es vol que es puguin sobreescriure certs valors d'aquesta directiva, es pot escriure *AllowOverride Options=unvalor,unaltre,... (així: AllowOverride Options=Indexes, Includes* per exemple)

AllowOverride None : No permet que es pugui emprar cap directiva a l'eventual arxiu d'accés existent dins de la carpeta associada a la secció <Directory> on es trobi aquesta línia. En altres paraules, desactiva els fitxers d'accés en aquella carpeta (i subcarpetes).
AllowOverride All : Permet que es puguin emprar totes les directives possibles a l'eventual arxiu d'accés existent dins de la carpeta associada a la secció <Directory> on es trobi aquesta línia. Es desaconsella el seu ús per seguretat.

Es poden combinar varis valors en una mateixa línia, p. ex així: *AllowOverride AuthConfig Indexes*

Options

Una altra directiva bàsica que es també habitual dins d'una secció <Directory> és *Options*, la qual igualment té diferents valors:

Options [+|-] Indexes : Indica que si a la carpeta associada a la secció <Directory> actual no hi ha cap arxiu per defecte (és a dir, un arxiu anomenat "index.html" o, més exactament, algun dels indicats a la directiva *DirectoryIndex* pertanyent a la configuració del mòdul "dir", del qual parlarem més endavant), retorna la llista del contingut d'aquesta carpeta. Útil per mostrar llistats de fitxers (a descarregar) en comptes de pàgines web.

Options [+|-] FollowSymLinks : Es poden seguir els accesos directes ("enllaços simbòlics)

Options [+|-] MultiViews : Permet negociar la mostra del contingut segons la configuració del client (llenguatge, codificació). Es fa servir per saber quin idioma té el client, per exemple.
Options [+|-] ExecCGI : Es poden executar scripts CGI que hi hagin (veure + endavant)
Options [+|-] Includes : Es poden executar scripts SSI que hi hagin dins d'aquesta carpeta
Options [+|-] IncludeNoExec : Es poden executar scripts SSI excepte l´"exec"

Options [+|-] All : Totes les anteriors

Si hi ha un + ó -, s'apliquen les directives a sobre de les que ja s´hagin heretat. Si no hi ha res, es reseteja i es torna a començar. Es poden combinar varis valors en una mateixa línia, així: *Options FollowSymLinks Indexes*, per exemple.

Require

Una altra directiva bàsica que es també habitual dins d'una secció <Directory> és *Require xxx valor ...*, la qual permet l'accés de les peticions de clients a la carpeta associada només si tenen cadascuna un valor determinat de la característica indicada a "xxx". Aquí poden haver molts casos particulars, així que llistarem els més habituals:

Require all granted : Es permet l'accés completament. També existeix *Require all denied*, que prohibeix l'accés completament

Require ip 192.168.1.34 192.168.2 172.20 10 : Es permet l'accés només des dels clients que tinguin (en aquest cas concret) la ip 192.168.1.34 o què pertanyin a la xarxa 192.168.2.0 (classe C), 172.20.0.0 (classe B) o 10.0.0.0 (classe A).També es podria haver escrit *Require ip 192.168.1.34 192.168.2.0/24 172.20.0.0/16 10.0.0.0/8* o també *Require ip 192.168.1.34 192.168.2.0/255.255.255.0 172.20.0.0/255.255.0.0 10.0.0.0/255.0.0.0* : és el mateix. Si volguéssim escriure *Require ip 127.0.0.1* per només donar accés a l'ordinador local a un determinat recurs, una alternativa equivalent és escriure *Require local*

Require host nomClient unAltre ... : Es permet l'accés només des dels clients amb els noms llistats (que poden ser DNS o bé reconeguts via "/etc/hosts").

Require user nomUsuari unAltre ... : Es permet l'accés només als usuaris indicats (que no són usuaris del sistema sinò uns propis de l'Apache...per més informació veure l'apartat posterior que parla sobre la protecció de recursos amb contrasenya). També existeix la directiva *Require group nomGrup unAltre ...*, la qual és similar però per a grups d'usuaris i *Require valid-user*, la qual permet l'accés a tots els usuaris que estiguin definits dins l'Apache sense distinció).
NOTA: Per realitzar el procés d'autenticació correctament les directives *Require user*, *Require group o Require valid-user* han de venir acompanyades d'altres directives auxiliars, concretament: *AuthName*, *AuthType*, *AuthBasicProvider* (o *AuthDigestProvider*), *AuthUserFile i AuthGroupFile*.

Require expr expressió (on expressió pot ser qualsevol de les indicades a la documentació oficial en http://httpd.apache.org/docs/2.4/expr.html) : Es permet l'accés només si l'expressió és certa. A continuació es proposen alguns exemples:

Require expr %{HTTP_USER_AGENT} != 'BadBot' : S'accepten peticions de tots els clients excepte de 'BadBot'

Require expr "%{TIME_HOUR} -ge 9 && %{TIME_HOUR} -le 17" : S'accepten peticions només entre les 9:00h i les 17:00h

Require expr "!(%{QUERY_STRING}=~/xx/) && %{REQUEST_URI} in {'/a, '/b'}" : S'accepten peticions que no continguin dins la seva "querystring" la cadena (o millor dit, l'expressió regular) "xx" i que, a més, demanin un recurs la url del qual no contingui les cadenes "/a" o "/b"

Require method GET POST ... : Es permet l'accés només als metodes HTTP indicats

Require env variableEntorn unaAltraVariable ... : Es permet l'accés només si la/es variable/s d'entorn del sistema operatiu indicada/es està/an definida/es (és a dir, té/tenen valor)

Totes les directives *Require xxx* anteriors admeten la possibilitat de ser negades mitjançant la paraula "not" així *Require xxx not valor*. És a dir, per exemple, per permetre l'accés a qualsevol client que NO tingui la IP 192.168.1.34 hauríem d'escriure *Require ip not 192.168.1.34* .

Si es volen indicar més d'una línia *Require xxx* per una secció <Directory> determinada (o un fitxer d'accés determinat, que ve a ser el mateix), és recomanable encabir-les totes dins d'una secció iniciada per l'etiqueta <RequireAll> i acabada per </RequireAll>. En aquest cas, cal tenir present que l'ordre en què estiguin escrites les diferents línies *Require xxx* dins d'aquesta secció és important perquè en cas de contradicció sempre guanya la darrera escrita; per això, en general, s'acostuma a escriure com a última línea la directiva *Require all denied :* per assegurar-se de què més enllà de les directives anteriors que permeten l'accés no hi ha cap més)

D'altra banda, les directives *Alias i ScriptAlias* també solen acompanyar a seccions <Directory>, però les estudiarem més endavant, en veure el mòdul "alias".

A l'arxiu "apache2.conf" hi ha per defecte un parell de seccions <Directory> interessants, com per exemple la que s'aplica a la carpeta arrel ("/") del sistema de fitxers, la qual és útil per proporcionar una configuració de base per tots els "virtual hosts". Aquí està (com es pot veure, denega l'accés a tot el contingut sota "/" -excepte el que s'especifiqui en carpetes subseqüents- i desactiva els arxius d'accés):

```
<Directory />
        Options FollowSymLinks
        AllowOverride None
        Require all denied
</Directory>
```

Una altra secció <Directory> que es troba definida per defecte a "apache2.conf" és la que correspon a la carpeta "/var/www", contenidora dels documents del "virtual host" definit a "000-default.conf". Aquesta secció pot servir de plantilla quan creem altres "virtual hosts" oferint documents en altres carpetes. Aquí està(es pot veure com aquí la línia *Require* sobrescriu el que s'havia indicat a la secció Directory de la carpeta arrel):

```
<Directory />
        Options Indexes FollowSymLinks
        AllowOverride None
        Require all granted
</Directory>
```

Configuració d'accés a fitxers

Si el que es necessita és controlar fitxers concrets (en comptes de carpetes) es pot usar la secció *<Files nomFitxer>...</Files>*. Cal tenir en compte, però, que en aquesta secció no s'especifica la ruta del fitxer sinò només el seu nom: això vol dir que si s'escrivís dins de "apache2.conf" (o similar), aquesta secció afectaria a tots els arxius amb el nom indicat que existissin a tot el sistema de fitxers del servidor; si s'escrivís dins de l'arxiu de configuració d'un "virtual host", afectaria a tots els arxius amb el nom indicat dins de la seva carpeta arrel (i subcarpetes); si volguéssim, en canvi, restringir la recerca d'aquests fitxers només a una carpeta (i subcarpetes) determinada, hauríem d'incloure la secció <Files> dins de la secció <Directory> adequada.

Més útil que la directiva <Files> és, però, la directiva *<FilesMatch nomFitxer>...</FilesMatch>*. La única diferència està en que a <Files> cal especificar el nom exacte del fitxer en qüestió, però a <FilesMatch es poden utilitzar expressions regulars, de manera que es poden encabir molts fitxers diferents de cop si aquests tenen un nom que segueix un mateix patró.

> NOTA: Igualment, existeix una directiva anomenada <DirectoryMatch> que permet especificar rutes de carpetes fent servir expressions regulars.
> NOTA: Escriure el símbol ~ després de "Files " a la etiqueta <Files> és equivalent a fer servir l'etiqueta <FilesMatch>.De forma similar, escriure'l després de "Directory " a la etiqueta <Directory> és equivalent a fer servir l'etiqueta <DirectoryMatch>. És a dir, <Files ~ "^\.ht"> és equivalent a <FilesMatch "^\.ht">

La directiva més important que pot escriure's dins d'una secció <Files> o <FilesMatch> és la *Require xxx* explicada als paràgrafs anteriors, però aplicada a fitxers en comptes de carpetes. Així, per exemple, trobem que a "apache2.conf" tenim unes línies com...:

```
<FilesMatch “^\.ht”>
        Require all denied
</FilesMatch>
```

...les quals deneguen l'accés (i per tant, la lectura) a tots els arxius del sistema de fitxers del servidor que el seu nom comenci amb ".ht" (el símbol ^ indica "començament i el símbol \ indica que el següent caràcter -el punt- deixa de tenir el significat especial que en una expressió regular habitualment té). És a dir, protegeix el contingut dels possibles arxius ".htaccess" (entre altres) de la vista dels visitants.

Un altre exemple (aquest cas no inclòs per defecte) seria aquest, on es defineix un "virtual host" que no permet l'accés a les carpetes anomenades "privat" que hi hagin al llarg de qualsevol URL (http://www.hola.org/privat, http://www.hola.org/xxx/privat, http://www.hola.org/xxx/yyy/privat , etc (el símbol . indica un caràcter qualsevol i el símbol + indica que aquest caràcter qualsevol es pot repetir una o més vegades, sent en cada repetició un altre caràcter qualsevol):

```
<VirtualHost *:80>
        ServerName www.hola.org
       DocumentRoot "/var/www/html/web"
       <FilesMatch “^.+/privat”>
            Require all denied
       </FilesMatch>
</VirtualHost>
```

Configuració de "locations" i "limits"

A més de les seccions <Directory> i <Files> (que controlen el comportament relacionat amb ubicacions dins del sistema de fitxers del servidor), també existeix la secció *<Location>* -i la seva corresponent *<LocationMatch>*, la qual controla el comportament relacionat amb la part de la URL rebuda del client posterior al nom DNS del servidor, la qual pot no tenir res a veure amb la jerarquia de carpetes reals (veure l'apartat següent que parla dels àlies). És a dir, si l'usuari escriu, per exemple, a la barra del navegador la direcció http://www.exemple.com/webmail/inbox, (i suposant que tenim un "virtual host"

vinculat al nom "www.exemple.com") l'Apache mirarà en la carpeta "/webmail/inbox" sota el DocumentRoot definit, el qual podria ser, per exemple, "/var/www/html/servidorCorreu" (essent, per tant, "/var/www/html/servidorCorreu/webmail/inbox" la carpeta real a accedir). Podríem, per tant, definir una secció en aquest "virtual host" tal com <Location /webmail/inbox> per gestionar aquesta petició vinculada a una carpeta determinada. Un ús comú d'aquesta secció és permetre a un script (PHP, per exemple) gestionar peticions fetes a una determinada URL on l'usuari no té perquè saber quina és la carpeta real associada.

Una altra secció interessant és és *<Limit protocol>...</Limit>* on "protocol" pot ser GET, POST, etc. Aquesta secció serveix per gestionar l'accés al servidor (sobre tot mitjançant directives *Require xxx*) pels protocols HTTP indicats. Similar a <Limit> existeix la directiva <LimitExcept>, que funciona de forma oposada.:

Com a síntesi, la següent llista proporciona una orientació sobre l'orientació que l'Apache utilitza per gestionar les diferents opcions de configuració. Les opcions de més al final sobrescriuren les més al principi:

1.-Es llegeixen les seccions <Directory> presents a l'arxiu "apache2.conf". Si hi ha vàries seccions cadascuna associades a vàries subcarpetes dins del mateix arbre, la primera secció en llegir-se és la que correspon a la ruta més curta i la darrera és la que correspon a la subcarpeta amb la ruta més llarga.

2.-Es llegeixen els fitxers d'accés (".htaccess), sobrescrivint les seccions <Directory> anteriors si això està permés amb l'opció *AllowOverride*

3.-Es llegeixen les seccions <DirectoryMatch> i <Directory ~> presents a l'arxiu "apache2.conf".

4.-Es llegeixen les seccions <Files> i <FilesMatch> presents a l'arxiu "apache2.conf".

5.-Es llegeixen les seccions <Location> i <LocationMatch> presents a l'arxiu "apache2.conf".

6.-Es tornen a repetir els passos anteriors per totes les directives que hi hagi dins tots els fitxers inclosos un rera l'altre via *Include* dins de l'arxiu "apache2.conf" (és a dir, tots els que hi ha dins de "conf-enabled/", "sites-enabled/", "mods-enabled/"...). És molt important, doncs, que l'ordre d'inclusió d'aquests fitxer sigui el correcte, perquè sempre guanya l'últim, recordem.

Ús de diferents mòduls (per defecte activats)

Si no s'indica el contrari, les directives definides per un determinat mòdul es poden deixar escrites dins del seu propi arxiu .conf (i d'aquesta manera, afectarien a tots els "virtual hosts" en conjunt, tal com si s'haguessin escrit directament a l'"apache2.conf" -que és de fet, on van a parar finalment via *Include*-) però també es poden escriure dins de l'arxiu de configuració d'un "virtual host" en particular (dins de la secció <VirtualHost *:80> pertinent), afectant llavors només a aquest "virtual host", evidentment.

*Directives definides al mòdul "dir" (dir.conf)

DirectoryIndex index.html ... : Indica el nom del fitxer -normalment de tipus HTML- que es mostrarà al client si aquest no especifica cap nom en particular quan fa la petició. Per exemple, si un usuari escriu la direcció http://www.exemple.com i la carpeta arrel és "/var/www/html", ¿quina és la pàgina ubicada dins d'aquesta carpeta que s'haurà de mostrar? La que marqui *DirectoryIndex* (per defecte, "index.html"). Així doncs, haver escrit http://www.exemple.com/index.html seria equivalent (però més llarg). Igualment, si l'usuari escriu http://www.exemple.com/unasubcarpeta , ¿quina és la pàgina ubicada dins d'aquesta subcarpeta que s'haurà de mostrar? Doncs la que allà s'anomeni "index.html" també. Si la directiva DirectoryIndex indiqués més d'un nom, l'Apache buscarà a la carpeta pertinent els fitxers per ordre,des del primer nom indicat fins l'últim, mostrant el primer que coincideixi i aturant-se allà. Si no en troba cap, no hi haurà fitxer per defecte i la configuració per defecte de l'Apache el que fa llavors és mostrar un llistat de tots els fitxers ubicats a la carpeta demanada, per a què el client esculli el fitxer concret que vol.

*Directives definides al mòdul "alias" (alias.conf)

Alias /àlies "/ruta/carpeta/real" : Permet que el client pugui accedir al contingut d'una carpeta (!la ruta real de la qual estarà fora del DocumentRoot de qualsevol "virtual host"!) simplement escrivint una paraula ("l'àlies") al final de la URL demanada. Per exemple, si tenim un servidor Apache accesible a través de la URL http://www.exemple.com i volem que un usuari pugui accedir a la carpeta real "/home/pepe" (la qual està fora de la carpeta arrel de qualsevol "virtual host") escrivint a la barra de direccions del navegador una URL tal com http://www.exemple.com/manolo , hauríem d'afegir una línia tal com *Alias /manolo "/home/pepe"* (hauríem a més de vigilar amb els permisos de sistema). Per altra banda, la directiva *Alias* normalment apareix escrita just per sobre d'una secció <Directory> (associada a la carpeta real de la qual són àlies) que conté les directives *Require xxx* pertinents.

També existeixen la directiva *AliasMatch*, que permet l'ús d'expressions regulars per indicar l'àlies i la directiva *ScriptAlias*, que permet definir àlies apuntant a carpetes que contenen fitxers executables (normalment scripts de tipus CGI...en parlarem d'això més endavant).

Redirect /ruta/relativa/desde/DocumentRoot/dela/carpeta/vella http://urlnova : Informa al client que una carpeta determinada ha sigut canviat de lloc, redireccionant-lo al lloc nou. Aquesta directiva fa que l'Apache envïi una resposta amb codi 301 (redirecció temporal); si es vol fer que envïi una resposta 302 (redirecció permanent), es pot escriure la paraula "permanent" entre *Redirect* i la ruta de la carpeta vella. També es pot enviar una resposta amb codi 303 si s'escriu la paraula "seeother" i una resposta amb codi 410 si s'escriu la paraula "gone" (en aquest cas, la URL nova no existeix i per tant, no s'ha d'escriure res al seu lloc) Aquesta directiva es podria escriure per exemple dins d'una secció <Directory> concreta (o en l´arxiu ".htaccess" equivalent) per a què quan s'entŕes en aquella carpeta es fes la redirecció; també es podria utilitzar per redirigir tot el web ("/") a un domini nou si s'escriu dins de la secció <VirtualHost *:80> pertinent.

També existeix la directiva *RedirectMatch* que permet l'ús d'expressions regulars per indicar vàries rutes de carpetes antigues de cop.

*Directives definides al mòdul "autoindex" (autoindex.conf)

Les directives d'aquest mòdul, a més de poder-se definir dins d'una secció <VirtualHost *:80> concreta (afectant a tot aquell "virtual host" determinat) es poden escriure dins de seccions <Directory> o en l'arxiu ".htaccess" equivalent (afectant llavors només a la carpeta associada a aquesta secció). Totes tenen a veure amb la configuració del llistat que apareix en accedir a una carpeta que no té establert fitxer per defecte (és a dir, especificat a *DirectoryIndex*).

HeaderName nomArxiu : Indica el fitxer (ubicat dins de la carpeta indicada a la secció <Directory> on s'hagi definit aquesta línia) que conté el text que sortirà al principi del llistat. Si en comptes d'un nom s'indica una ruta (començant per "/"), s'interpretarà aquesta ruta a partir del *DocumentRoot* del "virtual host" on s'hagi definit aquesta línia.

ReadmeName nomArxiu : Indica el fitxer (ubicat dins de la carpeta indicada a la secció <Directory> on s'hagi definit aquesta línia) que conté el text que sortirà al final del llistat. Si en comptes d'un nom s'indica una ruta (començant per "/"), s'interpretarà aquesta ruta a partir del *DocumentRoot* del "virtual host" on s'hagi definit aquesta línia

IndexIgnore nomArxiu unaltre : Indica els fitxers que no s'han de mostrar quan es fa un llistat (per falta d'arxius per defecte indicats a *DirectoryIndex*). Es poden fer servir comodins (* i ?)

IndexOrderDefault {Ascending|Descending} { Name | Date | Size | Description} : Estableix l'ordre dels elements mostrats al llistat (ascendent -de A a Z, de 0 a 9- o descendent -de Z a A, de 9 a 0-) respecte un de quatre ítems possibles: nom, data d'última modificació, tamany i descripció

IndexOptions [+|-] FoldersFirst IconsAreLinks FancyIndexing SuppressIcon SuppressDescription SuppressSize SuppressLastmodified HTMLTable... : Indica diferents opcions (bastant autoexplicatives) de què i com mostrar el llistat

AddDescription "descripcion" nomArxiu : Indica la descripció a mostrar per un fitxer en concret (en comptes del nom -útil si aquesta línia es troba dins d'una secció <Directory>- es pot escriure una ruta, que serà interpretada llavors com a relativa al *DocumentRoot).* Es pot especificar comodins, així que aquesta directiva també pot servir per indicar la descripció de tipus de fitxers (*.png, *.pdf, etc)

DefaultIcon /ruta/icona : Indica el fitxer utilitzat com a icona (normalment de tipus PNG) per defecte (la ruta és relativa al *DocumentRoot)*

AddIconByType (texteAlt, /ruta/icona) tipusMime unaAltreTipusMime ...: Estableix el fitxer utilitzat com a icona (normalment de tipus PNG) per tots els ítems el tipus MIME del qual sigui un dels indicats. Per exemple: *AddIconByType /icons/imatge.png image/* text/plain* mostraria la mateixa icona per totes les fotografies fossin del format que fossin i pels arxius de text pla. La ruta de la icona és relativa al *DocumentRoot i* el "textAlt" és una frase a mostrar en comptes de la icona si el client no la pot mostrar (perquè no és de tipus gràfic, per exemple). Una altra directiva semblant però que utilitza extensions de fitxers en comptes de tipus MIME per distinguir un ítem d'un altre (i per tant, no és tan fiable) és *AddIcon*

IndexStyleSheet /ruta/full/estill/css : Indica la ruta del fitxer emprat com a full d'estil per mostrar de forma més estètica el llistat. Per saber les classes CSS que es poden utilitzar en aquest full d'estil, consultar http://httpd.apache.org/docs/2.4/mod/mod_autoindex.html#indexstylesheet

Ús de diferents mòduls (per defecte desactivats)

*Directives definides al mòdul "userdir" (userdir.conf)

Si un usuari dins la seva carpeta personal té una carpeta anomenada "public_html", el seu contingut es podrà veure a través de http://ipservidor/~nomusuari, sempre i quan aquest mòdul estigui activat i existeixi la directiva *UserDir public_html*.

Si volguéssim canviar el nom d´aquesta carpeta per a què fos un altre diferent de "public_html" simplement cal substituir la directiva anterior per *UserDir nouNom*. També es podria escriure, en comptes d'un nom nou, la ruta absoluta (des de l´arrel del sistema) d´una carpeta a partir de la qual els usuaris col.locarien els seus continguts (cadascun en una subcarpeta concreta).

En activar el mòdul automàticament tots els usuaris del sistema poden gaudir de la possibilitat de publicar documents a través de la seva carpeta "public_html". Si es vol que algun usuari del sistema no ho pugui fer, caldrà especificar-ho explícitament amb la directiva *UserDir disable usuari1 usuari2 ...*

La secció <Directory> que és present a l'arxiu .conf del mòdul configura per defecte la ruta /home/*/public_html (on * equival al nom de l'usuari la carpeta de la qual es vulgui accedir en aquell moment per aquella petició en concret) amb una sèrie d'opcions *AllowOverride, Options, <Limits>*, etc. que en principi no caldrà modificar.

*Directives definides al mòdul "rewrite" (no té rewrite.conf)

Aquestes directives redireccionen al client, conduïnt-lo a pàgines noves des de links antics. També poden servir per convertir URLs molt llargues i complicades en URLs més senzilles per l'usuari o per denegar l'accés a determinades URLs depenent de segons quines circumstàncies (qui és el client o d'on ve, quina és l'hora, etc).

Aquestes directives s'han d'escriure dins de la secció <Directory> (o en l'arxiu ".htaccess" equivalent) associada a la carpeta corresponent al link antic, i obligatòriament sempre després de la línia *RewriteEngine On* (encarregada de posar en marxa el mòdul "rewrite" per aquella carpeta en concret). Concretament, estem parlant de dues directives fonamentals: *RewriteCond* (opcional) serveix per definir la característica a comprovar en cada petició rebuda (si n'hi ha més d'una, es poden escriure múltiples línies *RewriteCond*, una per cadascuna d'aquestes característiques) i *RewriteRule* serveix per realitzar el redireccionament pròpiament dit, el qual pot efectuar-se sempre o només si la petició concorda amb (totes o, si s'especifica el símbol[OR], amb almenys una de) la/es característica/es comprovada/es prèviament.

Algunes de les possibles característiques el valor de les quals es pot comprobar amb *RewriteCond* són (entre moltes altres):

%{HTTP_REFERER} : Url de la pàgina anterior d'on ve el visitant
%{REMOTE_ADDR} : IP de la màquina client
%{HTTP_USER_AGENT} : Nom del client web (navegador) emprat
%{HTTP_HOST} : IP o nom Dns del servidor
%{SERVER_PORT} : Port del servidor al que es dirigeix la petició del client
%{SCRIPT_FILENAME} : Ruta del fitxer sol·licitat pel client (sense incloure el nom del servidor)
%{QUERY_STRING} : Cua de variables posterior al fitxer sol.licitat (?var1=valor1&var2=valor2...)
%{REQUEST_URI} : Concatenació de SCRIPT_FILENAME i QUERY_STRING
%{TIME_HOUR} : Hora quan és realitzada la petició

Aquestes directives fan un ús extens d'expressions regulars i condicionals, cosa que fa la seva utilització més complicades del que és habitual respecte altres directives d'altres mòduls. Per aprofundir en el seu coneixement, recomano consultar http://httpd.apache.org/docs/2.4/rewrite . També és aconsellable la "xuleta" disponible a http://www.cheatography.com/davechild/cheat-sheets/mod-rewrite o el lloc interactiu de proves https://regex101.com. A continuació presentem, de totes formes, alguns exemples senzills d'ús:

*Exemple que permet escriure "about" al final de la URL en comptes del nom real de l'arxiu sol.licitat ("about.html"). És a dir, permet no haver d'escriure l'extensió (i així fer més còmoda i maca la URL). El símbol ^ indica el començament de la part de la URL que no conté el nom DNS del servidor (és a dir, si la URL completa és http://www.exemple.com/una/url, la part de la URL que processa *RewriteRule* és només "una/url") i el símbol $ indica el final de la URL (sense querystring). Això vol dir que ^about$ representa URLs del tipus http://www.exemple.com/about . El que fa *RewriteRule*, quan detecta una petició amb una URL d'aquest tipus, canviar la paraula "about" per "about.html", de tal manera que la nova URL quedi així: http://www.exemple.com/about.html, que és la direcció real de la pàgina a la què es vol accedir. El símbol [NC] indica que les lletres minúscules i majúscules es consideren iguals. En aquest exemple es pot veure que el patró general d'una directiva RewriteRule és *RewriteRule exprRegEscritaPelClientASubstituir urlRealOnEsVa [opcion1,opcio2]*

```
RewriteEngine on
RewriteRule ^about$ about.html [NC]
```

*Exemple per convertir una URL llarga en una URL curta (per exemple, del tipus http://www.exemple.com/results.php?item=camisa&estacio=estiu en una altra del tipus http://www.exemple.com/camisa/estiu) de manera que l'usuari pugui escriure aquesta darrera per anar a la primera. El símbol "|" significa "o" entre tots els elements que hi ha entre parèntesis. El símbol $1 representa l'element seleccionat d'aquest grup (primer -d'aquí el "1"- i únic grup d'elements -els grups s'identifiquen per la presència de parèntesis- que hi ha).

```
RewriteEngine on
RewriteRule ^camisa/(estiu|hivern|tardor|prima) results.php?item=camisa&estacio=$1
```

Si volguéssim especificar no només camises sinò qualsevol altre tipus d'item, podríem anar un pas més enllà i redactar les següents línies. Entre claudàtors s'indiquen els caracters acceptats per ocupar

la posició d'un caracter però seguidament hi ha el símbol "+" que indica que poden haver un número indeterminat de caracters (d'entre els llistats dins dels claudàtors). L'opció [QSA] serveix, a més, per mantenir a la URL redireccionada la querystring que l'usuari pugui haver escrit a la URL original.

```
RewriteEngine on
RewriteRulev^([A-Za-z0-9]+)/(estiu|hivern|tardor|prima) results.php?item=$1&estacio=$2 [QSA]
```

*Exemple per redireccionar qualsevol pàgina inexistent a la pàgina inicial del lloc, anomenada "home.html" (encara que seria més adient utilitzar la directiva *ErrorDocument 404 /ruta/error.html* per això, però és simplement com a demostració). El símbol "!" significa "no", i el símbol "-f" significa és un fitxer" (altres són, per exemple "-d" per dir "és un directori" o "-l" per dir "és un enllaç", entre d'altres). El símbol "." significa "qualsevol caràcter" i el "*" significa que aquest "qualsevol caràcter" pot aparèixer infinits cops (per tant, una expressió com ^(.*)$ en realitat equival a qualsevol URL.En aquest exemple es pot veure que el patró general d'una directiva RewriteCond és *RewriteCond caracteristica valorAComprovar [opcion1,opcio2]*

```
RewriteEngine on
RewriteCond %{REQUEST_FILENAME} !-f
RewriteRulev ^(.*)$ home.html
```

*Exemple per denegar l'accés a la carpeta des de totes les IPs de clients excepte la 12.34.56.78. L'opció [F] és la que prohibeix l'accés (generant una resposta 403) i l'opció [L] indica que no es continuarà llegint més regles Rewrite (és la darrera, per tant).

```
RewriteEngine on
RewriteCond %{REMOTE_ADDR} !^(12\.34\.56\.78)$
RewriteRule (.*) - [F,L]
```

Com a exercici, aquí es deixen més exemples de regles per a què el lector endevini les conseqüència d'executar-les:

```
RewriteEngine on
RewriteCond %{SERVER_PORT} !^443$
RewriteRule ^(.*)$ https//www.exemple.com/$1 [R,L]
```

```
RewriteEngine On
RewriteCond "%{TIME_HOUR}" ">=20" [OR]
RewriteCond "%{TIME_HOUR}" "<07"
RewriteRule "^nevera" - [F]
```

```
RewriteEngine on
RewriteRule ^pomes\.html$ peres.html
```

```
RewriteEngine on
RewriteRule ^post-id/([0-9]+)$ /posts/$1.html
```

```
RewriteEngine on
RewriteCond %{HTTP_HOST} ^exemple\.com$
RewriteRule ^(.*)$ http://www.exemple.com/$1 [R=301]
```

```
RewriteEngine on
RewriteCond %{HTTP_REFERER} !^$
RewriteCond %{HTTP_REFERER} !exemple\.com [NC]
RewriteRule \.(gif|jpg|png)$ noRobis.jpg [NC,L]
```

És interessant saber que es pot definir un arxiu de logs només pels redireccionaments amb la directiva *ReWriteLog /ruta/fitxer/log/modul/rewrite* i la seva verbositat amb *ReWriteLogLevel nº* (0=res... 9=molt)

*Directives definides al mòdul "proxy" (proxy.conf)

Aquest mòdul serveix per a què l'Apache actuï com un servidor proxy HTTP (a l'estil de programes com Squid). És a dir, per a què actuï com un simple intermediari entre el client i un altre servidor web remot. La utilitat d'aquesta mòdul es basa en la possibilitat que ens dóna de balancejar la càrrega de peticions entre diferents servidors web (entenent aquests tant com màquines com programes), repartint la feina a cadascun segons les seves possibilitats i capacitats.

Per fer-lo servir, no és necessari modificar res del seu arxiu de configuració: només cal activar-lo amb *a2enmod* i escriure la directiva *Proxypass / http://ip.del.altre.servidor:port* dins de l'"apache2.conf" (o dins de la configuració del "virtual host" que volguem utilitzar com a "trampolí" a un altre servidor web). El símbol "/" rera la paraula "Proxypass" representa el *DocumentRoot*, però es podria escriure la ruta de qualsevol subcarpeta que pengés d'allí: d'aquesta manera, només es reenviaria al servidor web extern les peticions a aquella subcarpeta.

Existeixen mòduls de tipus proxy més sofisticats (és a dir, especialitzats en tasques més específiques) que no estudiarem però convé saber que existeixen, com ara el "proxy_balancer" o el "proxy_connect", entre molts d'altres. Per obtenir la llista sencera, aneu a http://httpd.apache.org/docs/2.4/mod.

TRUC: Ús de l'arxiu ".htaccess" per protegir carpetes amb contrasenya

S'ha de tenir el mòdul "auth_basic" (ó "auth_digest" segons el cas) activat. Existeixen altres mòduls (del tipus "auth_*", "authn_*" i/o "authz_*") que són més sofisticats i que poden fer-se servir pel mateix objectiu , però no els veurem en ser més farragosos de configurar. Els passos a seguir són els següents:

1.-Canviar la configuració original de l'etiqueta <Directory /> corresponent a la carpeta que es vol protegir per a què posi *AllowOverride AuthConfig*.

2.-Generar un fitxer -ocult- que contingui la contrasenya de l´usuari (que pot no ser del sistema!) al que volem permetre l´accés. Això es fa amb la comanda `htpasswd -c /ruta/.fitxer nomusuariinventat` (preguntarà la contrasenya que volem). La ruta del fitxer creat NO ha de poder ser accesible a través de cap client web, lògicament. El paràmetre -c és per crear l´arxiu: si no el posem afegirem un nou usuari a l´arxiu existent.

NOTA: Altres paràmetres que poden ser interessants de la comanda htpasswd són -b (per poder introduir la contrasenya com a últim valor de la comanda en comptes d'interactivament), -s (per emmagatzemar la contrasenya encriptada mitjançant l'algoritme SHA en comptes de l'usat per defecte -el crypt(), que és menys segur-), -m (per emmagatzemar la contrasenya encriptada mitjançant l'algoritme MD5 en comptes de l'usat per defecte -el crypt(), que és menys segur-) o -n (per no guardar el resultat en el fitxer sinò treure'l per pantalla)

NOTA: Aquesta comanda ve dins del paquet "apache2-utils", el qual en instal.lar el servidor Apache s'haurà instal.lat com a dependència.

3.-Generar l'arxiu d'accés (".htaccess") dins la carpeta que es vol restringir amb aquest contingut:

AuthType Basic
AuthUserFile /ruta/arxiu/creat/pas/anterior
AuthName "Missatge a mostrar"
Require valid-user #Permet l'accés a tots els usuaris vàlids. Per usuaris concrets: *Require user usu1 usu2*

Aquestes línies es poden escriure perfectament dins d'una secció <Directory /ruta/carpeta/a/protegir> pertanyent al "virtual host" adient. L'elecció de fer servir un arxiu ".htaccess" ve motivada per la circumstància (bastant habitual en serveis de "hosting") de no poder modificar directament els arxius de configuració de l'Apache.

4.-La diferència entre el mètode Basic i el Digest és que el primer utilitza un arxiu de contrasenyes (amb format user:password) les quals, encara que dins el fitxer s'emmagatzemen encriptades es transmeten de client a servidor en text pla, però el segon transmet les contrasenyes en MD5, fet que aporta (una miiiica) més de seguretat. Si fem servir, de totes maneres, el mètode Digest, cal tenir en compte dues diferències:

*Per crear el fitxer de contrasenyes s'ha d'emprar la comanda `htdigest` en comptes de `htpasswd`. El seu funcionament és idèntic excepte en què ara a més s'ha d'indicar com a paràmetre el valor d'AuthName, així: `htdigest -c /ruta/.fitxer "Missatge a mostrar" nomusuariinventat`

*Les directives a incloure dins de l'arxiu ".htaccess" (o a la secció <Directory> adient) seran:
AuthType Digest
AuthDigestFile /ruta/arxiu/creat/pas/anterior
AuthName “Missatge a mostrar”
Require valid-user

Si es troben diferents arxius ".htaccess" al llarg de l´arbre de directoris, s´aniran tenint en compte a mesura que ens anem endinsant: per tant, tindrà preferència l´últim arxiu ".htaccess" que es trobi.

Tal com ja s'ha dit, existeixen altres mètodes d'autenticació,com ara fent que l'Apache consulti directament els usuaris en una base de dades ("auth_pgsql") o en un servidor Ldap ("auth_ldap"), entre d'altres, però això no ho veurem.