Curso de Proxy en GNU/Linux (20 horas)

Teoría y prácticas

Creative Commons

Reconocimiento-No comercial-Compartir bajo la misma licencia 3.0

Usted es libre de:

- copiar, distribuir y reproducir públicamente la obra
- hacer obras derivadas

Bajo las siguientes condiciones:

- **Reconocimiento**. Debe reconocer los créditos de la obra de la manera especificada por el autor o el licenciante (pero no de una manera que sugiera que tiene su apoyo o apoyan el uso que hace de su obra).
- **No comercial**. No puede utilizar esta obra para fines comerciales.
- **Compartir bajo la misma licencia**. Si altera o transforma esta obra, o genera una obra derivada, sólo puede distribuir la obra generada bajo una licencia idéntica a ésta.
- Al reutilizar o distribuir la obra, tiene que dejar bien claro los términos de la licencia de esta obra.
- Alguna de estas condiciones puede no aplicarse si se obtiene el permiso del titular de los derechos de autor
- Nada en esta licencia menoscaba o restringe los derechos morales del autor.

Los derechos derivados de usos legítimos u otras limitaciones reconocidas por ley no se ven afectados por lo anterior.

Esto es un resumen fácilmente legible del texto legal de versión original en Idioma Inglés (la licencia completa)

http://creativecommons.org/licenses/by-nc-sa/3.0/ec/legalcode

Índice de contenido

Introducción

Un servidor proxy-cache tiene básicamente dos funciones, como proxy actúa como intermediario en una transacción Web, el acepta la petición del cliente, la procesa y luego reenvía la solicitud al servidor original, como servidor de cache almacena el contenido de las solicitudes de los clientes para su posible uso de nuevo, cuando existe otra solicitud del mismo contenido el servidor devuelve el contenido del cache, sin tener que contactar al servidor de nuevo.

Un servidor proxy-cache puede ser utilizada por las siguientes razones :

- Usar menos ancho de banda de la conexión a Internet.
- Reducir el tiempo de carga de las paginas Web.
- Proteger los equipos de la red interna.
- Prever que los usuarios acceden a sitios prohibidos.
- Generar estadísticas del trafico de la red y por usuario.
- Asegurar que solo usuarios autorizados pueden conectarse a Internet.

Squid

Squid es un programa de software libre que implementa un servidor proxy y un demonio para caché de páginas web, publicado bajo licencia GPL. Tiene una amplia variedad de utilidades, desde acelerar un Servidor Web, guardando en caché peticiones repetidas a DNS y otras búsquedas para un grupo de usuarios que comparte recursos de la red, hasta caché de web, además de añadir seguridad filtrando el tráfico. Está especialmente diseñado para ejecutarse bajo entornos tipo Unix.

Características de Squid

Squid proporciona un servicio de Proxy que soporta peticiones HTTP, HTTPS y FTP a equipos que necesitan acceder a Internet y a su vez provee la funcionalidad de caché especializado en el cual almacena de forma local las páginas consultadas recientemente por los usuarios. De esta forma,

incrementa la rapidez de acceso a los servidores de información Web y FTP que se encuentra fuera de la red interna.

Proxy para SSL

Squid también es compatible con SSL (Secure Socket Layer) con lo que también acelera las transacciones cifradas, y es capaz de ser configurado con amplios controles de acceso sobre las peticiones de usuarios.

Jerarquías de caché.

Squid puede formar parte de una jerarquía de caches. Diversos proxys trabajan conjuntamente sirviendo las peticiones de las páginas. Un navegador solicita siempre las páginas a un sólo proxy, si este no tiene la página en la caché hace peticiones a sus hermanos, que si tampoco las tienen las hacen a su/s padre/s. Estas peticiones se pueden hacer mediante dos protocolos: HTTP e ICMP.

ICP, HTCP, CARP, caché digests

Squid sigue los protocolos ICP, HTCP, CARP y caché digests que tienen como objetivo permitir a un proxy "preguntarle" a otros proxys caché si poseen almacenado un recurso determinado.

Caché transparente

Squid puede ser configurado para ser usado como proxy transparente de manera que las conexiones son enrutadas dentro del proxy sin configuración por parte del cliente, y habitualmente sin que el propio cliente conozca de su existencia. De modo predefinido Squid utiliza el puerto 3128 para atender peticiones, sin embargo se puede especificar que lo haga en cualquier otro puerto disponible o bien que lo haga en varios puertos disponibles a la vez.

WCCP

A partir de la versión 2.3 Squid implementa WCCP (Web Cache Control Protocol). Permite interceptar y redirigir el trafico que recibe un router hacia uno o más proxys caché, haciendo control de la conectividad de los mismos. Además permite que uno de los proxys caché designado pueda determinar

como distribuir el tráfico redirigido a lo largo de todo el arreglo de proxys caché.

Control de acceso

Ofrece la posibilidad de establecer reglas de control de acceso. Esto permite establecer políticas de acceso en forma centralizada, simplificando la administración de una red.

Aceleración de servidores HTTP

Cuando un usuario hace petición hacia un objeto en Internet, este es almacenado en el caché, si otro usuario hace petición hacia el mismo objeto, y este no ha sufrido modificación alguna desde que lo accedió el usuario anterior, Squid mostrará el que ya se encuentra en el caché en lugar de volver a descargarlo desde Internet. Esta función permite navegar rápidamente cuando los objetos ya están en el caché y además optimiza enormemente la utilización del ancho de banda.

SNMP

Squid permite activar el protocolo SNMP, este proporciona un método simple de administración de red, que permite supervisar, analizar y comunicar información de estado entre una gran variedad de máquinas, pudiendo detectar problemas y proporcionar mensajes de estados.

Caché de resolución DNS.

Squid está compuesto también por el programa dnsserver, que se encarga de la búsqueda de nombres de dominio. Cuando Squid se ejecuta, produce un número configurable de procesos dnsserver, y cada uno de ellos realiza su propia búsqueda en DNS. De este modo, se reduce la cantidad de tiempo que la caché debe esperar a estas búsquedas DNS.

Proxy-Cache en Squid.

El proxy caché es una manera de guardar los objetos solicitados de Internet (por ejemplo, datos como páginas web) disponibles vía protocolos HTTP, FTP y en un sistema más cercano al lugar donde se piden. Los

navegadores web pueden usar la caché local Squid como un servidor proxy HTTP, reduciendo el tiempo de acceso así como el consumo de ancho de banda. Esto es muchas veces útil para los proveedores de servicios de Internet para incrementar la velocidad de sus consumidores y para las redes de área local que comparten la conexión a Internet.

Debido a que también es un proxy (es decir, se comporta como un cliente en lugar del cliente real), puede proporcionar un cierto grado de anonimato y seguridad. Sin embargo, también puede introducir problemas significativos de privacidad ya que puede registrar mucha información, incluyendo las URL solicitadas junto con otra información adicional como la fecha de la petición, versión del navegador y del sistema operativo.

Un programa cliente (por ejemplo, un navegador) o bien tiene que especificar explícitamente el servidor proxy que quiere utilizar (típico para consumidores de ISP) o bien podría estar usando un proxy sin ninguna configuración extra. A este hecho se le denomina caché transparente, en el cual todas las peticiones HTTP son interceptadas por squid y todas las respuestas guardadas en caché. Esto último es típico en redes corporativas dentro de una red de acceso local y normalmente incluye los problemas de privacidad mencionados previamente.

Squid tiene algunas características que pueden facilitar establecer conexiones anónimas. Características tales como eliminar o modificar campos determinados de la cabecera de peticiones HTTP de los clientes. Esta política de eliminación y alteración de cabeceras se establece en la configuración de Squid. El usuario que solicita páginas a través de una red que utiliza Squid de forma transparente, normalmente no es consciente de este proceso o del registro de información relacionada con el proceso.

Requerimientos de Hardware y Sistemas Operativos

Los requerimientos de Hardware de Squid son generalmente modestos, el parámetro mas importante a ser tomado en cuanta en una instalación de squid es la memoria. El no poseer memoria suficiente causa una drástica degradación en el rendimiento de los sistemas. Otro aspecto importante es el espacio en disco, mas espacio significa mayor cantidad de objetos en cache y menores búsquedas en Internet. Una interfaz de discos mas rápida es siempre recomendada, por ejemplo siempre prefiera a discos SCSI a discos ata o sata. El procesamiento no es una variable critica en el rendimiento de squid, aunque CPU mas rápidos siempre son preferidos.

Como squid utiliza una pequeña cantidad de memoria por cada respuesta de cache existe una relación entre el espacio de la memoria y el cache de disco, como regla general se puede utilizar que se necesitan 32 MB de memoria RAM por cada GB de cache de disco, por lo cual un sistema con 1 GB de memoria puede almacenar 32 GB de cache sin afectar su rendimiento.

Squid puede ser instalado en sistemas Linux, Unix, y Windows.

Instalación de Squid en Debian GNU/Linux

Para instalar en servidor Squid en Debian GNU/Linux se debe ejecutar desde una consola como usuario root :

#aptitude install squid3

Configuración de Squid

El archivo de configuración general de squid se llama squid3.conf y se encuentra ubicado en */etc/squid3/squid.conf.*

La configuración de Squid es similar a la de los archivos Unix, cada linea de configuración comienza con una directiva seguida de un valor o valores, squid ignora las lineas en blanco o las que comienzan con el símbolo numeral “#”.

Ejemplos de Configuración en squid.

cache_log /squid/var/cache.log
 # define the localhost ACL
acl Localhost src 127.0.0.1/32
connect_timeout 2 minutes
log_fqdn on

Algunas Directivas poseen un solo valor, por lo cual lineas repetidas harán que se tomen solamente la ultima, en el siguiente ejemplo la segunda linea es la que es tomada como valor del servidor:
connect timeout 2 minutes
connect timeout 1 hour

Controles de Acceso (ACL)

Los controles de acceso son la parte mas importante en la configuración del servidor squid, estas se utilizan para dar acceso a los usuarios y también para negarlo. Las ACL pueden usarse para restringir o prevenir acceso a ciertos sitios o contenidos. Utilizan la siguiente sintaxis:

acl nombre_acl tipo_acl descripción ...
acl nombre_acl tipo_acl "archivo_de_descripciones" ...

Cuando usamos un "archivo_de_descripciones", cada descripción se corresponde con una línea del archivo.

Tipos de ACL

src

Especifica una dirección origen de una conexión en formato IP/máscara.
Por ejemplo, utilizaremos una acl de tipo src para especificar la red local:

acl red_local src 192.168.1.0/24

También podemos especificar rangos de direcciones mediante una acl de tipo src:

acl jefes src 192.168.1.10-192.168.1.25/32

dst

Especifica una dirección destino de una conexión en formato IP/máscara.

acl google dst 216.239.0.0/24

También podemos especificar hosts concretos mediante una acl de tipo dst:
acl google dst 216.239.59.104/32 216.239.39.104/32 216.239.57.104/32

Las definiciones son idénticas a las acl de tipo src salvo que se aplican al destino de las conexiones, no al origen.

srcdomain y dstdomain

Estos tipos de acl especifican un nombre de dominio.
En el caso de srcdomain es el dominio origen y se determina por resolución DNS inversa de la IP de la máquina, es decir, tendremos que tener bien configurado el DNS de la red local.
En el caso de dstdomain el nombre del dominio se comprueba con el dominio que se haya especificado en la petición de página web.
Por ejemplo:

acl google_com dstdomain google.com

srcdom_regex y dstdom_regex

Especifican una expresión regular que verifican los dominio origen o destino. La expresión regular hace distinción entre mayúsculas y minúsculas salvo que incluyamos la opción "-i" que evita dicha distinción.
Ejemplo:

*acl google_todos dstdom_regex -i google\..**

Observamos como al incluir "-i" estamos indicando que no haga distinción entre mayúsculas y minúsculas.

time

Este tipo de acl permite especificar una franja horaria concreta dentro de una semana. La sintaxis es la siguientes

acl nombre_acl_horaria time [días-abrev] [h1:m1-h2:m2]

Donde la abreviatura del día es:
S - Sunday (domingo)
M - Monday (lunes)
T - Tuesday (martes)
W - Wednesday (miércoles)
H - Thursday (jueves)
F - Friday (viernes)
A - Saturday (sábado)

además la primera hora especificada debe ser menor que la segunda, es decir h1:m1 tiene que ser menor que h2:m2

Ejemplo
acl horario_laboral time M T W H F 8:00-17:00

Estaríamos especificando un horario de 8 a 17 y de lunes a viernes.

url_regex

Permite especificar expresiones regulares para comprobar una url completa, desde el http:// inicial.
Por ejemplo, vamos a establecer una acl que se verifique con todos los servidores cuyo nombre sea adserver:

*url_regex serv_publicidad ^http://adserver.**

En otro ejemplo podemos ver una acl que verifique las peticiones de archivos mp3:

url_regex archivos_mp3 -i mp3$

referer_regex

Define una acl que se comprueba con el enlace que se ha pulsado para acceder a una determinada página. Cada petición de una página web incluye la dirección donde se ha pulsado para acceder. Si escribimos la dirección en el navegador entonces estaremos haciendo una petición directa.

Por ejemplo vamos a establecer una acl para todas las páginas a las que hayamos accedido pulsando en una ventana de búsqueda de google:

*acl pincha_google referer_regex http://www.google.**

req_mime

Las acl de tipo req_mime se utilizan para comprobar el tipo de petición mime que realiza un cliente, y se puede utilizar para detectar ciertas descargas de archivos o ciertas peticiones en túneles HTTP.
Esta acl sólo comprueba las peticiones que realiza el cliente, no comprueba la respuesta del servidor. Esto es importante para tener claro qué estamos haciendo y qué no.

Ejemplo

acl subida req_mime_type -i ^multipart/form-data$
acl javascript req_mime_type -i ^application/x-javascript$
acl estilos req_mime_type -i ^text/css$
acl audiompeg req_mime_type -i ^audio/mpeg$

rep_mime_type

Este tipo de acl se utiliza para verificar el tipo de respuesta recibida por el proxy. Este tipo de acl, analiza una respuesta del servidor por lo que sólo le afectas las reglas de respuesta como http_reply_access y no las reglas http_access que se aplican a las peticiones.

Ejemplo:

acl javascript rep_mime_type -i ^application/x-javascript$

acl ejecutables rep_mime_type -i ^application/octet-stream$

acl audiompeg rep_mime_type -i ^audio/mpeg$

Ejercicios con ACL

Ejercicio 1

Crear un archivo de configuración para denegar el acceso a todos los equipos a la dirección www.google.com

acl all src 0.0.0.0/0.0.0.0

acl no_permitido1 dstdomain www.google.com

acl localhost src 127.0.0.1

http_access deny no_permitido1 !localhost

En este ejemplo construimos la acl llamada localhost que representa el servidor proxy (dirección loopback). Observe que en http_access se ha puesto !localhost para denegar el acceso a todos los equipos de la red menos al equipo local. De esta manera el equipo en el que está instalado squid puede acceder a la página www.google.com

Ejercicio 2

Crear un archivo de configuración que deniegue el acceso a las direcciones www.google.com y www.hotmail.com.

acl all src 0.0.0.0/0.0.0.0

acl localhost localhost src 127.0.0.1

acl no_permitido1 dstdomain www.google.com www.hotmail.com

http_access deny no_permitido1 !localhost

En este caso el archivo de configuración es similar al caso anterior. Tan sólo

hay que poner www.hotmail.com después de www.google.com en la acl no_permitido1.

Ejercicio 3

Crear un archivo llamado no_permitidos, en la ruta /etc/squid3 que contenga las direcciones de los tres siguientes dominios:

http://www.google.com
http://www.yahoo.com/
http://www.hotmail.com/

A continuación crea un archivo de configuración squid.conf que deniegue las conexiones a las direcciones que se encuentran en el archivo no_permitidos.

acl all src 0.0.0.0/0.0.0.0
acl localhost src 127.0.0.1
acl no_permitido1 url_regex "/etc/squid3/no_permitidos"
http_access deny no_permitido1 !localhost

Ejercicio 4

Se dispone de una red local con dirección 192.168.1.0 y máscara 255.255.255.0. Crear un archivo de configuración squid.conf que permita el acceso a Squid a todos los equipos de la red y no lo permita a los restantes.

acl all src 0.0.0.0/0.0.0.0
acl todalared src 192.168.1.0/255.255.255.0
acl localhost src 127.0.0.1
http_access allow todalared
http_access deny all !localhost

Ejercicio 5

Se dispone de una red de área local con dirección 192.168.1.0 y máscara 255.255.255.0. Se desea permitir el acceso a Squid a los equipos con las IP que están comprendidas en el rango 192.168.1.1 y 192.168.1.10 (ambas incluidas). Crea en el directorio /etc/squid3/ un archivo llamado ip_permitidas que tenga estas direcciones (cada dirección en una línea diferente). Con está configuración para Squid se permite el acceso a Squid a todas estas direcciones y denegar el acceso a las restantes.

acl all src 0.0.0.0/0.0.0.0
acl red_local src "/etc/squid3/ip_permitidas"
acl localhost src 127.0.0.1
http_access allow red_local
http_access deny all !localhost

Ejercicio 6

Impide la conexión a Internet a todos los equipos en horario de 18:00 a 21:00 horas.

acl all src 0.0.0.0/0.0.0.0
acl localhost src 127.0.0.1
acl horario time 18:00-21:00
http_access deny horario !localhost

Ejercicio 7

Niega las conexiones a todos los equipos en horario de 18:00 a 21:00 horas, pero sólo los lunes, martes y miércoles.

acl all src 0.0.0.0/0.0.0.0
acl horario time MTW 18:00-21:00
http_access deny horario !localhost

Ejercicio 8

Deniega el acceso a Squid al equipo con IP 192.168.1.5. Permite el resto de accesos a Squid.

```
acl all src 0.0.0.0/0.0.0.0
acl equipo5 src 192.168.1.5
http_access deny equipo5
```

Ejercicio 9

Deniega el acceso a Squid al equipo con IP 192.168.1.5 en horario de 18:00 a 21:00 horas. Permite el resto de accesos a Squid.

```
acl all src 0.0.0.0/0.0.0.0
acl equipo5 src 192.168.1.5
acl horario 18:00-21:00
http_access deny equipo5 horario
```

Ejercicio10

Niega el acceso a Squid al equipo con IP 192.168.1.5 en horario de 18:00 a 21:00 horas. Para el resto de equipos permitir el acceso sólo en horario de 10:00 a 14:00 horas. Se supone que los equipos pertenecen a la red 192.168.1.0 con máscara 255.255.255.0.

```
acl all src 0.0.0.0/0.0.0.0
acl red_local src 192.168.1.0/255.255.255.0.
acl equipo5 src 192.168.1.5
acl horario1 18:00-21:00
acl horario2 10:00-14:00
http_access deny equipo5 horario1
http_access allow red_local horario2
http_access allow equipo5
```

Ejercicio 11

En el archivo /etc/squid3/permitidos se tiene una lista de todas las direcciones IP de la red local. El equipo10 tiene la dirección IP 192.168.1.10. Se permite el acceso a Internet al equipo10 de lunes a miércoles de 9:00 a 14:00 horas. También se permite el acceso a los equipos de la red local de lunes a miércoles. Se prohibe el acceso en el resto de casos.

acl all src 0.0.0.0/0.0.0.0
acl localhost src 127.0.0.1
acl redlocal src "/etc/squid3/permitidos"
acl equipo10 src 192.168.1.10
acl horario time MTWHF 9:00-14:00
acl horario2 time MTW
http_access allow equipo10 horario
http_access allow redlocal horario2
http_access allow localhost
http_access deny all

Ejercicio 12

Restringe el acceso a todo el contenido con extensión .mp3 a los equipos de la red.

acl all src 0.0.0.0/0.0.0.0
acl redlocal src 192.168.1.0/255.255.255.0
acl musica urlpath_regex \.mp3
http_access allow redlocal !musica
http_access deny all

Autenticación en Squid

Es muy útil el poder establecer un sistema de autenticación para poder acceder hacia Internet, pues esto permite controlar quienes si y quienes no accederán a Internet sin importar desde que máquina de la red local lo hagan. Será de modo tal que tendremos un doble control, primero por dirección IP y segundo por nombre de usuario y clave de acceso.

Eligiendo el módulo de autenticación.

Se consideraran dos opciones de autenticación una a través de texto simple con claves de acceso creadas con htpasswd o bien a través de un servidor LDAP, lo cual constituye una solución más robusta.

Autenticación Simple

Squid puede utilizar el módulo ncsa_auth, de la NCSA (National Center for Supercomputing Applications), y que ya viene incluido como parte del paquete principal de Squid en la mayoría de las distribuciones actuales. Este módulo provee una autenticación muy sencilla a través de un archivo de texto simple cuyas claves de acceso fueron creadas con htpasswd.
Creación del archivo de claves de acceso.

Se requerirá la creación previa de un archivo que contendrá los nombres de usuarios y sus correspondientes claves de acceso (cifradas). El archivo puede localizarse en cualquier lugar del sistema, con la única condición que sea asequible para el usuario squid.

Debe procederse a crear un archivo /etc/squid3/claves:

#touch /etc/squid3/claves

Salvo que vaya a utilizarse un guión a través del servidor web para administrar las claves de acceso, como medida de seguridad, este archivo debe hacerse para que solo el usuario squid pueda leerlo o escribirlo:

#chmod 600 /etc/squid3/claves
#chown squid:squid /etc/squid3/claves

A continuación deberemos dar de alta las cuentas que sean necesarias, utilizando el mandato htpasswd -mismo que viene incluido en el paquete de apache. Ejemplo:

#htpasswd /etc/squid/claves areyes

Lo anterior solicitará teclear una nueva clave de acceso para el usuario joseperez y confirmar tecleando ésta de nuevo. Repita con el resto de las cuentas que requiera dar de alta.

Todas las cuentas que se den de alta de este modo son independientes a las ya existentes en el sistema. Al dar de alta una cuenta o cambiar una clave de acceso lo estará haciendo EXCLUSIVAMENTE para el acceso al servidor Proxy. Las cuentas son independientes a las que se tengan existentes en el sistema como serían shell, correo y Samba.
Parámetros en /etc/squid3/squid.conf

Lo siguiente será especificar que programa de autenticación se utilizará. Localice la sección que corresponde a la etiqueta auth_param basic program. Por defecto no está especificado programa alguno. Considerando que ncsa_auth se localiza en /usr/lib/squid3/ncsa_auth, procederemos a añadir el siguiente parámetro:

auth_param basic program /usr/lib/squid/ncsa_auth /etc/squid3/claves

/usr/lib/squid3/ncsa_auth corresponde a la localización de el programa para autenticar y /etc/squid3/claves al archivo que contiene las cuentas y sus claves de acceso.

Autenticación a través del módulo LDAP.

Considerando que se ha configurado exitosamente OpenLDAP como servidor de autenticación, solo basta definir el directorio y el servidor LDAP a utilizar.

La sintaxis utilizada para squid_ldap_auth es la siguiente:

#squid_ldap_auth -b "Directorio-o-DN-a-utilizar" servidor-ldap-a-utilizar

Ejemplo:

#squid_ldap_auth -b "cn=people,dc=su-dominio,dc=com" 127.0.0.1

Parámetros en /etc/squid3/squid.conf

Se debe editar el archivo /etc/squid3/squid.conf y se especificar el programa de autenticación se utilizará. Localice la sección que corresponde a la etiqueta auth_param basic program. Por defecto no está especificado programa alguno. Considerando que squid_ldap_auth se localiza en /usr/lib/squid3/ncsa_auth, procederemos a añadir el siguiente parámetro:

auth_param basic program /usr/lib/squid/squid_ldap_auth -b "cn=people,dc=su-dominio,dc=com" 127.0.0.1

Lo anterior conecta al directorio dc=su-red-local,dc=com en el servidor LDAP en 127.0.0.1.

Listas y reglas de control de acceso.

El siguiente paso corresponde a la definición de una Lista de Control de Acceso. Especificaremos una denominada passwd la cual se configurará para utilizar obligatoriamente la autenticación para poder acceder a Squid. Debe localizarse la sección de Listas de Control de Acceso y añadirse la siguiente línea:

acl password proxy_auth REQUIRED

Habiendo hecho lo anterior, deberemos tener en la sección de Listas de Control de Acceso algo similar a lo siguiente:

Listas de Control de Accesos: autenticación.

#
 # Recommended minimum configuration:
acl all src 0.0.0.0/0.0.0.0
acl manager proto cache_object
acl localhost src 127.0.0.1/255.255.255.255

acl redlocal src 192.168.1.0/255.255.255.0
acl password proxy_auth REQUIRED

Procedemos entonces a modificar la regla de control de accesos que ya teníamos para permitir el acceso a Internet. Donde antes teníamos lo siguiente:
http_access allow redlocal

Le añadimos passwd, la definición de la Lista de Control de Acceso que requiere utilizar clave de acceso, a nuestra regla actual, de modo que quede como mostramos a continuación:
http_access allow redlocal password

Habiendo hecho lo anterior, la zona de reglas de control de acceso debería quedar de este modo:

Reglas de control de acceso: Acceso por clave de acceso.
#
 # INSERT YOUR OWN RULE(S) HERE TO allow ACCESS FROM YOUR CLIENTS
#
http_access allow localhost
http_access allow redlocal password
http_access deny all