Instalación de un escudo de aplicación

Objetivos: **Mod Security**

Angel Alonso Párrizas

Índice

1. Introducción

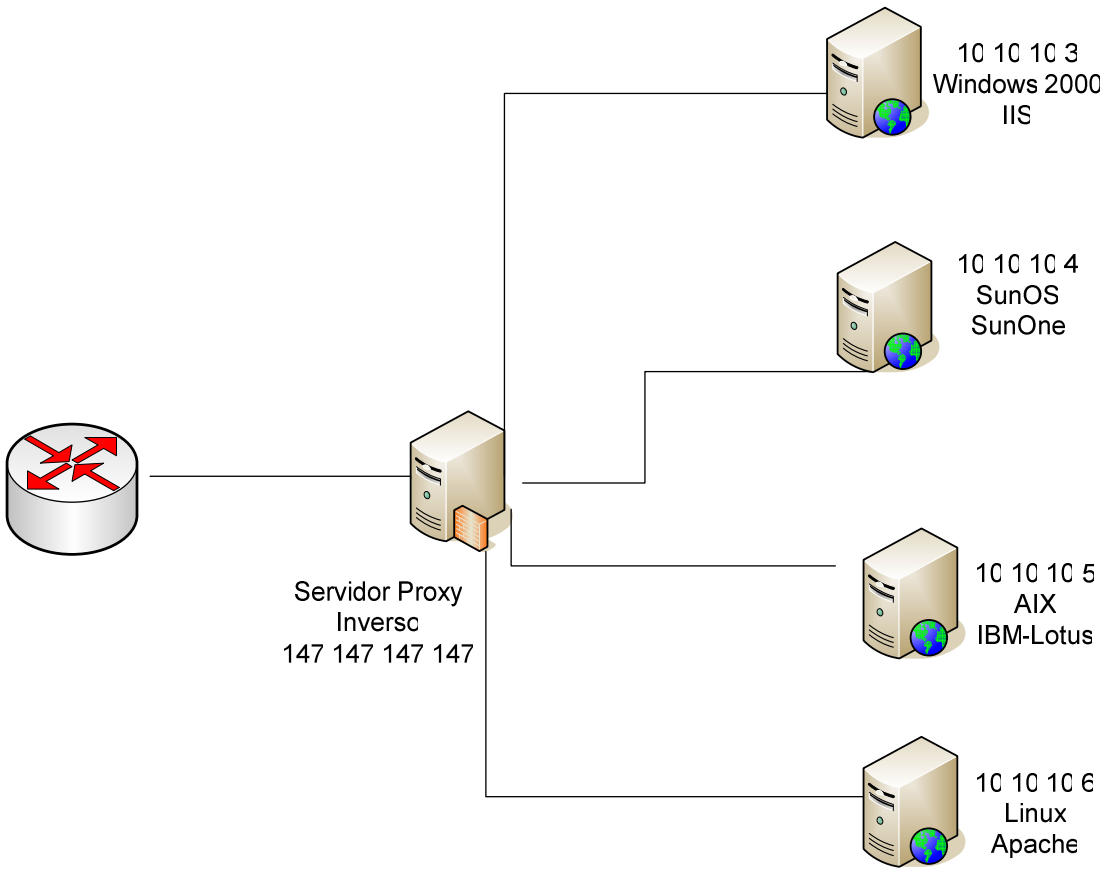
1.1. Objeto

Se pretende configurar un servidor Web Apache como escudo de aplicación. La idea es configurar el servidor para que analice las peticiones 'HTTP', mire si la petición es correcta y la reenvíe al servidor correspondiente, es decir realizará la vez de Proxy-Inverso.

1.2. Definiciones y funcionamiento.

Un Proxy inverso es un tipo Proxy que funciona de manera opuesta a un Proxy tradicional. Se podría definir como un 'front-end' entre varios servidores y el usuario que quiere ver el contenido de alguno/s de ellos.
El modulo 'mod_security' es un módulo de seguridad para apache que analiza las peticiones 'HTTP', bien en la propia URL o bien mediante el método POST buscando patrones definidos por el usuario, con el fin de poder evitar ataques SQL-Injection, Cross-site-scripting, LDAP-injection, etc.

Un esquema de una posible topología podría ser:



Como se ve en la figura, cualquier petición a un recurso web de cualquiera de los servidores con IP privada será hecha al servidor Proxy inverso, quien a su vez reenviará la petición al servidor correspondiente.

2. Instalación y configuración sobre Apache 2.x bajo Linux.

La versión de apache que se utilizará, será cualquiera de la rama 2.x. La rama 2.x incorpora una serie de mejoras para el tratamiento de las peticiones mediante el Proxy-inverso, de tal manera que con la instalación de los módulos correspondientes no habrá que realizar ninguna configuración adicional.

2.1. Instalación de los binarios del apt.

La distribución elegida para la instalación del servidor apache es una Debian 3.0. Los paquetes binarios que se necesitan no están en la rama estable, por lo que habrá que configurar el 'sources.list' con algún servidor fuente de la rama inestable.
Los paquetes a instalar son: apache2, libapache2-mod-php4 (para integrar PHP), libapache2-mod-security y todas las dependencias que se instalen con estos paquetes (apt-get update; apt-get install apache2 libapache2-mod-php4 libapache2-mod-security).

2.2. Configuración de los ficheros.

Una vez instalado el servidor apache, debe de configurar el mod_security y el Proxy-inverso a modo de Proxy transparente.
Una de las cosas que cambia en la versión de apache 2.x frente a la 1.x es los ficheros de configuración, aunque la sintaxis es la misma, la forma de organizar los datos dentro de los ficheros difiere.

2.2.1 Cargar los módulos del apache.

En apache2 existen dos directorios para manejar los módulos, estos dos directorios son: `/etc/apache2/mods-available` cuyo contenido serán los módulos que se han integrado con el apache y pueden ser utilizados en cualquier momento, y `/etc/apache/mods-enabled` que sirven para activar los módulos correspondientes. Para activar un módulo es tan sencillo como crear un link simbólico en el directorio mods-enabled al módulo correspondiente en mods-available.

En el caso de este documento.

```
bash# cd /etc/apache2/mods-enabled/;
bash# ln –s /etc/apache2/mods-available/proxy.load proxy.load
```

Se hará para los módulos (si no está hecho): `actions.load mod-security.load proxy.conf proxy_ftp.load rewrite.load userdir.load cgi.load php4.load proxy_connect.load proxy.load userdir.conf.`

2.2.2 Configuración del fichero /etc/apache2/mods-enabled/proxy.conf

Este fichero debe contener estas líneas:

```
<IfModule mod_proxy.c>

    # Enable/disable the handling of HTTP/1.1 "Via:" headers.
    # ("Full" adds the server version; "Block" removes all outgoing Via: headers)
    # Set to one of: Off | On | Full | Block

    ProxyVia On

    # To enable the cache as well, edit and uncomment the following lines:
    # (no cacheing without CacheRoot)

    CacheRoot "/var/cache/apache2/proxy"
    CacheSize 5
    CacheGcInterval 4
    CacheMaxExpire 24
    CacheLastModifiedFactor 0.1
    CacheDefaultExpire 1
    # Again, you probably should change this.
    #NoCache a_domain.com another_domain.edu joes.garage_sale.com


</IfModule>
```

Que es una configuración global del Proxy. Básicamente se configura la caché del Proxy inverso.

2.2.3 Configuración de virtual-host (vhost) para cada máquina.

Por cada máquina de detrás del Proxy habrá que configurar un virtual-host donde se especificará las políticas concretas del mod_security para esa máquina partícular, es decir, según el tipo de ataque que pueda darse en cada servidor (dependerá del servicio que se de, BD, LDAP..) se definirá unas políticas u otras.
Existe una configuración por defecto que es la del servidor raíz llamada ‘default’ que no habrá que tocar.

El funcionamiento de los directorios donde se definen los vhost es análogo al de los módulos, es decir en /etc/apache2/sites-available/ se crear los vhost y en /etc/apache2/sites-enabled/ se dice mediante un link simbólico cuales están activos.

En el caso del que se ocupa este documento:

1) <VirtualHost *>
2) ServerName www.seguridad.com
3) # ServerAlias domain.tld *.domain.tld
4) DocumentRoot /var/www/objetivo

```
5)    ProxyRequests Off
6)          <Proxy *>
7)                  Order deny,allow
8)                   #Deny from all
9)                   Allow from all
10)         </Proxy>
11)    ProxyPass / http://10.10.10.3/
12)    ProxyPassReverse / http://10.10.10.3/
13)     ErrorLog /var/log/apache2/seguridad/error.log
14)     CustomLog /var/log/apache2/seguridad/access.log combined
15)
16) <IfModule mod_security.c>
17) # Turn the filtering engine On or Off
18) SecFilterEngine On
19)
20) # Make sure that URL encoding is valid
21) SecFilterCheckURLEncoding On
22)
23) # Unicode encoding check
24) SecFilterCheckUnicodeEncoding Off
25) # Only allow bytes from this range
26) SecFilterForceByteRange 0 255
27) # Only log suspicious requests
28) SecAuditEngine RelevantOnly
29) # The name of the audit log file
30) SecAuditLog /var/log/apache2/seguridad/audit_log
31)
32) # Debug level set to a minimum
33) SecFilterDebugLog /var/log/apache2/seguridad/modsec_debug_log
34) SecFilterDebugLevel 0
35)
36) # Should mod_security inspect POST payloads
37) SecFilterScanPOST On
38) # By default log and deny suspicious requests  with HTTP status 500
39) SecFilterDefaultAction "deny,log,status:500"
40)
41) #accesos a directorios invalidos
42) SecFilter /bin/*
43) SecFilter /etc/*
44) #ataques de directorio transversal
45) SecFilter "\.\./"
46)
47) #ataques XSS
48) SecFilter "<(.|\n)+>"
```

```
49) SecFilter "<[[:space:]]*script"
50)
51) #ataques SQL-Injection
52) SecFilter "delete[[:space:]]+from"
53) SecFilter "insert[[:space:]]+into"
54) SecFilter "select.+from"
55)
56) SecFilterSelective ARG_highlight %27
57) SecFilter cd\x20/tmp
58) SecFilter wget\x20
59)
60) </IfModule>
61) </VirtualHost>
```

De la línea 1 a la 5 definimos el virtual host.
Las lineas 6-14 indican la configuración del Proxy inverso, las maquinas que tendran acceso y a donde se redirigan las peticiones que tengan en la cabecera host HTTP 1.1 el dominio www.seguridad.com que en este caso es el directorio raíz de la máquina con IP 10.10.10.3.
Las líneas siguientes son las reglas para el módulo mod_security, en las que se definen según el ataque una expresión regular. Además se define la acción por defecto ante un intento de ejecución de algunas de las reglas prohibidas, como en este caso un código de error HTTP 500. Además se define los ficheros de logs donde se almacenan las peticiones en las que el mod_security ha tenido que intervenir.

Para cada máquina de detrás del Proxy habría que definir un fichero como este.

2.2.4 Configuración con soporte HTTPS desde el Proxy hacia una máquina back-end.

La idea de está configuraciones permitir conexiónes cifradas desde el Proxy hacia un servidor que existe detrás.
Para ello la configuración ideal es:

```
<VirtualHost *>
ServerName www.sanidad.com
# ServerAlias domain.tld *.domain.tld
 DocumentRoot /var/www/cominio
  ProxyRequests Off

      <Proxy *>
           Order deny,allow
                #Deny from all
                Allow from all
```

```
            </Proxy>
        AllowCONNECT 443

      ProxyPass / https://dominio.el.que.sea /
        ProxyPassReverse / https://dominio.el.que.sea /
      SSLProxyEngine on
          ErrorLog /var/log/apache2/dominio/error.log
          CustomLog /var/log/apache2/dominio access.log combined
</VirtualHost>
```

2.2.5 Conexión mediante HTTPS desde el cliente hacia el Proxy.

```
NameVirtualHost *:443  #se especifica el puerto 443 para conexion cifrada

<VirtualHost *:443>

ServerName www.dominio.com
# ServerName dominio.el.que.sea.es
# ServerAlias domain.tld *.domain.tld
DocumentRoot /var/www/cominio
  ProxyRequests Off

      <Proxy *>
            Order deny,allow
                  #Deny from all
                  Allow from all
            </Proxy>
      AllowCONNECT 443

      ProxyPass / https://dominio.es/  #Donde nos queremos conectar (back-end)
      ProxyPassReverse / https://dominio.es/  #Donde nos queremos conectar (back-
        end)

      ServerSignature On  #activamos el cifrado. Importante para que vaya el https
      SSLProxyEngine on
      SSLEngine on
      SSLCertificateFile /etc/apache2/ssl/apache.pem
#      SSLRequireSSL

          ErrorLog /var/log/apache2/sanidad/error.log
          CustomLog /var/log/apache2/sanidad/access.log combined
</VirtualHost>
```