Índice:

LILLO ROMERO MIGUEL - 71368153G

Firmado digitalmente por LILLO ROMERO MIGUEL - 71368153G
Fecha: 2024.01.15 11:12:57 +01'00'

Resumen informe de INCIBE (INTECCO): “Guía básica para la securización del servidor Web Apache”.

a)Instalación

Solución:

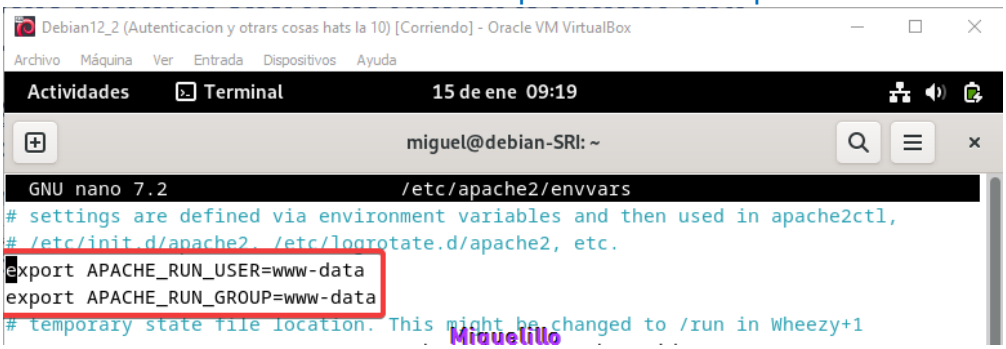
A la hora de instalara apache es recomendable borrar el fichero predeterminado index.htmls ya que puede exponer la configuración de nuestro servidor apache. Que por defecto se encuentra en **/var/www/html/index.html**

b)Configuración usuario y grupo

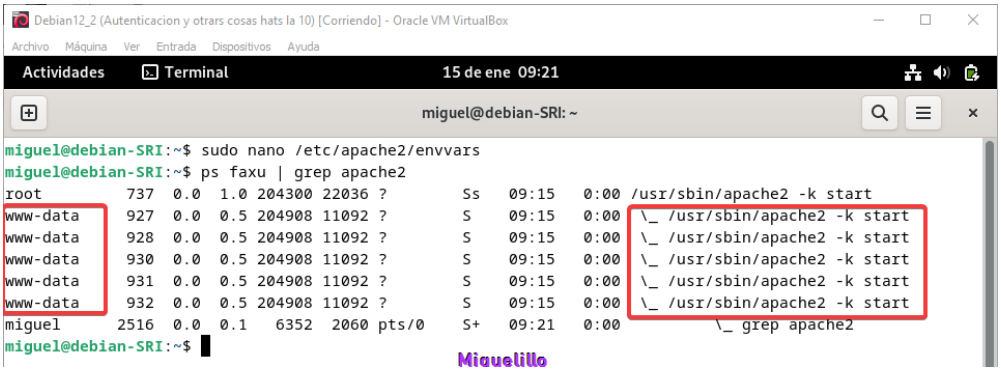
Solución:

Si arrancamos el servidor web con root corremos el riesgo que frente a un ataque exitoso poder otorgar permisos de administrador al atacante, con lo cual es recomendable configurar los servicios acorde a un usuario con las funcionalidades justas para minimizar el impacto de un ataque. Por defecto viene configurado pero es importante verificar que el usuario y el grupo sea **www-data**

Vemos el archivo de variables /etc/apache2/envvars para verificarlo



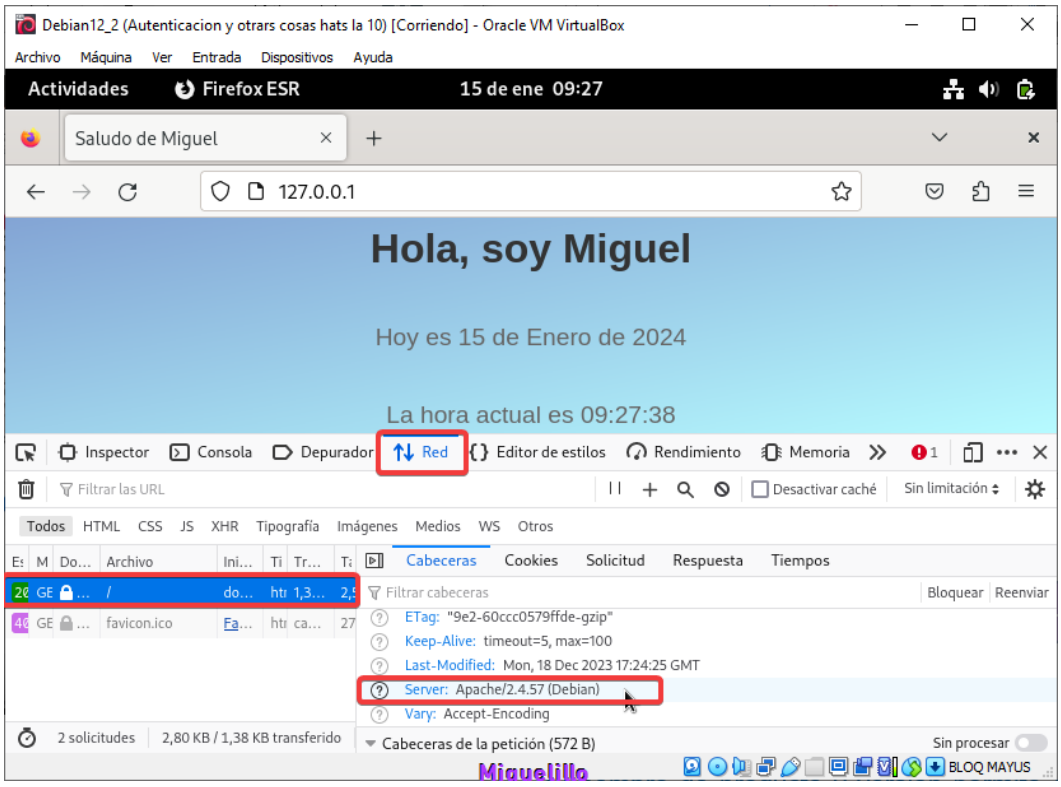
Y miramos los procesos para verificar la ejecución con ese usuario



c)Ocultar versiones

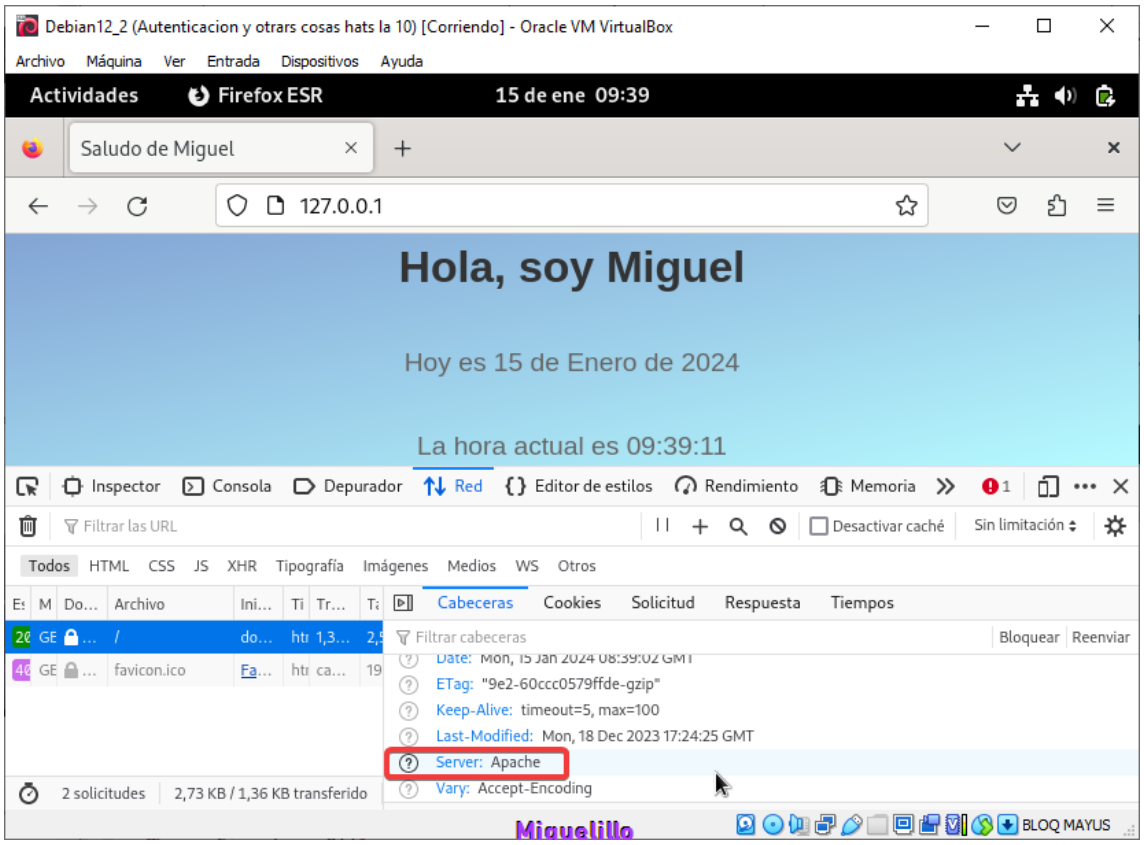
Solución:

El nombre de producto y versión permite a los atacantes obtener información sobre la versión permitiéndoles buscar exploids. Por defecto son visibles si inspeccionamos la pagina vamos a networks y vamos a / y a Server podemos ver la versión y sistema operativo



Para deshabilitarlo editaremos el archivo **/etc/apache2/conf-enabled/security.conf** y comentaremos con un # las directivas "**ServerSignature On**" y "**ServerTokens OS**". Remplazandolas (descomentando) por "**ServerSignature Off**" y "**ServerTokens ProductOnly**"

Una vez reiniciado el servicio y aplicado podemos ver que no se ve la versión solamente el sevidor web que se está utilizando

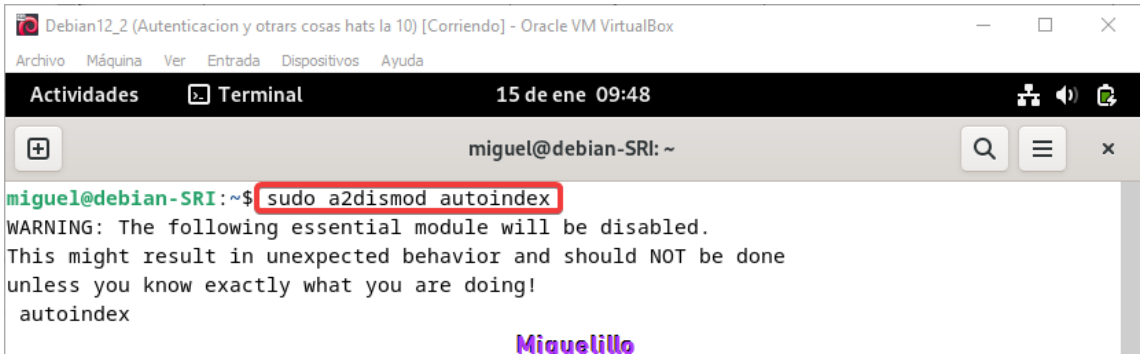


d)Exposición mínima de modulos

Solución:

Desactivar módulos por defecto que no estemos utilizando reducirá la exposición y la carga del servidor. Para ver los módulos por defecto podemos hacer un *sudo apache2ctl -M*

Para desactivarlo usamos *a2dismod*. Es importante que investiguemos que hace cada modulo porque podemos dejar el servidor no operativo

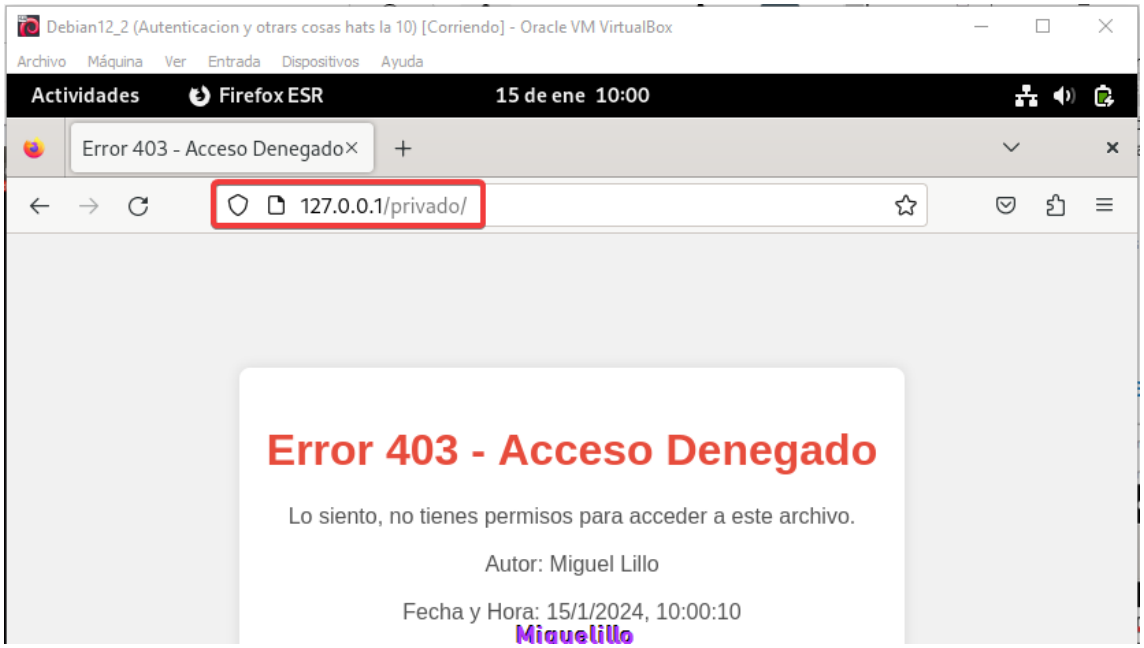


e)Directiva “Options”

Solución:

Si no queremos que se indexen los ficheros de un directorio usamos la opción “Options -Indexes +FollowSymlinks”

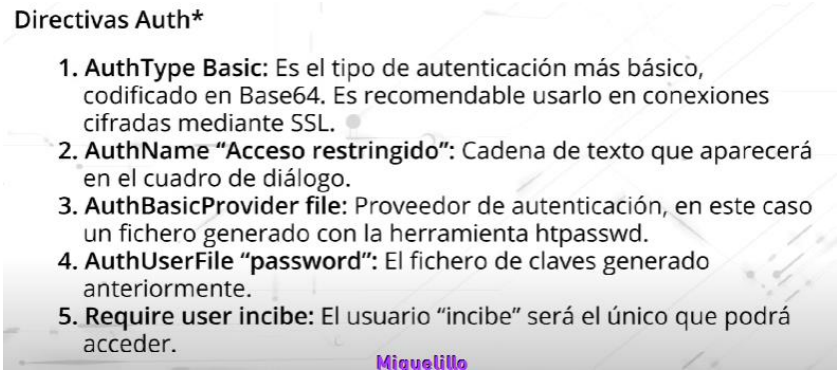
Si probamos a acceder nos da un fallo por permisos



f)Directiva “Auth*” y “Require”

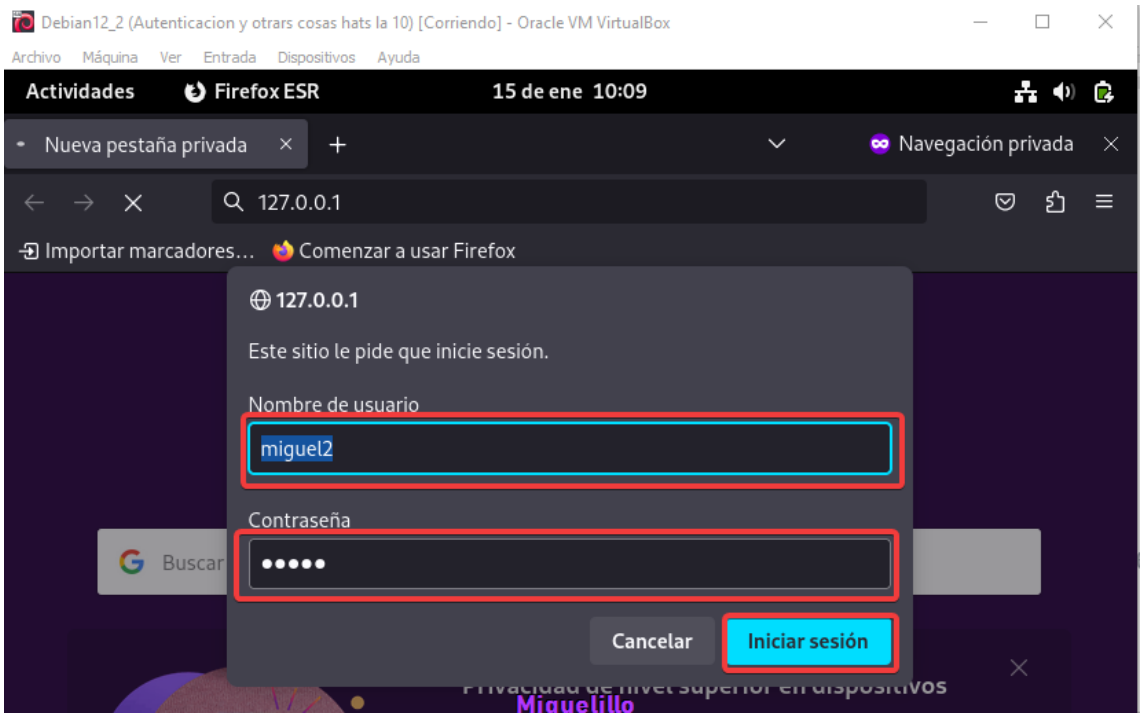
Solución:

Si queremos proteger la web podemos poner directivas de autenticación para ello generamos una cotraseña con *sudo htpasswd -c passwords inves* D Directivas Auth

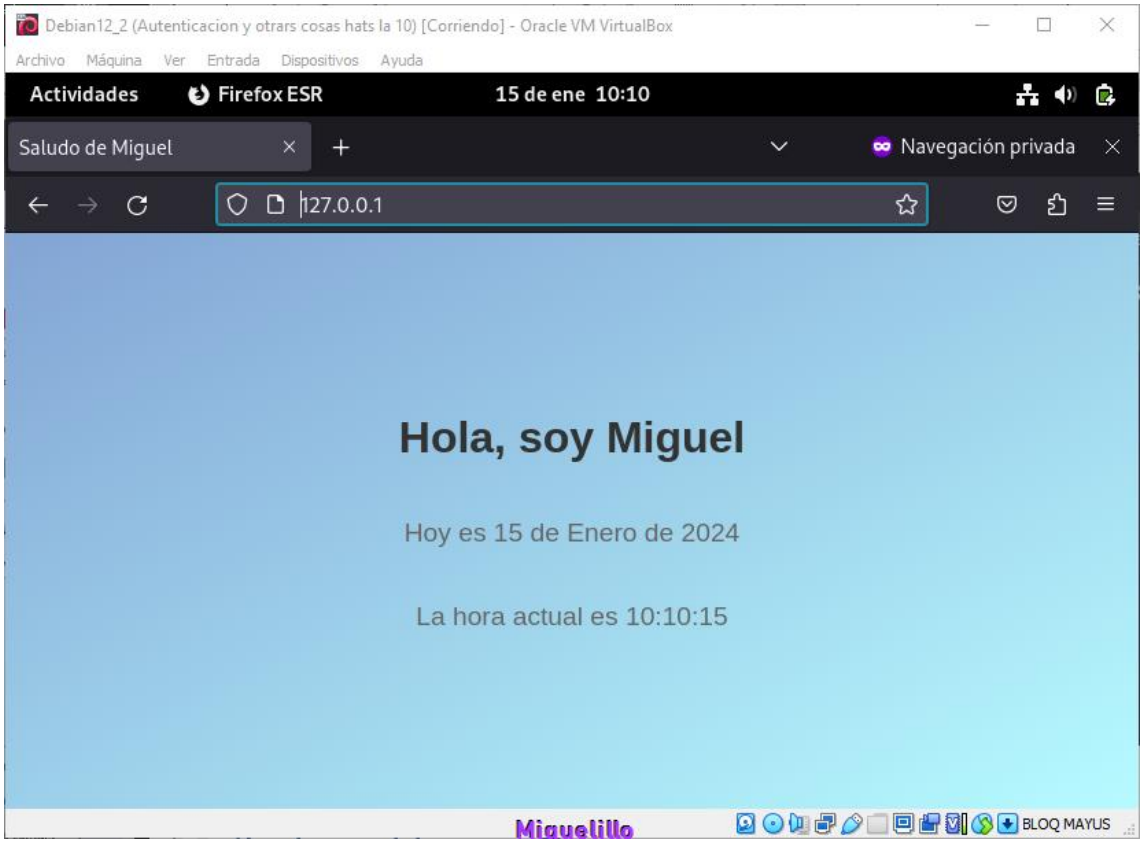


Configuramos las directivas

Nos pide las credenciales



Accedemos



Con required podemos restringir el acceso al contenido por host por ip por usuario por grupo. Por ejemplo en este ejemplo se deniega el acceso al local host pero se permite el acceso a todos los demás usuarios. Con lo cual podemos usarlo de manera jerárquica

Otros tipos de required

Require all granted: Proporciona acceso de forma incondicional a los recursos donde se aplique.

Require all denied: Deniega acceso de forma incondicional a los recursos donde se aplique.

Require env nombre: Proporciona acceso sólo si existe una variable de entorno de nombre "nombre" en el sistema.

Require method htttp-method: Proporciona acceso a través de métodos HTTP definidos en "httpmethod".

Require [not] ip <IP/IPs>: Proporciona acceso a una lista de IPs. Si incluye [not] deniega el acceso a esa lista.

Require [not] host <host/hosts>: Proporciona acceso a una lista de hosts. Si incluye [not] deniega el acceso a esa lista.

Miguelillo

g)Fichero .htaccess

Solución:

Permiten personalizar directivas y parámetros que algunos pueden ser muy útiles como las siguientes directivas

- **Control de acceso al sitio:** Deniega el acceso a ciertas IPs.
  - *Allow from all*
  - *Deny from 1.2.3.4*
  - *Deny from 12.24*
- **Forzar protocolo HTTPS:** Aplicar un redireccionamiento al protocolo seguro HTTPS.
  - RewriteEngine on
  - RewriteCond %{HTTPS} !=on [NC]
  - RewriteRule ^.*$ https://%{SERVER_NAME}%{REQUEST_URI} [R,L]
- **Evitar "HotLinking":** El "hotlinking" es el abuso de recursos ajenos a través de enlaces directos desde otras webs con el consiguiente consumo de datos.
- *RewriteEngine on*
- *RewriteCond %{HTTP_REFERER} !^$*
- *RewriteCond %{HTTP_REFERER} !^http://(www\.)?midominio.org/.*$ [NC]*
- *RewriteRule .*\.(jpe?g|gif|bmp|png)$ [L]*

Miguelillo

h) Módulo “mod_security”

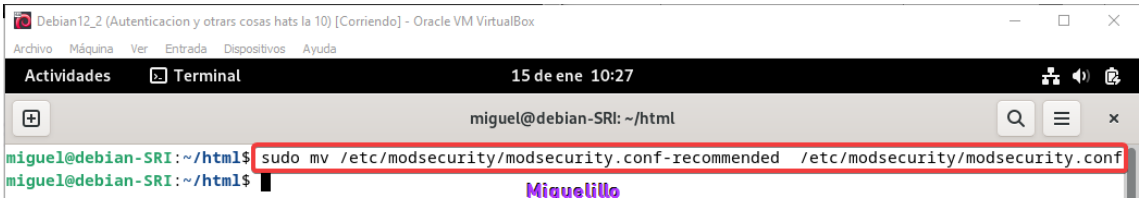
Solución:

Este módulo protege de ataques web comúnmente conocidos. Actúa como firewall de aplicación. Dispone de modos de actuación activos y pasivos el método pasivo almacena los ataques, pero no los para. Inspecciona el traficos HTTP y deniega acceso a aquellas peticiones donde encuentra algún patrón de ataque conocido.

Lo instalamos

Activamos el módulo

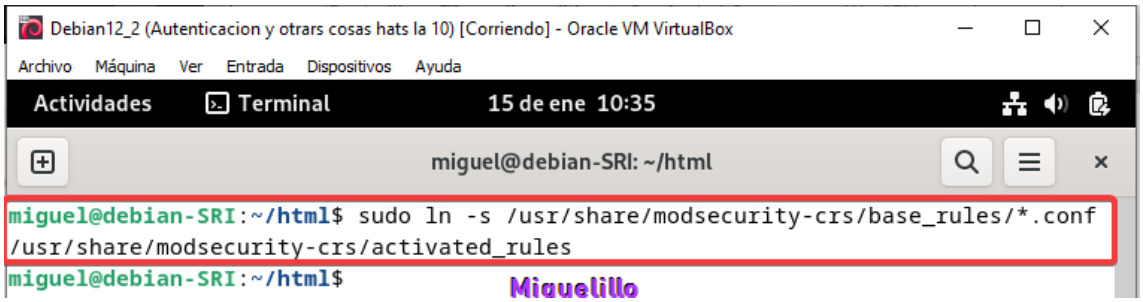
Renombramos el fichero de configuración



Editamos el fichero de configuración y cambiamos “SecRuleEngine” a “On” y “SecRequestBodyAccess” a “Off”

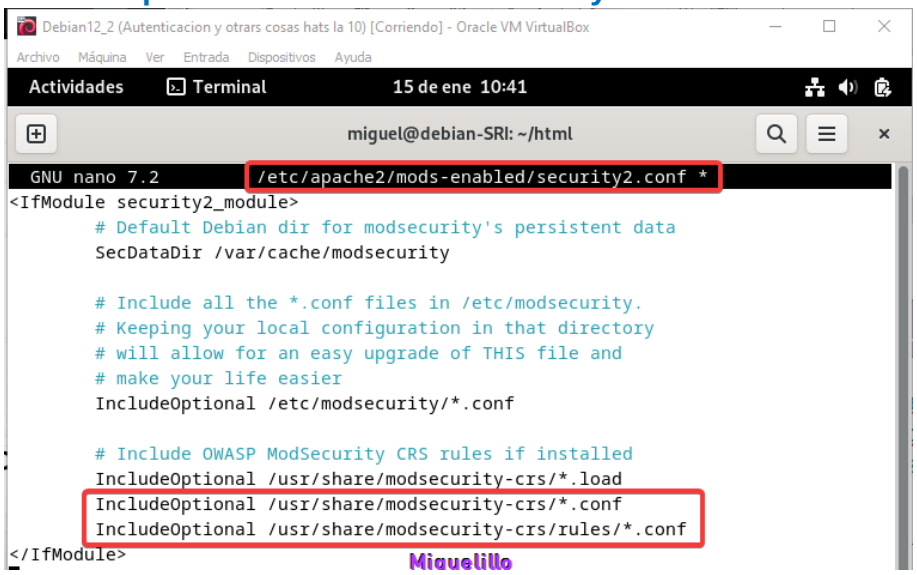
Las reglas están disponible en **/usr/share/modsecurity-crs/base_rules**, **/usr/share/modsecurity-crs/optional_rules** y **/usr/share/modsecurity-crs/experimental_rules.** Para habilitarlas creamos un enlace simbólico

Para activarlas hacemos un enlace simbólico



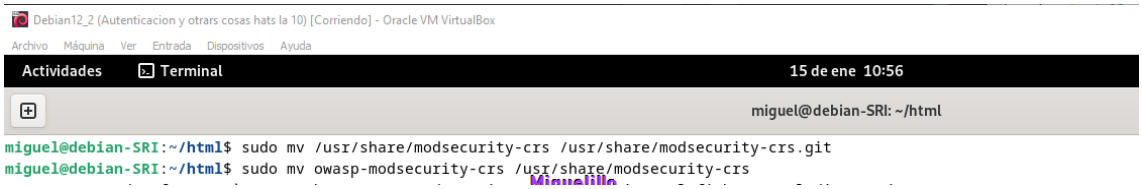
Para añadir las reglas editamos el fichero **/etc/apache2/mods-enabled/security2.conf** y añadimos estas 2 líneas de comando "**IncludeOptional /usr/share/modsecurity-crs/*.conf**

IncludeOptional /usr/share/modsecurity-crs/rules/*.conf"

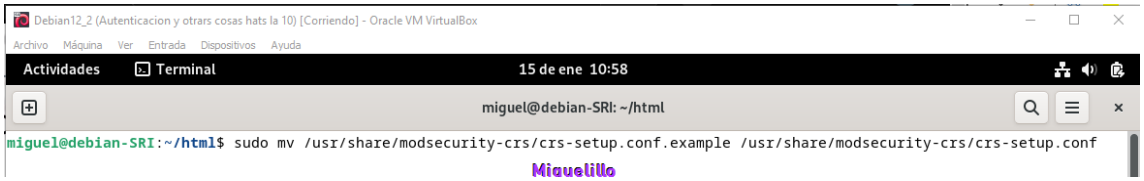


Para descargar las reglas OWASP creadas por la comunidad recomendadas para la mayoría de los ataques web, estas reglas detecta inyecciones SQL y las deniega. También detecta ataques de inclusión de ficheros entre otros las instalamos desde github. c

Cambiamos le nombre al directorio y movemos el archivo que hemos descargado

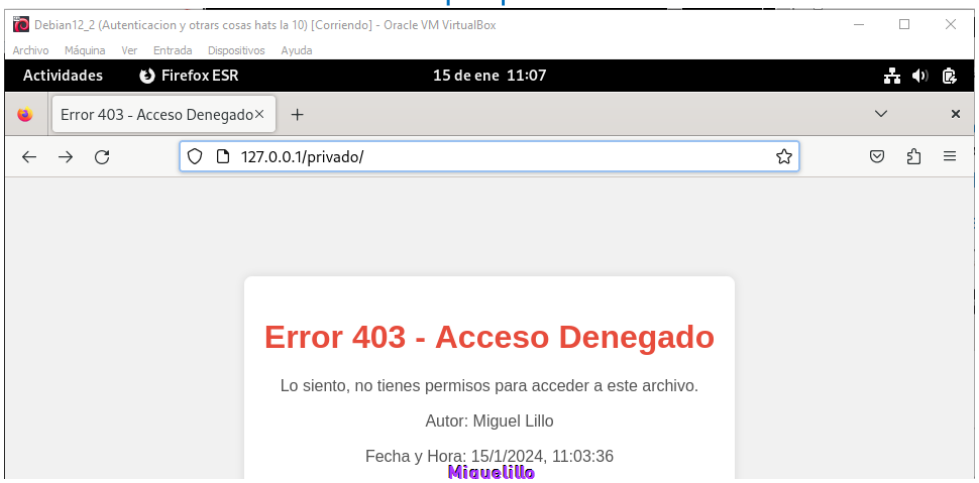


Renombramos el fichero del interior



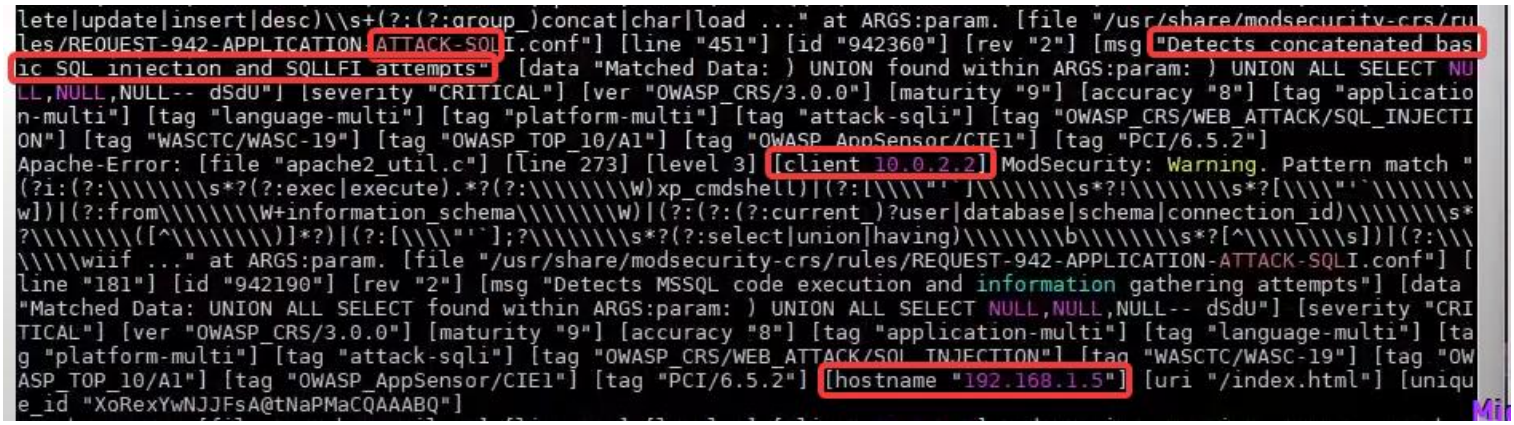
Reiniciamos el servicio apache y para monitorear ponemos *sudo tail -f /var/log/apache2/modsec_audit.log* dependiendo lo que queramos monitorizar pondremos filtros al final del comando como | *grep ATTACK-SQL*

Prueba con un fallo de acceso por permisos de la web



Este registro se va actualizando dinámicamente hasta que lo cierras con **Ctrl + C**

Un ejemplo de un ataque SQL



Fuente

https://youtu.be/rjsgaxV8irM?si=8YJ9egRFiJV3EHjX