Práctica 2.3. Autenticación en Nginx

LA PRÁCTICA 2.1 DEBE ESTAR REALIZADA

Introducción

En el contexto de una transacción HTTP, la autenticación de **acceso básica** es un método diseñado para permitir a un navegador web, u otro programa cliente, proveer credenciales en la forma de usuario y contraseña cuando se le solicita una página al servidor.

La autenticación básica, como su nombre lo indica, es la forma más básica de autenticación disponible para las aplicaciones Web. Fue definida por primera vez en la especificación HTTP en sí y no es de ninguna manera elegante, pero cumple su función.

Este tipo de autenticación es el tipo más simple disponible, pero adolece de importantes problemas de seguridad que no la hacen recomendable en muchas situaciones. No requiere el uso ni de cookies, ni de identificadores de sesión, ni de página de ingreso.

SE NECESITA EL PAQUETE OPENSSL INSTALADO

1. Creación de usuario y contraseñas para el acceso web

Crearemos un archivo oculto llamado *".htpasswd"* en el directorio de configuración */etc/nginx* donde guardar nuestros usuarios y contraseñas (la -c es para crear el archivo):

```
sudo sh -c "echo -n 'vuestro_nombre:' >> /etc/nginx/.htpasswd"
```

Ahora crearemos un pasword cifrado para el usuario:

```
sudo sh -c "openssl passwd -apr1 >> /etc/nginx/.htpasswd"
```

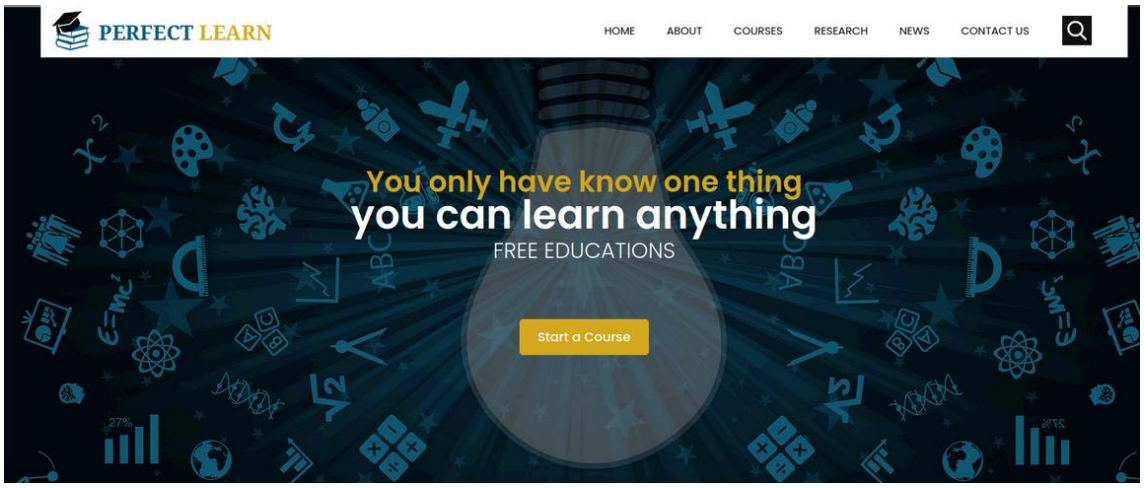
Este proceso se podrá repetir para tantos usuarios como haga falta.

- Crea dos usuarios, uno con tu nombre y otro con tu primer apellido
- Comprueba que el usuario y la contraseña aparecen cifrados en el fichero:

```
cat /etc/nginx/.htpasswd
```

2. Configurando el servidor Nginx para autenticación básica

Editaremos la configuración del *server block* sobre el cual queremos aplicar la restricción de acceso. Utilizaremos para esta autenticación el sitio web de *Perfect Learn*:



Recuerda que un *server block* es cada uno de los dominios (`server {...}` dentro del archivo de configuración) de alguno de los sitios web que hay en el seridor.

```
sudo nano /etc/nginx/sites-available/nombre_web
```

Debemos decidir qué recursos estarán protegidos. Nginx permite añadir restricciones a nivel de servidor o en un location (directorio o archivo) específico. Para nuestro ejemplo, vamos a proteger el document root (la raíz, la página principal) de nuestro sitio.

Utilizaremos la directiva `auth_basic` dentro del **location** y le pondremos el nombre a nuestro dominio que será mostrado al usuario al solicitar las credenciales. Por último, configuramos Nginx para que utilice el fichero que previamente hemos creado con la directiva `auth_basic_user_file` :

```
server {
        listen 80;
        listen [::]:80;

        root /var/www/webraul/html/simple-static-website;
        index index.html index.htm index.nginx-debian.html;

        server_name nombre_web;

        location / {
```

```
        auth_basic  "Área restringida";
        auth_basic_user_file    /etc/nginx/.htpasswd;
            try_files $uri $uri/ =404;
        }
}
```

Una vez terminada la configuración, reiniciamos el servicio para que aplique nuestra política de acceso.

```
sudo systemctl restart nginx
```

¡TAREA!

Borra las dos líneas que hacen referencia a la autenticación básica en el location del directorio raíz. Tras ello, añade un nuevo *location* debajo con la autenticación básica para el archivo/sección `contact.html` únicamente.

3. Combinación de la autenticación básica con la restricción de acceso por IP

La autenticación básica HTTP puede ser combinada de forma efectiva con la restricción de acceso por dirección IP. Se pueden implementar dos escenarios:

- Un usuario debe estar ambas cosas, autenticado y tener una IP válida
- Un usuario debe o bien estar autenticado, o bien tener una IP válida

Veamos cómo lo haríamos:

1. Como permitir o denegar acceso sobre una IP concreta (directivas **allow** y **deny**, respectivamente). Dentro del block server o archivo de configuración del dominio web, que recordad está en el directorio `sites-available`:

```
location /api {
    #...
    deny  192.168.1.2;
    allow 192.168.1.1/24;
    allow 127.0.0.1;
    deny  all;
}
```

El acceso se garantizará ala IP `192.168.1.1/24`, excluyendo a la dirección `192.168.1.2`.

Hay que tener en cuenta que las directivas `allow` y `deny` se irán aplicando en el orden en el que aparecen el archivo.

Aquí aplican sobre la location /api (esto es sólo un ejemplo de un hipotético directorio o archivo), pero podrían aplicar sobre cualquiera, incluida todo el sitio web, la location raíz /.

La última directiva `deny all` quiere decir que por defecto denegaremos el acceso a todo el mundo. Por eso hay que poner los `allow` y `deny` más específicos justo antes de esta, porque al evaluarse en orden de aparición, si los pusiéramos debajo se denegaría el acceso a todo el mundo, puesto que `deny all` sería lo primero que se evaluaría.

2. Combinar la restricción IP y la autenticación HTTP con la directiva **satisfy**.
   Si establecemos el valor de la directiva a “all”, el acceso se permite si el cliente satisface ambas condiciones (IP y usario válido). Si lo establecemos a “any”, el acceso se permite si se satisface al menos una de las dos condiciones

```
location /api {
    #...
    satisfy all;

    deny  192.168.1.2;
    allow 192.168.1.1/24;
    allow 127.0.0.1;
    deny  all;

    auth_basic           "Administrator's Area";
    auth_basic_user_file conf/htpasswd;
}
```

TAREA

Configura Nginx para que no deje acceder con la IP de la máquina anfitriona al directorio raíz de tu web. Modifica su server block o archivo de configuración. Comprueba como se deniega el acceso.

Muestra la página de error en el navegador

TAREA

Configura Nginx para que desde tu máquina anfitriona se tenga que tener tanto una IP válida como un usuario válido, ambas cosas a la vez, y comprueba que sí puede acceder sin problemas

ACTIVIDADES

1. Supongamos que yo soy el cliente con la IP 172.1.10.15 e intento acceder al directorio *web_muy_guay* de mi sitio web, equivocándome al poner el usuario y contraseña. ¿Podré acceder? ¿Por qué?

```
    location /web_muy_guay {
    #...
    satisfy all;
    deny  172.1.10.6;
    allow 172.1.10.15;
    allow 172.1.3.14;
    deny  all;
    auth_basic "Cuestión final 1";
    auth_basic_user_file conf/htpasswd;
}
```

2. Supongamos que yo soy el cliente con la IP 172.1.10.15 e intento acceder al directorio *web_muy_guay* de mi sitio web, introduciendo correctamente usuari y contraseña. ¿Podré acceder? ¿Por qué?

```
    location /web_muy_guay {
    #...
    satisfy all;
    deny  all;
    deny  172.1.10.6;
    allow 172.1.10.15;
    allow 172.1.3.14;

    auth_basic "Cuestión final 2: The revenge";
    auth_basic_user_file conf/htpasswd;
}
```

3. Supongamos que yo soy el cliente con la IP 172.1.10.15 e intento acceder al directorio *web_muy_guay* de mi sitio web, introduciendo correctamente usuario y contraseña. ¿Podré acceder? ¿Por qué?

```
    location /web_muy_guay {
    #...
    satisfy any;
    deny  172.1.10.6;
    deny 172.1.10.15;
    allow 172.1.3.14;

    auth_basic "Cuestión final 3: The final combat";
    auth_basic_user_file conf/htpasswd;
}
```

4. A lo mejor no sabéis que tengo una web para documentar todas mis excursiones espaciales con Jeff, es esta: Jeff Bezos y yo

Supongamos que quiero restringir el acceso al directorio de proyectos porque es muy secreto, eso quiere decir añadir autenticación básica a la URL:Proyectos

Completa la configuración para conseguirlo:

```
    server {
        listen 80;
        listen [::]:80;
        root /var/www/freewebsitetemplates.com/preview/space-science;
        index index.html index.htm index.nginx-debian.html;
        server_name freewebsitetemplates.com www.freewebsitetemplates.com;
        location              {

            try_files $uri $uri/ =404;
        }
    }
```