Práctica 3 – Servicios avanzados de red: HTTP (1ª sesión)

1.1 Objetivo

El objetivo de esta práctica es conocer y familiarizarse con las tareas habituales en la administración y configuración, en un S.O. LINUX, de HTTP (*HyperText Transfer Protocol*) y concretamente de Apache2. Se proponen 3 ejercicios:

- Instalación y configuración básica de Apache2
- Configuración de *Virtual Hosts*
- Restricción de accesos

1.2 Información básica para la realización de la práctica

En esta sección se ofrece la información básica y las referencias necesarias para llevar a cabo las tareas que se proponen en la práctica.

1.2.1 Acceso al sistema y elección de sistema operativo

Para la realización de esta práctica, es necesario arrancar el equipo con la opción "Redes"→"Ubuntu 20.04". La práctica se realizará en parejas en donde uno de los equipos actuará como cliente y otro como servidor web. Será en este último en donde se configure adecuadamente Apache2.

Una vez que se haya identificado, puede pasar a modo *superusuario* mediante el siguiente comando, y utilizando la contraseña "**finisterre**":

```
# sudo su
```

1.2.2 Gestión de servicios básicos

Hay dos formas de gestionar servicios. Por un lado, los servicios básicos de red se pueden ejecutar y configurar desde un único superservidor denominado `xinetd`. En este caso, con un único proceso se monitorizan tantos puertos como sean necesarios y a petición de los clientes, el superservidor ejecutará el proceso correspondiente al servicio demandado. `xinetd` facilita una gestión unificada de todos los servicios, con mecanismos de seguridad de control de accesos, incluyendo restricciones horarias o accesos dependiendo de la IP de origen, además de otras muchas condiciones, con ayudas para la contabilidad y con posibilidad de registro de todos los eventos (en ficheros de `log`). La configuración de `xinetd` se realiza mediante el fichero `/etc/xinetd.conf`.

Para obtener más ayuda sobre la configuración de este servidor, consulte la sintaxis del fichero de configuración en el manual del sistema:

```
# man xinetd.conf
```

Alternativamente o complementariamente, todo servidor se puede ejecutar en modo *standalone*. Esto consiste básicamente en, tras editar los preceptivos ficheros de configuración del propio servicio, lanzar un ejecutable (`daemon`) en *background*, controlándolo posteriormente si fuese necesario con el comando `systemctl`.

1.2.3 Gestión y configuración básica de Apache2 (HTTP)

El protocolo HTTP facilita el acceso desde un cliente (que ofrece una interfaz universal o *navegador*) a objetos web (texto, imágenes, vídeo, etc) situados en un servidor identificado por su dirección IP o su nombre de dominio. El servidor Apache2 una de las implementaciones de HTTP de mayor difusión, ya está instalado en las máquinas Ubuntu 20.04 y se ejecutará en modo *standalone*. Con el siguiente comando podemos gestionar y comprobar el estado del servicio (iniciar, parar, reiniciar, re-ejecutar sin perder conexiones, deshabilitar o habilitar el inicio automático):

```
# sudo systemctl [start|stop|restart|reload|disable|enable] apache2
```

En un despliegue real en explotación lo habitual sería tener cualquier tipo de *firewall* instalado. En ese caso, habría que incluir una regla que habilite el acceso a los puertos correspondientes (dependiendo si queremos facilitar el tráfico sin cifrar directamente sobre TCP en el puerto 80, o tráfico cifrado en el puerto 443 usando el protocolo seguro TLS, o ambos).

Para comprobar la correcta instalación y funcionamiento del servidor, después de iniciar este, acced a la URL `http://localhost` o `http://direccion_ip` desde cualquier navegador usando cualquiera de las direcciones IP locales disponibles. La página web que aparecerá será la que se define por defecto durante la instalación de Apache2.

La configuración de Apache2 se lleva a cabo principalmente mediante la edición de una serie de ficheros de texto. Los más relevantes son:

- `/var/www/html`: Aquí se almacena el contenido del sitio web. Se puede modificar en los ficheros de configuración.
- `/etc/apache2/apache2.conf`: Configuración principal de Apache2. Las diferentes características y parámetros se definen mediante directivas.
- `/etc/apache2/ports.conf` : Puertos en los que Apache2 atenderá solicitudes.
- `/etc/apache2/sites-available/`: Directorio donde almacenan los *hosts* virtuales. Ver Sección 1.2.4 para más detalles.
- `/etc/apache2/sites-enabled/`: Directorio donde se almacenan los *hosts* virtuales.
- `/etc/apache2/mods-available/` y `/etc/apache2/mods-enabled/`: Contienen los módulos disponibles y habilitados, respectivamente. Por ejemplo para dar soporte a transacciones MySQL o PHP.
- `/var/log/apache2/access.log`: Fichero de trazas donde se almacenan todas las transacciones.
- `/var/log/apache2/error.log`: Fichero donde se registran los errores.

Para obtener más ayuda sobre la configuración de Apache2 y las directivas disponibles, abra el navegador web y visite la dirección:

```
http://httpd.apache.org/docs/2.4/
```

1.2.4 Configuración de Virtual Hosts

Un servidor Apache2 puede dar servicio a varios sitios webs (con diferentes nombres de dominio) simultáneamente como si fueran funcionalmente diferentes, aunque en realidad es un único proceso el que los sirve. A cada uno de estos sitios se les denomina *host* virtual.

Cada *host* virtual puede tener su propio directorio raíz, su propia política de seguridad y en definitiva, su propia configuración.

La configuración comienza creando un directorio raíz diferente para cada uno de los sitios web o *hosts* virtuales a configurar:

```
# sudo mkdir -p /var/www/midominio1.com/public_html
# sudo mkdir -p /var/www/midominio2.com/public_html
```

Dentro de cada dominio debe crearse un fichero `index.html` como por ejemplo:

```
<!DOCTYPE html>
  <head>
    <meta charset="utf-8">Mi dominio 1</title>
  </head>
  <body>
  <h1>Home de mi dominio 1</h1>
  </body>
</html>
```

Es necesario cambiar el propietario de los directorios raíz y de sus ficheros al usuario de apache2 `www-data`, por ejemplo:

```
# chown -R www-data: /var/www/ midominio1.com
```

En el directorio `/etc/apache2/sites-available` editamos los ficheros de configuración de los diferentes *hosts* virtuales a crear. Es habitual nombrar los ficheros con el correspondiente nombre de dominio, por ejemplo `midominio1.com.conf`

```
<VirtualHost *:80>
   ServerName midominio1.com
   ServerAlias www.midominio1.com
   ServerAdmin webmaster@midominio1.com
   DocumentRoot /var/www/midominio1.com/public_html

   <Directory /var/www/midominio1.com/public_html>
       Options -Indexes +FollowSymLinks
       AllowOverride All
   </Directory>
   ErrorLog ${APACHE_LOG_DIR}/midominio1.com-error.log
   CustomLog ${APACHE_LOG_DIR}/midominio1.com-access.log combined
</VirtualHost>
```


Para obtener más ayuda sobre la configuración de Apache2 y las directivas disponibles, abra el navegador web y visite la dirección:

```
http://httpd.apache.org/docs/2.4/vhosts/
```

Una vez creados los ficheros de configuración, hay que crear enlaces simbólicos desde el directorio `sites-available` al directorio `sites-enabled`. Esto se puede hacer directamente con el comando `ln`, o bien usando un *script* especialmente preparado en la distribución de apache2:

```
# sudo a2ensite midominio1.com
```

Comprobamos si la configuración tiene algún error:

```
# sudo apachectl configtest
```

La creación de *hosts* virtuales no implica que se creen automáticamente las entradas para que DNS sepa resolver adecuadamente los nombres de los dominios creados. Para ello, habría que configurar adecuadamente DNS para los nombres de dominios virtuales. No obstante, para hacer pruebas, esto se puede hacer localmente editando el fichero `/etc/hosts`, añadiendo los nombres de dominios virtuales creados a una de las IP locales, por ejemplo la IP de la interfaz de datos. Una vez hecho esto, reiniciamos el servicio:

```
# sudo systemctl restart apache2
```

Y finalmente, desde un navegador comprobamos que los diferentes dominios creados son servidos sin problema y de acuerdo con la configuración deseada.

1.2.5 Restricción de accesos

En muchas ocasiones es conveniente restringir el acceso ciertas zonas o directorios del sitio web con algún nivel de protección. Apache2 proporciona funcionalidad para ello. En este ejercicio aprenderemos a restringir el acceso al directorio http://localhost/mi_zona_restringida.

En primer lugar hay que crear el directorio `/var/www/html/mi_zona_restringida`

Las directivas para restringir accesos, se pueden definir en el fichero de configuración principal del servidor (típicamente en la sección `<Directory>` de `httpd.conf`), o en cada uno de directorios a restringir mediante ficheros `.htaccess`

Si se hace con `.htaccess`, en el servidor hay que permitir que se puedan definir directivas de autenticación en estos ficheros. Esto se hace con la directiva `AllowOverride`, la cual especifica qué directivas pueden ser definidas en el fichero de configuración del directorio. Por ejemplo, si queremos permitir la configuración de directivas de autenticación con `.htaccess`, modificaremos el fichero `/etc/apache2/apache2.conf` tal y como sigue:

```
AllowOverride All
```

Además, es necesario crear un fichero de contraseñas, para ello usaremos la utilidad `htpasswd` con la siguiente sintaxis, en la que la opción `-c` es para crear por primera vez el fichero si no existe:

```
# sudo htpasswd -c /usr/local/apache/passwd/passwords miusuario
```

Una vez hecho esto, el siguiente paso se puede realizar editando el fichero `.htaccess` ubicado en `/var/www/html/mi_zona_restringida` como se muestra a continuación

```
AuthType Basic
AuthName "Directorio con control de acceso"
# (las siguientes directivas son opcionales)
AuthUserFile "/usr/local/apache/passwd/passwords"
Require user miusuario
```

`AuthType` define la metodología de autenticación. Nótese que `Basic` implica un mecanismo de identificación más que de autenticación, ya que `Basic` consiste en enviar un *login* y un *password* en texto plano desde el cliente, procedimiento este vulnerable a ataques de repetición. Para evitar esta vulnerabilidad, es recomendable usar el módulo `mod_ssl` que encripta toda la transacción.
`AuthName` define un nombre para la zona de seguridad. Una vez que nos hayamos autenticado en esta zona, el cliente reintentará automáticamente las mismas credenciales en todas las zonas protegidas con el mismo nombre en este servidor.
`AuthUserFile` define el fichero de contraseñas.

`Require` define al usuario al que se le permite el acceso. Alternativamente se puede usar `Require valid-user` lo que implicaría que se permite el acceso genérico a cualquier usuario definido en el fichero de contraseñas.

Finalmente, para que el proceso se reconfigure leyendo las nuevas directivas, tras la edición realizada es necesario re-arrancar el proceso mediante la utilidad `systemctl`.

1.3 Realización práctica

1) Habilite el servicio *http* en su equipo servidor. Abra un navegador web y pruebe a visitar la página de inicio desde dicho equipo (`http://localhost` o `http://127.0.0.1`). Modifique el contenido de la página de inicio, y compruebe que la dirección de su servidor es accesible.
2) Inspeccione el fichero `apache2.conf` e identifique las directivas más relevantes.
3) Cree 2 hosts virtuales con páginas de inicio diferentes y compruebe que son servidos convenientemente ante peticiones desde el cliente.
4) Cree una página de acceso restringido (es decir, que requiera usuario y contraseña antes de mostrarla) en *http://<IP o nombre del servidor>/restringida/*. Utilice como credenciales de acceso el usuario *admin* y la contraseña *1234*.