Qué

Se ha de instalar un servidor web que implemente HTTPS.

También se tienen que crear subdominios con sus respectivas páginas web para que los clientes pueden hacer sus propias páginas dentro de nuestro dominio, que será 2daw.iesmariaenriquez.es. Para lo que se usará un servidor web en tándem con un DNS y un servidor FTP.

Todos los sitios web de los diferentes subdominios habrán de estar configurados en el mismo servidor web.

Además, habrá una carpeta compartida entre todos los sitios web.

Para qué

El objetivo es crear ISP que de a sus clientes un subdominio propio, con una página asegurada con HTTPS propia y una forma de modificar dicha página a su gusto.

Cómo

El servidor que se instalara es `apache2`, también conocido como `httpd`. Se ha elegido este porque de entrada soporta todo lo que hace falta para llevar a cabo este proyecto. Además, es uno de los dos servidores web más usados y, a pesar que Nginx es más popular, Apache tiene gran historial de recursos de ayuda al ser un proyecto bastante más maduro.

Para la funcionalidad que no tiene que ver con el servicio web se usarán los dos servidores ya configurados en las prácticas anteriores: Bind y ProFTPd.

Preparación

Se tienen que preparar unas pocas cosas antes instalar el servidor web. Se asume que se tiene configurado el servidor DNS tal y como se indica en mi práctica de DNS. Del mismo modo, se asume que el servidor FTP está configurado igual que en mi práctica de FTP.

A estos dos se les añaden cosas a lo largo de este proyecto, que se mencionarán más adelante.

`/etc/resolv.conf`

Para poder comprobar correctamente el funcionamiento de la página, hace falta configurar que servidor DNS se va a usar en el cliente.

En Linux, eso se hace desde `/etc/resolv.conf`, pero en el caso de Ubuntu, el resolv.conf es un enlace a un archivo que se reestablece periódicamente. En el mismo directorio que el objetivo de este enlace, hay otro archivo que si conserva los cambios que se le hacen, `/run/systemd/resolve/resolv.conf`

Para sobreescribir este enlace, se ejecuta este comando:

```
sudo ln -sf /etc/resolv.conf /run/systemd/resolve/resolv.conf
```

Y por último, se añade `nameserver <ip>` justo después de los comentarios, dónde <ip> sea la ip del servidor web.

> *Para más información, ve al apartado 'Modificar permanentemente el resolv.conf' de mi práctica de DNS*

Navegadores y caché

Un problema que puede surgir a lo largo de esta práctica es que el resultado parezca ser el mismo tras un cambio en la configuración. Esto se debe que los navegadores modernos guardan archivos en caché para agilizar próximas peticiones a la misma página.

Hay tres opciones para mitigar esto:

- Borrar la caché del navegador periódicamente.
- Hacer que la caché del navegador se borre cada vez que se cierre.
- Usar un programa como `links` o `httpie` para comprobar los resultados; son herramientas de CLI que hacen peticiones GET sin guardar ningún tipo de información.

Instalación y puesta a punto

Para instalar el servidor web Apache en Ubuntu se ejecuta el comando `sudo apt install apache2 -y`.

Tras comprobar que el servicio (cuyo nombre es `apache2.service`) se ha habilitado correctamente, se hace copia de seguridad del archivo de configuración global:

```
sudo cp /etc/apache2/apache2.conf{,.bck}
```

En este punto todo lo que se tiene es un servidor web con un sitio web por defecto. Dicho sitio está definido en `/etc/apache/sites-enabled/000-default.conf`. Este archivo en realidad es un enlace a un archivo localizado en `/etc/apache/sites-available/`, que es donde se configurarán los sitios web.

Para crear un nuevo sitio web se copia el archivo `000-default.conf` de este directorio en el mismo directorio. Para los própositos de esta práctica se harán 3 copias:

```
sudo cp /etc/apache2/sites-available/{000-default.conf,000-inicio.conf,clientes.conf,ftp.conf}
```

Estos tres archivos habilitarán un sitio web para cada uno de los siguientes dominios (que se tendrán que crear por separado):

- `000-inicio.conf`: 2daw.iesmariaenriquez.es y www.2daw.iesmariaenriquez.es.
- `clientes.conf`: <cliente>.2daw.iesmariaenriquez.es.
- `ftp.conf`: ftp.2daw.iesmariaenriquez.es

En cada uno de estos archivos se tiene que poner una etiqueta `<VirtualHost>`. En cada uno de ellos hará falta lo siguiente:

- Como atributo, dominio y puerto al que sirve (Ej.: `<VirtualHost dominio.es:80>`)
- La directiva `ServerName`, que indica el nombre de dominio que corresponde al sitio web. La diferencia entre esto y lo anterior es que esto es **a qué dominio sirve** y lo anterior es **por dónde escucha**.
- La directiva `DocumentRoot`, que indica dónde está el directorio en el que se ubican las páginas web del sitio. Es necesario que todo el mundo tenga permisos de lectura y ejecución en este directorio, que por defecto y por norma general estará ubicado en `/var/www`.

Aparte de eso, cada uno de los sitios web necesitará de directivas adicionales, pero eso se verá más adelante.

En el caso de clientes.conf, se han puesto múltiples `VirtualHost` que corresponderán a diferentes clientes.

Tras crear los directorios, las páginas y los `VirtualHost`, se tienen que habilitar las páginas. Se puede hacer un enlace simbólico a `sites-enables` manualmente, o se puede usar la herramienta `a2ensite`, incluida con `apache2`:

```
sudo a2ensite ftp.conf
sudo a2ensite 000-inicio.conf
sudo a2ensite clientes.conf
```

Además, se dehabilita la página por defecto, pues no se va a usar más:

```
sudo a2dissite 000-default.conf
```

Finalmente, se reinicia el servicio y se comprueba que funcione todo en el navegador.

Habilitar HTTPS

Para habilitar HTTPS primero hay que habilitar el módulo de SSL de Apache:

```
sudo a2enmod ssl
```

Después se crea un certificado con `openssl`. En este caso se han copiado y renombrado el certificado y su clave usados en ProFTPd al directorio `/etc/apache2/ssl`, tras crearlo.

Para crear un certificado nuevo, se ejecutan estos comandos:

```
sudo openssl genrsa -out /etc/ssl/private/proftpd.key 4096

sudo openssl req -new -x509 -days 1460 -key /etc/ssl/private/proftpd.key \
-out /etc/ssl/certs/proftpd.crt
```

Cuando se tienen ya el módulo de SSL y los certificados, se añaden las siguientes directivas al archivo de configuración global, `apache2.conf`:

```
SSLEngine on
SSLCertificateFile  /etc/apache2/ssl/apache2.crt
SSLCertificateKeyFile   /etc/apache2/ssl/apache2.key
```

Estas directivas activan el módulo de SSL y especifican dónde están el certificado y su clave.

Además, se cambia el puerto indicado en los `VirtualHost` de las páginas a 443, el puerto por defecto de HTTPS.

Por último, se reinicia el servicio y se comprueba que todo funcione de nuevo en el navegador. Ahora en la URL se tendrá que especificar el protocolo HTTPS, pues este servidor ya no sirve sitios web por HTTP.

Redirigir de HTTP a HTTPS

El primer paso es habilitar otro módulo diferente de apache, Rewrite que permite redirigir a sitios diferentes mediante expresiones regulares y variables:

```
sudo a2enmod rewrite
```

Luego se crea un nuevo fichero en `/etc/apache2/sites-available/`, en este caso llamado `http-to-https.conf`, y se añade un `VirtualHost` que escuche y sirva por el mismo dominio que la página inicial del dominio deseado (2daw.iesmariaenriquez.es, en este proyecto), pero que escuche por el puerto 80.

En él añadirán las siguientes directivas (Se elimina `DocumentRoot`, pues no queremos servir una página, pero se conserva `ServerName`):

```
ServerAlias *.2daw.iesmariaenriquez.es
SSLEngine off

RewriteEngine On
RewriteCond %{HTTPS} off
RewriteRule (.*) https://%{HTTP_HOST}%{REQUEST_URI}
```

Estas directivas cumplen lo siguiente:

- Sirven a todos los subdominios del dominio 2daw.iesmariaenriquez.es por HTTP
- Redirigen las conexiones que no sean HTTPS a una URI con el mismo dominio y pidiendo el mismo recurso, pero especificando el protocolo HTTPS.

Tras esto, se reinicia el servicio y se comprueba el funcionamiento en el navegador.

Sitios web y dominios

Estos son los sitios web que se han definido en el el servidor web.

www.2daw.iesmariaenriquez.es

El archivo de configuración de este sitio web es `000-inicio.conf`. Muestra una lista con las páginas de todos los clientes usando PHP.

Se le ha añadido *000-* al nombre para que vaya alfabéticamente antes que el resto de sitios web. Si no se hace esto, si hay sitios que van antes que esta que correspondan a un subdominio del dominio del sitio web de inicio, se podría servir esa en lugar de la de inicio (como sería el caso de `clientes.conf`)

```
<IfModule mod_ssl.c>
    <VirtualHost 2daw.iesmariaenriquez.es:443>
        ServerName 2daw.iesmariaenriquez.es
        ServerAlias www.2daw.iesmariaenriquez.es
        DocumentRoot /var/www/inicio
    </VirtualHost>
</IfModule>
```

Esto es lo que se consigue con este fichero de configuración:

- Sólo se sirve el sitio web si el módulo HTTPS está habilitado.
- El `VirtualHost` y `ServerName` hacen referencia a 2daw.iesmariaenriquez.es, pero se usa la directiva `ServerAlias` para servir también a www.2daw.iesmariaenriquez.es.

ftp.2daw.iesmariaenriquez.es

El archivo de configuración de este sitio web es `ftp.conf`. Muestra instrucciones de como acceder por SFTP al servidor.

```
<IfModule mod_ssl.c>
    <VirtualHost ftp.2daw.iesmariaenriquez.es:443>
        ServerName ftp.2daw.iesmariaenriquez.es
        DocumentRoot /var/www/ftp
    </VirtualHost>
</IfModule>
```

De nuevo, este sitio web sólo se habilita si se puede usar HTTPS.

<cliente>.2daw.iesmariaenriquez.es

El archivo de configuración de este sitio web es `clientes.conf`. Este contiene múltiples `VirtualHost`, cada uno referenciando a un subdominio de 2daw.iesmariaenriquez.es diferente, cada uno de los cuales hace referencia a un cliente y páginas web diferentes.

Las páginas de los sitios web de cada cliente están todas en directorios separados dentro de `/var/www/cliente/`, por lo que ninguna no se puede ver el sitio web de un usuario desde otro sitio web diferente.

Además, todos los sitios web de los clientes usan la directiva `Alias` poder ver la carpeta compartida que se encuentra también en `/var/www/cliente/`, que no podrían ver de otro modo.

Esta es la plantilla que se usa para crear los sitios web de los clientes en este proyecto:

```
<VirtualHost user.2daw.iesmariaenriquez.es:443>
    ServerName user.2daw.iesmariaenriquez.es
    DocumentRoot /var/www/cliente/user
    Alias "/compartido" "/var/www/cliente/compartido"
</VirtualHost>
```

FTP y DNS

También son necesarios un servidor FTP y un servidor DNS. Como se ha mencionado anteriormente, las configuraciones anteriores de ProFTPd y Bind han sido recicladas, pero falta mencionar un par de cosas con respecto a estos dos servicios:

Usuarios virtuales y carpeta compartida en ProFTPd.

Para que los clientes puedan modificar su página web se ha de habilitar un usuario virtual, que tenga como directorio personal la carpeta en la que se aloja el sitio web. Además, el usuario tiene que ser el propietario de esta carpeta.

Además, la carpeta compartida que se ha configurado para Apache no es visible desde FTP y, como la raíz del usuario virtual es su directorio personal, un simple enlace no funciona.

Para hacer disponible la carpeta compartida al usuario virtual, se han hecho puntos de montaje a cada uno de los directorios de los usuarios virtuales, modificando además `/etc/fstab` para que se haga automáticamente al iniciar el sistema.

Se puede hacer de este modo:

```
sudo mount --bind /var/www/cliente/compartido/ /var/www/cliente/user/compartido/
echo "/var/www/cliente/compartido/ /var/www/cliente/user/compartido/ none bind 0 0" |
sudo tee -a /etc/fstab
```

Los usuarios deberían, por supuesto, tener permisos suficientes como para ver el contenido, pero no como para modificarlo.

Esto consigue poniéndoles a todos un grupo común y dándole a ese grupo permisos de lectura y ejecución en el directorio, pero no de escritura.

> *Para crear a los usuarios y cambiar los permisos, ve el apartado de 'Creación de usuarios virtuales en ProFTPd' de mi práctica de FTP.*

La directiva `Alias` de Apache no sería necesaria con esto, pero conviene dejarla para poder seguir acciendo a los recursos compartidos en la web si en algún momento momento fallase el punto de montaje.

Subdominios *www*, *ftp* y para los clientes.

Se han creado poniéndolos como *Canonical names* del dominio 2daw.iesmariaenriquez.es, añadiendo algo como esto en el archivo `/etc/bind/2daw.iesmariaenriquez.es.zone`:

```
user    IN  CNAME   2daw.iesmariaenriquez.es.
```

Misceláneo

Script para dar de alta a clientes

Se ha hecho un script, `web_user.sh`, que crea automáticamente al usuario virtual de FTP, el registro DNS del usuario, el `VirtualHost` correspondiente a ese cliente, el punto de montaje de la carpeta compartida, crea una página web por defecto y establece los permisos adecuados a su directorio personal.

Su funcionamiento depende de que se haya puesto una cadena de caracteres concreta (Por defecto `<===>`) en forma de comentario en los archivos a modificar, tras la cual insertará las líneas necesarias.

Se puede modificar fácilmente a través de variables:

- El grupo que comparten los usuarios virtuales.
- El directorio personal de los usuarios virtuales.
- El directorio compartido.
- El *string* tras el cual insertar las líneas.
- El dominio.
- El archivo de configuración del sitio web.
- El archivo de registros DNS.

Explicación de las Directivas usadas en `apache2.conf`

Hay muchas directivas que se han usado en `apache2.conf` pero no se ham mencionado porque vienen por defecto y está bien no tocarlas o porque no son menester para cumplir los objetivos de este proyecto.

Algunas de ellas tienen como parámetro una variable de entorno (`${VARIABLE}`). Estas están definidas en `/etc/apache2/envvars`. Es ideal mantener estos parámetros como variables de entorno para evitar inestabilidad o fallos en diferentes servidores que usen usuarios, directorios, etc.

De todos modos, conviene hacer una copia de seguridad de `envvars` por si algún servidor careciese de este archivo.

Para hacer la copia se ejecuta esto:

```
sudo cp /etc/apache2/envvars{,.bck}
```

Y para restaurarla esto:

```
sudo cp /etc/apache2/envvars{.bck,}
```

Directivas globales

- **`DefaultRuntimeDir ${APACHE_RUN_DIR}`**: Directorio en el que sea crean archivos necesarios para la ejecución de `apache2`.
- **`PidFile ${APACHE_PID_FILE}`**: Para retrocompatibilidad con `sysvinit`.
- **`SSLEngine on`**: Habilita HTTPS.
- **`SSLCertificateFile /etc/apache2/ssl/apache2.crt`**: Certificado SSL que se usará en las conexiones seguras.
- **`SSLCertificateKeyFile  /etc/apache2/ssl/apache2.key`**: La clave del certificado SSL.
- **`User ${APACHE_RUN_USER}`**: El usuario propietario del proceso del servicio.
- **`Group ${APACHE_RUN_GROUP}`**: El grupo propietario del proceso del servicio.
- **`ErrorLog ${APACHE_LOG_DIR}/error.log`**: Nombre del archivo en el que se guardan los registros de error
- **`IncludeOptional mods-enabled/*.load`**: Importa los módulos de apache en formato `load`, si existen.

- **`IncludeOptional mods-enabled/*.conf`**: Importa los módulos de apache en formato `conf`, si existen.
- **`Include ports.conf`**: Importa el archivo en el que se especifican por qué puertos escucha apache.
- **`IncludeOptional conf-enabled/*.conf`**: Importa archivos de configuración global habilitados, si existen.
- **`IncludeOptional sites-enabled/*.conf`**: Importa archivos de configuración los sitios web habilitados, si existen.
- **`LogFormat`**: Establece el formato que tienen los diferentes tipos de registro. Conviene no tocarlo para obtener la máxima información de ellos de forma clara, a no ser que se sepa lo que se está haciendo y haga falta hacer modificiaciones.

Directivas de directorios y archivos

- Deniega el acceso a todas los directorios y subdirectorios de la raíz. Además, el usuario no puede sobreescribir los permisos usando un archivo de acceso propio.

```
<Directory />
    Options FollowSymLinks
    AllowOverride None
    Require all denied
</Directory>
```

- Permite el acceso a los recursos del directorio de recursos compartidos para usuarios del sistema.

```
<Directory /usr/share>
    AllowOverride None
    Require all granted
</Directory>
```

- Permite el acceso a todos los directorios y subdirectorios de la carpeta en la que se almacenan las páginas web. También permite seguir los enlaces simbólicos. Por defecto tenía la opción `Indexes` además de `FollowSymLinks`, que se ha borrado para no listar todos los archivos del en caso de que no haya una página web.

```
<Directory /var/www/>
    Options FollowSymLinks
    AllowOverride None
    Require all granted
</Directory>
```

- Habilita como archivo de acceso, un archivo que puede usar el cliente para establecer sus propias restricciones en la página web, `.htaccess` y lo hace inaccesible desde el sitio web.

```
AccessFileName .htaccess
<FilesMatch "^\.htaccess">
    Require all denied
</FilesMatch>
```

Conclusión

Esta práctica ha sido mucho más sencilla que las anteriores, y está vez si se podían encontrar muchas de las soluciones en internet y en la documentación.

A pesar de esto, sigo sin tener la sensación de que haya aprendido demasiado durante la práctica.

En definitiva, considero que está práctica ha sido mucho más justa que las anteriores, pero sigo sin tener la impresión de que sean muy didácticas.