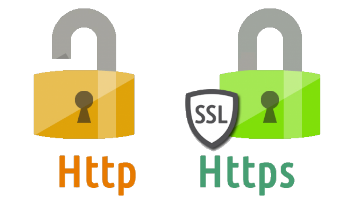
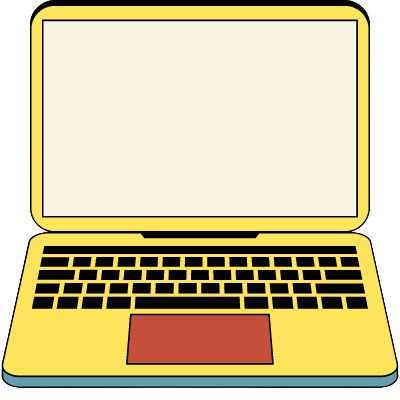
Instalar HTTPS en el servidor

**Christian Millán Soria**





- Curso: 2023-2024

- Clase: 2º DAW Tarde

- Profesor: David Hormigo Ramírez

- Módulo: Despliegue de Aplicaciones Web

- Fecha: 10/11/2023

**1. Regla de entrada** 3

**2. Certbot** 4

**2.1. SSH into the server** 6

**2.2. Install snapd** 6

**2.3. Remove certbot-auto and any Certbot OS packages** 6

**2.4. Install Certbot** 6

**2.5. Prepare the Certbot command** 7

**2.6. Choose how you’d like to run Certbot** 7

**2.6.1. Correo electrónico** 7

**2.6.2. Términos de servicio** 7

**2.6.3. Facilitar nuestro correo a Certbot** 7

**2.7. Comprobación de HTTPS** 8

**3. Securizar FTP** 10

**4. Módulo de protección Cross-Site Request** 12

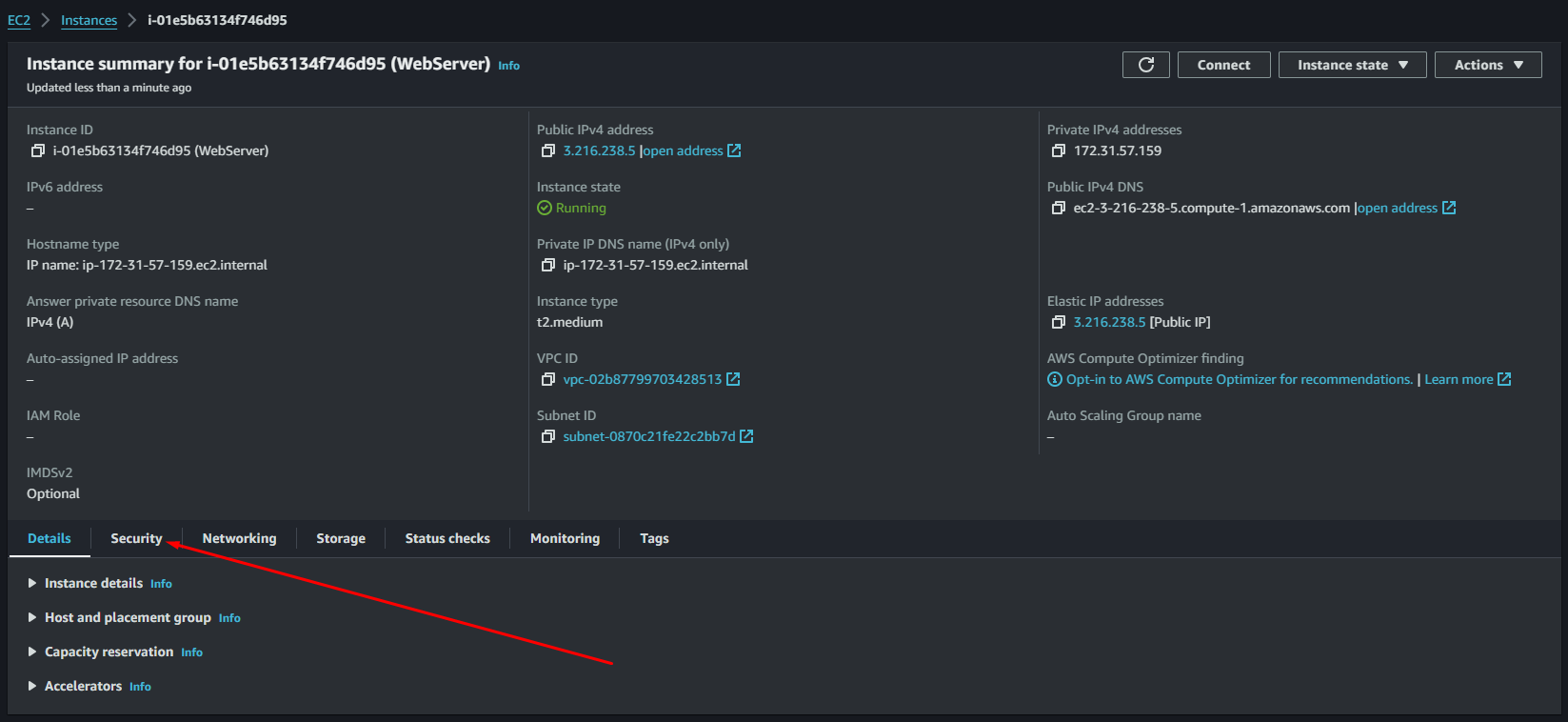
El objetivo de la práctica es conseguir un certificado que establece que la conexión con nuestro servidor es segura.

Vamos paso a paso:

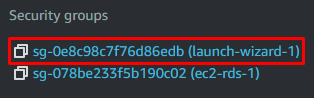
**1. Regla de entrada**

Dado que ahora sí que vamos a hacer uso del protocolo [**HTTPS🔗**](https://developer.mozilla.org/en-US/docs/Glossary/HTTPS), necesitamos primero habilitar la entrada de conexiones mediante este protocolo en nuestro servidor.

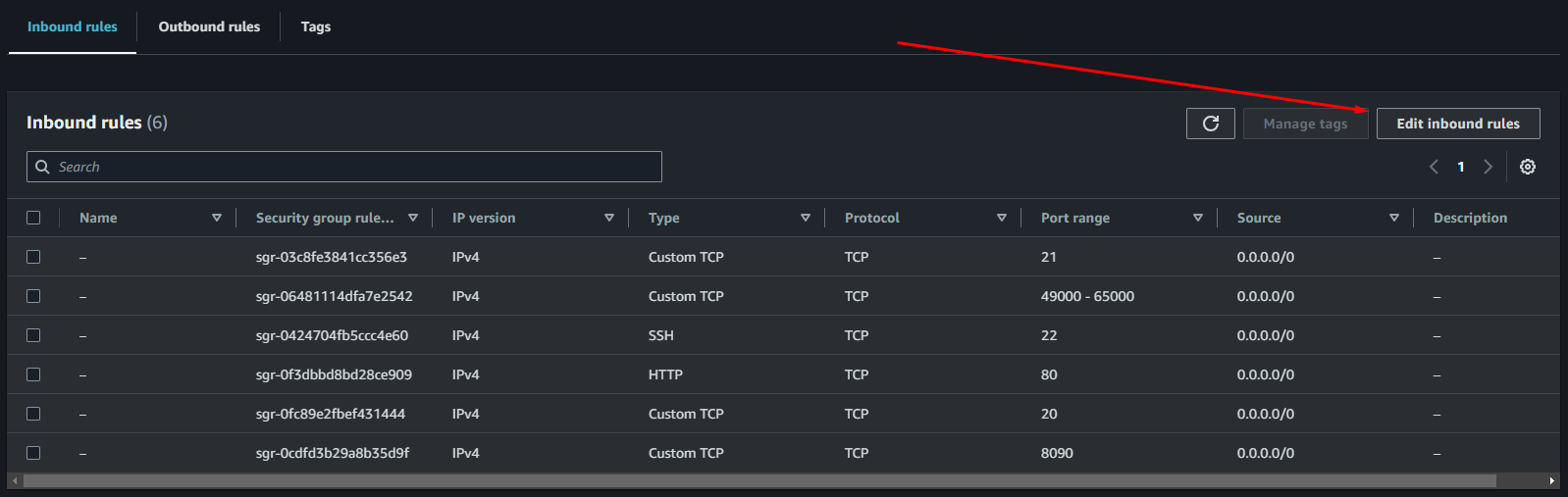
Para ello nos dirigimos a AWS, al resumen de nuestra instancia, y entramos en la pestaña de seguridad.



Aquí dentro, seleccionamos el grupo de seguridad propio de nuestra instancia.

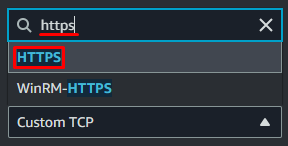


Entramos a editar las reglas de entrada de nuestra instancia.

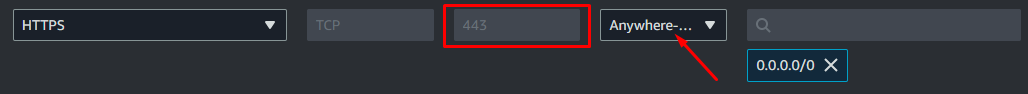


Una vez estamos en este punto, debemos hacer clic en el botón “**Add rule**” para crear una nueva regla de seguridad.

En el buscador que se nos presenta, escribimos “**https**” y seleccionamos la primera opción.



Por último, establecemos que las conexiones mediante el uso de este protocolo se podrán realizar desde cualquier IP del mundo.



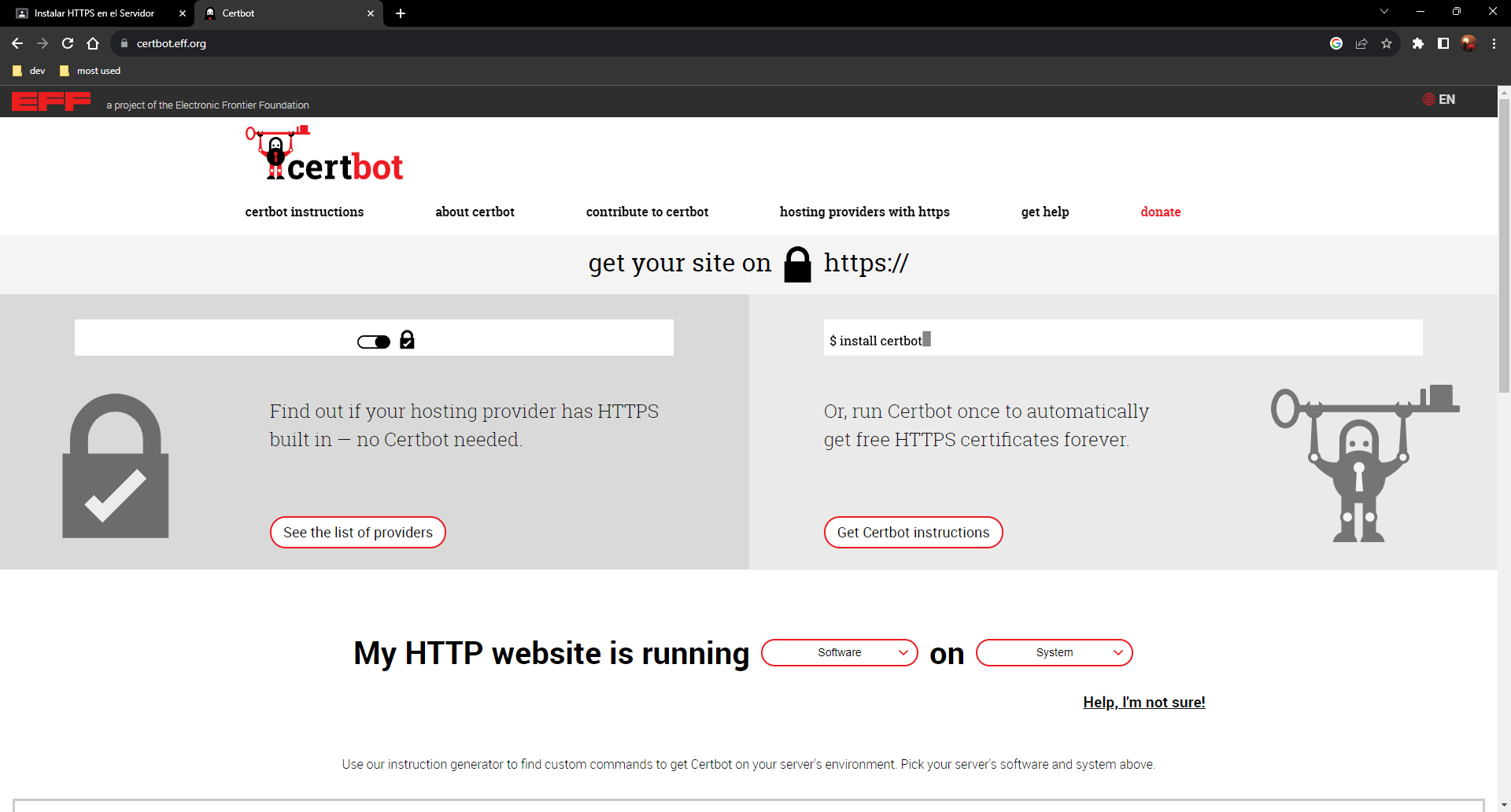
Guardamos los cambios.



**2. Certbot**

Como se ha mencionado antes, el objetivo es conseguir un certificado que autentifique las conexiones a nuestro servidor para poder utilizar HTTPS. Para poder llevar esto a cabo, y según se especifica en la documentación de la práctica, vamos a hacer uso de [**Certbot🔗**](https://certbot.eff.org/), una herramienta que ofrece certificados es este tipo de forma gratuita, aunque no es la más segura del mercado.

Nos dirigimos a la página web enlazada en el párrafo anterior.

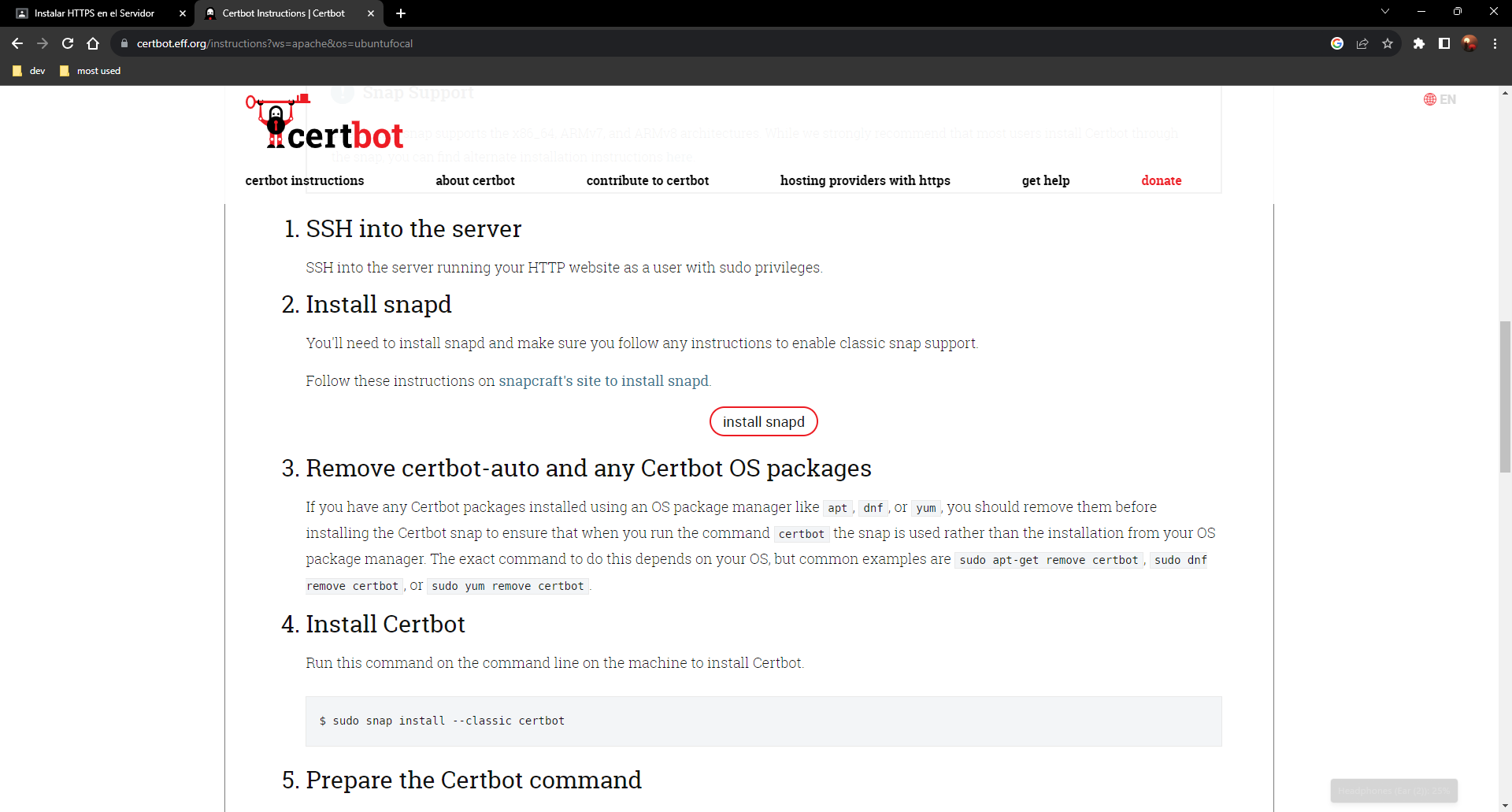


En los dos desplegables que aparecen en el centro de la pantalla, seleccionamos las opciones “**Apache**” y “**Ubuntu 20**”.



La web se actualiza para mostrarnos una guía de instalación de Certbot en nuestro servidor, con las especificaciones que le hemos dado.

Si bajamos un poco, podemos ver los pasos a seguir para conseguir el certificado.



Vamos a realizar los pasos necesarios.

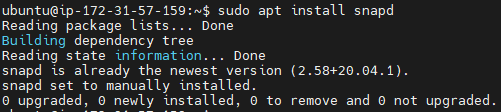
**2.1. SSH into the server**

Nos dirigimos a nuestro cliente SSH para conectarnos a nuestra instancia para ejecutar los comandos de esta guía.

**2.2. Install snapd**

El instalador que utilizaremos para la certificación es el conocido como [“**snapd**”🔗](https://snapcraft.io/docs/installing-snap-on-ubuntu), un instalador desarrollado por la propia [**Canonical🔗**](https://ubuntu.com/community/governance/canonical), creadora de Ubuntu.

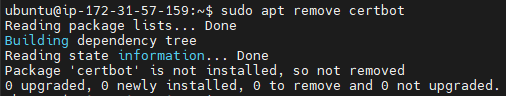
En principio, el instalador snapd viene instalado de forma predeterminada en el momento que se instala Ubuntu, pero para comprobar (e instalarlo si no lo está ya), ejecutamos el comando **sudo apt install snapd**.



**2.3. Remove certbot-auto and any Certbot OS packages**

La guía recomienda que hagamos una limpieza preventiva de cualquier posible instalación previa de software relacionado con Certbot, para que no haya ningún problema de compatibilidad o conflictos futuros por esta razón.

Utilizamos el comando **sudo apt remove certbot** para realizar esto.



**2.4. Install Certbot**

Ahora sí, habiendo hecho los pasos anteriores, podemos pasar a instalar Certbot en nuestra máquina.

**sudo snap install --classic certbot**



**2.5. Prepare the Certbot command**

Para asegurarnos de que el comando **certbot** puede ser utilizado de ahora en adelante vamos a utilizar el comando **sudo ln -s /snap/bin/certbot /usr/bin/certbot**.

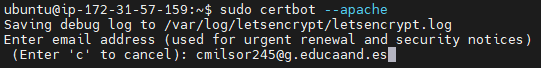
**2.6. Choose how you’d like to run Certbot**

Para obtener e instalar los certificados que necesitamos para esta práctica, vamos a usar el comando **sudo certbot --apache**, el cual se encargará de lo anterior mencionado.

Se nos piden una serie de acciones para llevar a cabo todo esto.

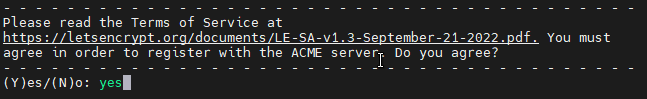
**2.6.1. Correo electrónico**

Se nos solicita un correo electrónico donde se nos enviará información sobre este servicio. En mi caso, he utilizado el correo corporativo.



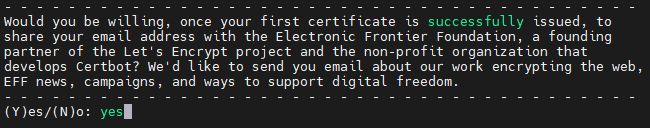
**2.6.2. Términos de servicio**

Debemos aceptar los términos de servicio de Certbot para seguir con la instalación.



**2.6.3. Facilitar nuestro correo a Certbot**

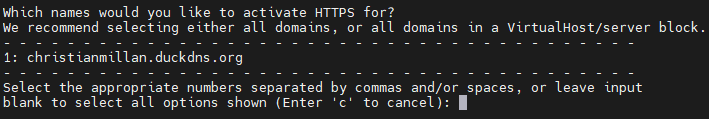
Podemos elegir si compartir o no nuestro correo electrónico introducido con Certbot para que puedan enviarnos más información sobre posibles herramientas que nos puedan interesar y demás. En mi caso, he aceptado esto.



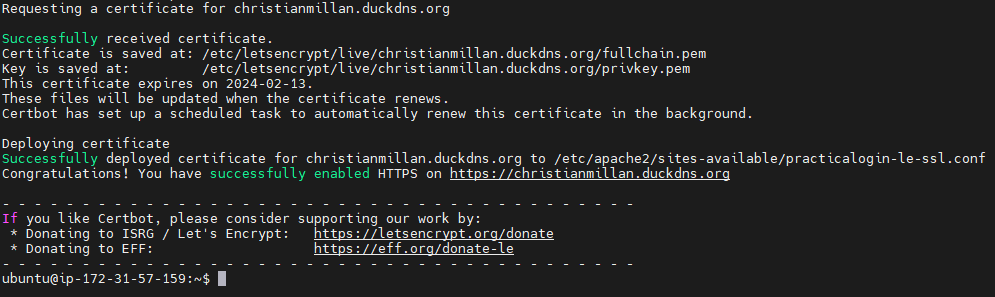
**2.6.4. Sitios a registrar**

Si no hemos hecho ningún cambio desde **la anterior práctica🔗**, deberíamos tener un único sitio activo en nuestro servidor, el cual corresponde a nuestro dominio DNS creado en Duck DNS para que apunte directamente a nuestro sitio de login básico con base de datos y local.

Aquí se nos pide que especifiquemos qué sitios queremos registrar, a lo que debemos responder con un clic en la tecla **Enter** sin escribir nada en el input.

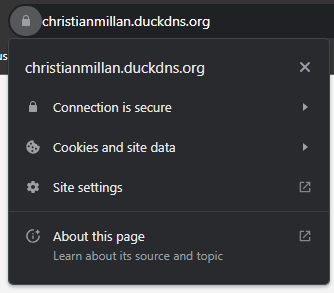


Una vez hecho todo esto, hemos conseguido un certificado para nuestro sitio de login.



**2.7. Comprobación de HTTPS**

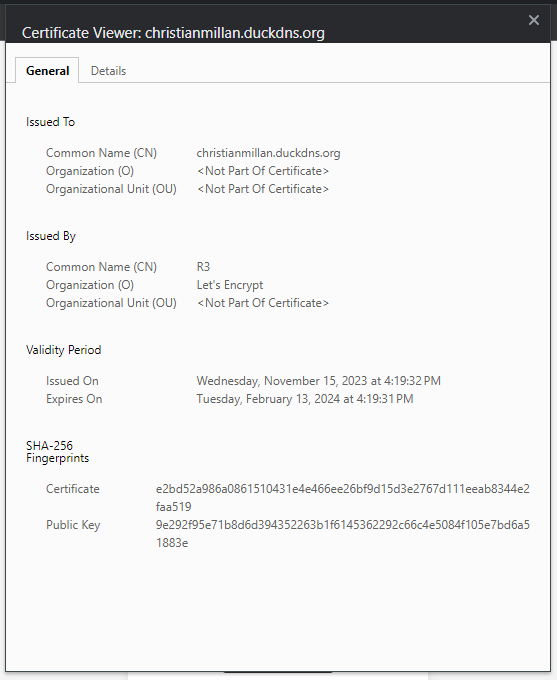
Una vez terminados todos los pasos anteriores, podemos volver a visitar nuestro sitio en un navegador web y comprobar que ahora aparece un icono que señaliza que la conexión es segura.



Adicionalmente, si hacemos clic en la barra de búsqueda podemos ver que se hace uso del protocolo HTTPS.



Tenemos disponible también el certificado de seguridad.



**3. Securizar FTP**

Como vimos en [**la práctica de FTP🔗**](https://github.com/cmilsor245/daweb/blob/main/05-instalacion_servidor_ftp/Instalaci%C3%B3n%20de%20Servidor%20FTP.pdf), este método no es del todo seguro. Es por ello que existen otros protocolos más seguros y que FTP está mayoritariamente en desuso a día de hoy.

Podemos arreglar esto (más o menos) habilitando el conocido como [**SSL/TLS🔗**](https://www.cloudflare.com/learning/ssl/what-is-ssl/#:~:text=Started%20For%20Free-,What%20is%20SSL%3F,TLS%20has%20%22HTTPS%22%20in%20its%20URL%20instead%20of%20%22HTTP.%22,-How%20does%20SSL), que añadirá esa capa extra de protección al servidor.

Lo primero que vamos a hacer es comprobar que el módulo para TLS está habilitado en nuestro servidor FTP.

Ejecutamos el comando **sudo nano /etc/proftpd/modules.conf** para entrar a editar el archivo donde se hayan los módulos del servidor.

Buscamos la línea “**LoadModule mod\_tls.c**”. Quitamos el comentario si es que existe para que el servidor cargue el módulo.



Reiniciamos el servidor para aplicar los cambios.

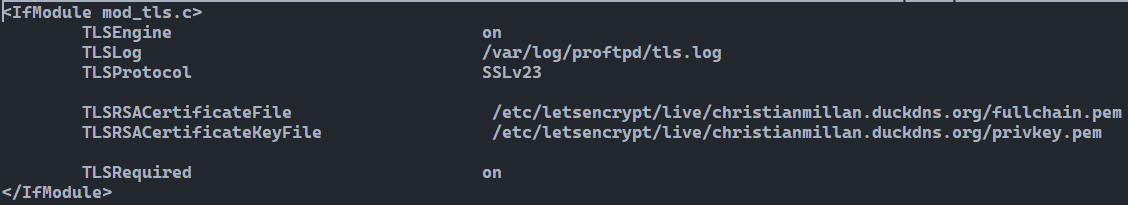
El siguiente paso es entrar a editar el archivo **proftpd.conf**. Aquí dentro buscaremos la directiva “**Include /etc/proftpd/tls.conf**” y descomentarla para que se cargue el archivo de configuración de TLS.

Adicionalmente, al final del archivo, agregamos la siguiente directiva que especifica que el servidor ProFTPD no requerirá la reutilización de sesiones TLS. Normalmente, cuando se establece una conexión TLS, **se pueden reutilizar sesiones anteriores para agilizar el proceso de conexión y reducir la sobrecarga asociada con la creación de una nueva sesión TLS**. Sin embargo, al utilizar **NoSessionReuseRequire**, se deshabilita esta reutilización obligatoria de sesiones previas.



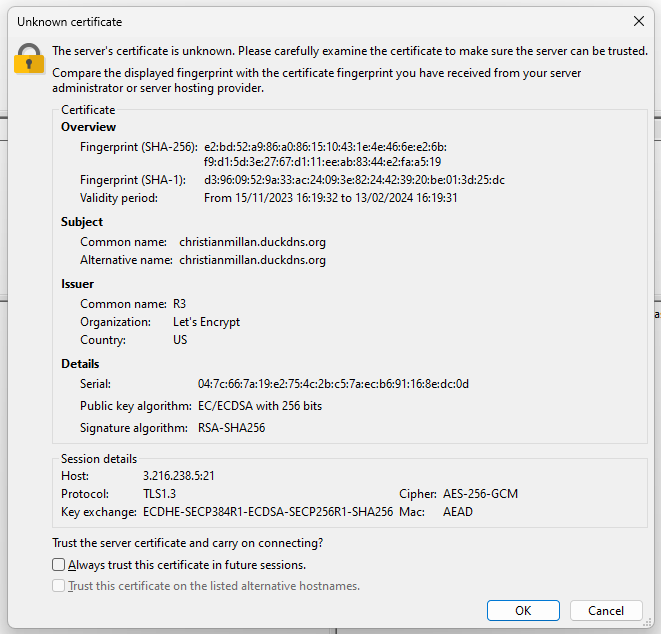
Por último, editamos dicho archivo con el comando **sudo nano /etc/proftpd/tls.conf**.

Dentro de la directiva “**<IfModule mod\_tls.c>**” ingresamos el siguiente código:

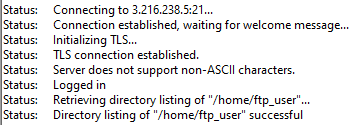


Reiniciamos el servidor para aplicar los cambios.

Ahora, una vez realicemos una conexión con el servidor FTP en FileZilla, por ejemplo, aparecerá el siguiente aviso:



Y aparece el siguiente mensaje en los logs del programa:



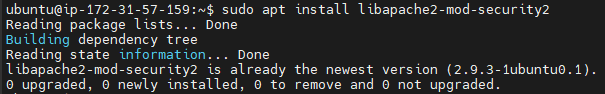
**4. Módulo de protección Cross-Site Request**

Como vimos en clase, el cifrado de nuestras contraseñas y demás información sensible está totalmente expuesta en nuestro servidor Apache.

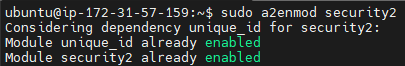
Para evitar esto, haremos uso del [**módulo CSRF**🔗](https://mod-csrf.sourceforge.net/) de Apache, el cual se encargará de que programas como [**WireShark**🔗](https://www.wireshark.org/) no puedan descifrar nuestras contraseñas, o al menos no de una forma tan sencilla.

Lo primero que debemos hacer es instalar el módulo, ya que (al menos en nuestra versión) en Apache no viene instalado de forma nativa.

Para ello, ejecutamos el comando **sudo apt install libapache2-mod-security2**.



Habilitamos el nuevo módulo en Apache con el comando **sudo a2enmod security2**.



Reiniciamos el servidor.



Para terminar, verificamos que el módulo está en funcionamiento con el siguiente comando: **sudo apache2ctl -M**.

Este comando devuelve una lista de todos los módulos de Apache que están instalados y habilitados. Si el módulo de CSRF está instalado y habilitado correctamente, el resultado del comando debería incluir la siguiente línea: **security2\_module (shared)**

