HTTPS con Let’s Encrypt, Docker y Docker Compose

Implantación de Aplicaciones Web

José Juan Sánchez Hernández

Curso 2023/2024

Índice

Índice de figuras

Índice de cuadros

1 HTTPS con Let's Encrypt, Docker y Docker Compose

En esta práctica vamos a habilitar el protocolo HTTPS en un sitio web PrestaShop que se estará ejecutando sobre contenedores Docker en una instancia EC2 de Amazon Web Services (AWS).

1.1 Conceptos básicos

1.1.1 ¿Qué es HTTPS?

HTTPS (*Hyptertext Transfer Protocol Secure*) o protocolo seguro de transferencia de hipertexto, es un protocolo de la capa de aplicación basado en el protocolo HTTP, destinado a la transferencia segura de datos de hipertexto. (Fuente: Wikipedia)

Para poder habilitar el protocolo HTTPS en un sitio web es necesario obtener un **certificado de seguridad**. Este certificado tiene que ser emitido por una **autoridad de certificación** (AC). En esta práctica vamos a obtener un certificado para un dominio de la Autoriidad de Certificación Let's Encrypt.

1.1.2 ¿Qué es Let's Encrypt?

Let's Encrypt es una autoridad de certificación que se puso en marcha el 12 de abril de 2016 y que proporciona certificados X.509 **gratuitos** para el cifrado de seguridad de nivel de transporte (TLS) a través de un proceso automatizado diseñado para eliminar el complejo proceso actual de creación manual, la validación, firma, instalación y renovación de los certificados de sitios web seguros. (Fuente: Wikipedia)

1.1.3 Cómo funciona Let's Encrypt

Se recomienda la lectura de la sección **Cómo Funciona Let's Encrypt** de la documentación oficial.

1.1.4 ¿Qué es el protocolo ACME?

Para poder obtener un certificado de Let's Encrypt para un dominio de un sitio web es necesario demostrar que se tiene control sobre ese dominio. Para realizar esta tarea es necesario utilizar un cliente del protocolo ACME (Automated Certificate Management Environment).

1.1.5 ¿Qué es HTTPS-PORTAL?

HTTPS-PORTAL es una imagen Docker que contiene un servidor HTTPS totalmente automatizado que hace uso de las tecnologías Nginx y Let's Enctrypt. Los certificados SSL se obtienen y renuevan de Let's Encrypt automáticamente.

Esta imagen está preparada para permitir que cualquier aplicación web pueda ejecutarse a través de HTTPS con una configuración muy sencilla.

Puede encontrar más información sobre HTTPS-PORTAL en la web oficial de Docker Hub.

1.1.6 Cómo usar HTTPS-PORTAL

Para usar la imagen HTTPS-PORTAL con Docker Compose sólo tenemos que crear un nuevo servicio en nuestro archivo `docker-compose.yml` que al menos incluya las siguientes opciones de configuración.

```
1 https-portal:
2   image: steveltn/https-portal:1
3   ports:
4     - 80:80
5     - 443:443
6   environment:
7     DOMAINS: 'practicahttps.ml -> http://prestashop:80'
8     #STAGE: 'production' # Don't use production until staging works
```

Este servicio será el único servicio del archivo `docker-compose.yml` que estará utilizando los puertos 80 y 443 de nuestra máquina.

En la variable `DOMAINS` tenemos que configurar el nombre de dominio público de nuestro sitio web y el nombre del servicio al que vamos a redireccionar todas las peticiones que se reciban por los puertos 80 y 443.

En el ejemplo anterior hemos configurado que todas las peticiones que se reciban en el dominio `practicahttps.ml` se van a reenviar al servicio `prestashop` que estará definido dentro del archivo `docker-compose.yml`.

La variable `STAGE` puede almacenar los siguientes valores:

- `local`: Crea un certificado autofirmado para hacer pruebas en local.
- `staging`: Solicita un **certificado de prueba a Let's Encrypt** para nuestro entorno de pruebas.
- `production`: Solicita un **certificado válido a Let's Encrypt**. Esta opción sólo la usaremos para poner nuestro sitio web en producción.

Si no se especifica ningún valor, la opción por defecto será `staging`.

1.1.7 Ejemplo de uso con HTTPS-PORTAL

A continuación, se muestra un ejemplo completo que utiliza HTTPS-PORTAL para habilitar HTTPS en un sitio web PrestaShop.

`docker-compose.yml`

```yaml
version: '3.4'

services:
  mysql:
    image: mysql
    command: --default-authentication-plugin=mysql_native_password
    ports:
      - 3306:3306
    environment:
      - MYSQL_ROOT_PASSWORD=${MYSQL_ROOT_PASSWORD}
      - MYSQL_DATABASE=${MYSQL_DATABASE}
      - MYSQL_USER=${MYSQL_USER}
      - MYSQL_PASSWORD=${MYSQL_PASSWORD}
    volumes:
      - mysql_data:/var/lib/mysql
    networks:
      - backend-network
    restart: always

  phpmyadmin:
    image: phpmyadmin
    ports:
      - 8080:80
    environment:
      - PMA_ARBITRARY=1
    networks:
      - backend-network
      - frontend-network
    restart: always
    depends_on:
      - mysql

  prestashop:
    image: prestashop/prestashop
    environment:
      - DB_SERVER=mysql
    volumes:
      - prestashop_data:/var/www/html
    networks:
      - backend-network
      - frontend-network
    restart: always
    depends_on:
      - mysql

  https-portal:
    image: steveltn/https-portal:1
    ports:
      - 80:80
      - 443:443
    restart: always
    environment:
      DOMAINS: 'practicahttps.ml -> http://prestashop:80'
      STAGE: 'production' # Don't use production until staging works
      # FORCE_RENEW: 'true'
    networks:
```

```
      - frontend-network

volumes:
  mysql_data:
  prestashop_data:

networks:
  backend-network:
  frontend-network:
```

.env

```
MYSQL_ROOT_PASSWORD=root
MYSQL_DATABASE=prestashop
MYSQL_USER=ps_user
MYSQL_PASSWORD=ps_password
```

1.2 Tareas a realizar

A continuación se describen **muy brevemente** algunas de las tareas que tendrá que realizar.

1.2.1 Paso 1

Crear una instancia EC2 en Amazon Web Services (AWS).

Cuando esté creando la instancia deberá **configurar los puertos** que estarán abiertos para poder conectarnos por SSH y para poder acceder por HTTP/HTTPS.

- SSH (22/TCP)
- HTTP (80/TCP)
- HTTPS (443/TCP)

1.2.2 Paso 2

Obtener la dirección IP pública de su instancia EC2 en AWS.

1.2.3 Paso 3

Registrar un nombre de dominio en algún proveedor de nombres de dominio gratuito. Por ejemplo, puede hacer uso de Freenom.

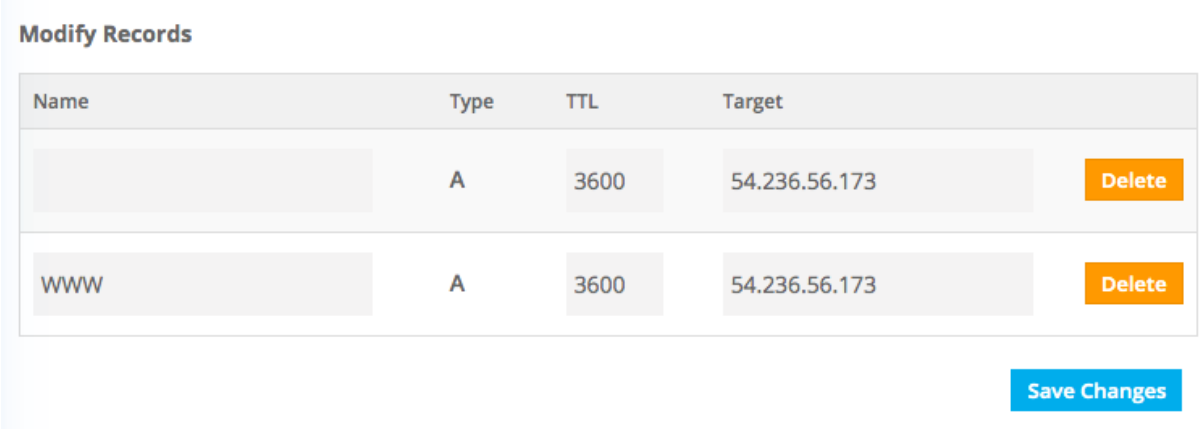
1.2.4 Paso 4

Configurar los registros DNS del proveedor de nombres de dominio para que el nombre de dominio de ha registrado pueda resolver hacia la dirección IP pública de su instancia EC2 de AWS.

Si utiliza el proveedor de nombres de dominio Freenom tendrá que acceder desde el panel de control, a la sección de sus dominios contratados y una vez allí seleccionar **Manage Freenom DNS**.

Tendrá que añadir dos registros DNS de tipo A con la dirección IP pública de su instancia EC2 de AWS. Un registro estará en blanco para que pueda resolver el nombre de dominio sin las www y el otro registro estará con las www
.

Ejemplo: En la siguiente imagen se muestra cómo sería la configuración de los registros DNS para resolver hacia la dirección IP 54.236.57.173.



Nota: Tenga en cuenta que una vez que ha realizado los cambios en el DNS habrá que esperar hasta que los cambios se propaguen. Puede hacer uso de la utilidad dnschecker.org para comprobar el estado de propagación de las DNS.

1.2.5 Paso 5

Realizar la instalación y configuración de Docker y Docker Compose en la instancia EC2 de AWS.

1.2.6 Paso 6

Modificar el archivo `docker-compose.yml` de alguna de las prácticas anteriores para incluir el servicio de HTTPS-PORTAL.

Una vez llegado a este punto, sólo queda desplegar los servicios con Docker Compose y ya tendríamos nuestro sitio web con **HTTPS habilitado y todo configurado para que el certificado se vaya renovando automáticamente**.

1.3 Entregables

En esta práctica habrá que entregar un **documento técnico** con la descripción de los pasos que se han llevado a cabo durante todo el proceso.

El documento debe incluir **como mínimo** lo siguientes contenidos:

- URL del repositorio de GitHub donde se ha alojado el documento técnico escrito en Markdown.
- Descripción de la configuración del archivo `docker-compose.yml` que se ha utilizado en esta práctica.
- Descripción de las acciones que ha realizado durante durante la puesta en producción
- URL del sitio web con HTTPS habilitado.

2 Referencias

- HTTPS en Wikipedia.
- WordPress.
- Amazon Web Services (AWS).
- Let's Encrypt.
- TLS (Transport Layer Security).
- ACME (Automated Certificate Management Environment).
- Certbot.
- Ecosistema PKI y Certificados digitales. Conceptos básicos. Charla de Tomás de Hidalgo.

3 Licencia