OpenMeetings

Creación de certificados SSL Let´s Encrypt e instalación de Coturn en Ubuntu 20.04 para OpenMeetings 5.0.0-M4

El presente tutorial está hecho básicamente para aquellos que tienen instalado OpenMeetings 5.0.0 - M4 en su computadora tras un router NAT. Está testeado y funciona correctamente.

Si no tuvieran instalado OpenMeetings, puede descargar directamente la guía de instalación desde aquí:

Descargar Instalacion OpenMeetings 5.0.0-M4 en Ubuntu 20.04 lts

Doy las gracias a Maxim Solodovnik y a Carlos Heras, sin cuya colaboración en las pruebas prácticas no habría podido confirmar el correcto funcionamiento y así poder publicar el presente tutorial.
Igualmente doy las gracias a todos aquellos que han contribuido, tales como Marcus Schulz y Daniel Baker. Gracias a todos ellos.

Comenzamos…

1)

------ Creación del certificado Let´s Encrypt SSL ------

Creamos los certificados SSL de su dominio:

```
sudo apt install git
```

Descargamos git del sitio oficial para clonar Let´s Encrypt, en /opt:

```
sudo git clone https://github.com/letsencrypt/letsencrypt /opt/letsencrypt
```

...vamos al nuvo directorio:

```
cd /opt/letsencrypt
```

Es importante que su servidor-pc no tenga en uso el puerto 80 con algún servidor web o algun otro. Si fuera así deténgalo y continúe con este paso. Cuando concluya los certificados podrá lanzarlo nuevamente.

Let´s Encrypt valida “SSL Certificate Authority (CA)” el o los dominios de tu servidor.

Lo ejecutaremos con el parámetro --standalone, para que usted pueda añadir al final, cada dominio que requiera un certificado, por ejemplo: -d nuevoejemplo.com
Cambie “ejemplo.com” por el verdadero dominio de su servidor:

sudo -H ./letsencrypt-auto certonly --standalone -d ejemplo.com -d www.ejemplo.com

Preguntará por una dirección de correo de administración. Ponga uno verdadero para que le mantenga informado acerca de los certificados:

Installation succeeded.
Saving debug log to /var/log/letsencrypt/letsencrypt.log
Plugins selected: Authenticator standalone, Installer None
Enter email address (used for urgent renewal and security notices) (Enter 'c' to cancel): ...aquí ponga su dirección de correo y pulse **Enter**

Preguntará si está de acuerdo;

- -
Please read the Terms of Service at https://letsencrypt.org/documents/LE-SA-v1.2-November-15-2017.pdf. You must agree in order to register with the ACME server at https://acme-v02.api.letsencrypt.org/directory
- -
(A)gree/(C)ancel: ...escriba... a ...y pulse **Enter**

Preguntará si quiere compartir su dirección de correo:

- -
Would you be willing to share your email address with the Electronic Frontier Foundation, a founding partner of the Let's Encrypt project and the non-profit organization that develops Certbot? We'd like to send you email about our work encrypting the web, EFF news, campaigns, and ways to support digital freedom.
- -
(Y)es/(N)o: ...escriba... n ...y pulse **Enter**

...cuando finalice de hacer los certificados con éxito, mostrará lo siguiente:

IMPORTANT NOTES:
- Congratulations! Your certificate and chain have been saved at:
  /etc/letsencrypt/live/tu_dominio/fullchain.pem
  Your key file has been saved at:
  /etc/letsencrypt/live/tu_dominio/privkey.pem
  Your cert will expire on 2020-06-24. To obtain a new or tweaked
  version of this certificate in the future, simply run
  letsencrypt-auto again. To non-interactively renew *all* of your
  certificates, run "letsencrypt-auto renew"
- If you like Certbot, please consider supporting our work by:

  Donating to ISRG / Let's Encrypt: https://letsencrypt.org/donate
  Donating to EFF: https://eff.org/donate-le

2)

------ Chequear certificado de dominio ------

Vamos a ver donde están almacenados los certificados que acabamos de crear, que en nuestro caso es /etc/letsencrypt/live:

sudo ls /etc/letsencrypt/live

...mostrará el nombre de su dominio: tu_dominio

Todos los dominios que usted haya especificado en el paso anterior, se hallarán en el mismo certificado. Podemos verificarlo:

cd /opt/letsencrypt

sudo ./certbot-auto certificates ...atención al punto antes de ./cerbot-auto

...y mostrará algo similar a lo siguiente:

Found the following certs:
  Certificate Name: tu_dominio
    Domains: tu_dominio www.tu_dominio
    Expiry Date: 2020-03-24 20:49:02+00:00 (VALID: 89 days)
    Certificate Path: /etc/letsencrypt/live/tu_dominio/fullchain.pem
    Private Key Path: /etc/letsencrypt/live/tu_dominio/privkey.pem

3)

------ Renovación del certificado SSL ------

El certificado Let´s Encrypt tiene un inconveniente, y es que su validez es solo de 90 días, por lo que habremos de renovarlo.

Podemos hacer esto manualmente (siempre conectados a Internet):

cd /opt/letsencrypt

sudo ./letsencrypt-auto renew ...atención al punto en ./letsencrypt-auto.

...o podemos hacerlo automáticamente añadiendo al cron la línea de abajo para que cada domingo compruebe si hay que renovar el certificado y lo haga si fuera preciso:

sudo crontab -e

...mostrará varios editores a elegir:

Select an editor. To change later, run 'select-editor'.
1. /bin/nano <---- easiest
2. /usr/bin/vim.tiny
3. /bin/ed

Choose 1-3 [1]: ...pulse **Enter** (para elegir nano)

...y al final del archivo pegamos la línea de abajo:

30 2 * * 1 /opt/letsencrypt/letsencrypt-auto renew

...salimos del editor nano pulsando las teclas **Ctrl+x**, preguntará si guarda y pulsamos **S** y después **Enter** para salir.

4)

------ Configuración de Tomcat-OpenMeetings con los certificados SSL ------

Esta configuración que haremos ahora es solo para la serie **5.0.x** (no 4,x.x) de OpenMeetings.
Este paso número 4 hay que repetirlo cada 80 días, tras actualizar los certificados, pues son 90 los días válidos de Letsencrypt.

He seguido la ruta de instalación de OM que muestran los tutoriales de OpenMeetings que se encuentran en su sitio wiki oficial. Es decir **/opt/open504.**

Si usted hubiera hecho la instalación en una ruta distinta, modifique lo que indico a continuación.
Ya hicimos los certificados letsencrypt de nuestro dominio en el paso 1.
Ahora vamos a crear un PKCS12 que contenga la full chain y la privada. Es necesario tener instalado openssl. Lo instalamos si no:

sudo apt install openssl

Ahora lanzamos el siguiente comando:

(En una sola línea con espacio entre cada una de ellas)

sudo openssl pkcs12 -export -out /tmp/example.com_fullchain_and_key.p12
-in /etc/letsencrypt/live/example.com/fullchain.pem
-inkey /etc/letsencrypt/live/example.com/privkey.pem -name tomcat

…sustituir example.com por tu verdadero dominio (el mismo de cuando hemos hecho los certificados letsencrypt) Pedirá una contraseña, elija a su gusto y guardela en un archivo de texto,

...y ahora convertimos esa PKCS12 en un JKS empleeando java keytool:

(En una sola línea con espacio entre cada una de ellas)

sudo keytool -importkeystore -deststorepass samplePassword -destkeypass samplePassword
-destkeystore /tmp/example.com.jks -srckeystore /tmp/example.com_fullchain_and_key.p12
-srcstoretype PKCS12 -srcstorepass samplePassword -alias tomcat

…sustituya example.com por tu verdadero dominio (dos vecs), y samplePassword (tres veces) por la contraseña que haya elegido anteriormente y que había guardado en un archivo de texto.

Copiamos el archivo generado example.com.jks al directorio de instalación de Tomcat-OpenMeetings:

sudo cp /tmp/example.com.jks /opt/open504/conf

…sustituya example.com por tu verdadero dominio.

 # Pasamos a configurar Tomcat con la Java Keystore que hemos generado.

Para ello editamos el archivo server.xml:

sudo nano /opt/open504/conf/server.xml

...vamos al bloque:

```
<Connector port="5443" protocol="org.apache.coyote.http11.Http11NioProtocol"
           maxThreads="150" SSLEnabled="true">
    <SSLHostConfig>
      <Certificate certificateKeystoreFile="conf/localhost.jks"
                   certificateKeystorePassword="openmeetings"
                   certificateKeystoreType="JKS"
                   certificateVerification="false"
                   sslProtocol="TLS"
                   type="RSA" />
    </SSLHostConfig>
```

...y lo modificamos dejándolo así:

```
<Connector port="5443" protocol="org.apache.coyote.http11.Http11NioProtocol"
           maxThreads="150" SSLEnabled="true">
    <SSLHostConfig>
      <Certificate certificateKeystoreFile="conf/example.com.jks"
                   certificateKeystorePassword="samplePassword"
                   certificateKeystoreType="JKS"
                   certificateVerification="false"
                   sslProtocol="TLS"
                   type="RSA" />
    </SSLHostConfig>
```

...sustituya example.com por su verdadero dominio, y samplePassword por la contraseña que recien haya escogido (la que acaba de guardar en un archivo de texto)

...salimos del editor nano pulsando las teclas **Ctrl+x**, preguntará si guarda y pulsamos **S** y después **Enter** para salir.

5)

------ Instalación de Coturn ------

Instalaremos Coturn.

```
sudo apt install coturn
```

...editamos el siguiente archivo para que el servidor Turn pueda trabajar:

```
sudo nano /etc/default/coturn
```

...y descomentamos la línea:

```
#TURNSERVER_ENABLED=1
```

….dejándola así:

```
TURNSERVER_ENABLED=1
```

...salimos del editor nano pulsando las teclas **Ctrl+x**, preguntará si guarda y pulsamos **S** y dspués **Enter** para salir.

6)

------ Configuración del servidor Turn ------

Ahora configuraremos Turn. Creamos antes una carpeta donde turn almacene sus logs:

sudo mkdir -p /var/log/turnserver

...creamos una contraseña que necesitaremos para ponerla en el archivo de configuración del servidor turn y más tarde en un archivo de OpnMeetings. La creamos:

sudo openssl rand -hex 32

...generará algo similar a esto:
751c45cae60a2839711a94c8d6bf0089e78b2149ca602fdXXXXXXXXXXXXXXXX

…copie esa larga contraseña y péguela en un archivo de texto guardándolo.

Ahora editamos el archivo de configuración de turn:

sudo nano /etc/turnserver.conf

...en este archivo habremos de descomentar (borrar #) solo las siguientes líneas:

use-auth-secret

static-auth-secret=751c45cae60a2839711a94c8d6bf0089e78b2149ca602fdXXXXXXXXXXXXX

(en la línea de arriba pongan la larga contraseña que acabamos de guardar en un archivo de texto)

user=nobody:una_nueva_contraseña

(una nueva contraseña, esta para **nobody**, a su gusto, que ha de guardar en un archivo de texto ya que después la neesitaremos también)

realm=su_verdadero_dominio ...cambiar company.org por su verdadero dominio

stale-nonce=0 ...cambiar 600 por 0 (cero)

log-file=/var/log/turnserver/turnserver.log .

(arriba cambiar /var/log/turnserver.log por /var/log/turnserver/turnserver.log)

...salimos del editor nano pulsando las teclas **Ctrl**+**x**, preguntará si guarda y pulsamos **S** y después **Enter** para salir.

7)

------ Configuración de OpenMeetings 5.0.0-M4 con Kurento ------

Editamos el archivo applicationContext.xml de OpenMeetings:

sudo nano /opt/open504/webapps/openmeetings/WEB-INF/classes/applicationContext.xml

...y al final del archivo, en la sección <-- Kurento --> lo dejamos así:

```
<!-- Kurento -->
    <bean id="kurentoHandler" class="org.apache.openmeetings.core.remote.KurentoHandler"
init-method="init" destroy-method="destroy"
            p:kurentoWsUrl="ws://127.0.0.1:8888/kurento"
            p:checkTimeout="10000"
            p:watchThreadCount="10"
            p:turnUrl="IP publica de tu servidor:3478"
            p:turnUser="nobody:aquí la contraseña que escogió para nobody en el paso 6"
    p:turnSecret="751c45cae60a2839711a94c8d6bf0089e78b2149ca602fdXXXXXXXXXXXXX"
            p:turnMode="rest"
            p:turnTtl="60"
            p:objCheckTimeout="200"
            p:flowoutTimeout="5"
            />
```

...arriba, en:

```
 p:turnSecret="751c45cae60a2839711a94c8d6bf0089e78b2149ca602fdXXXXXXXXXXXXX"
```

...sustituya la línea:
751c45cae60a2839711a94c8d6bf0089e78b2149ca602fdXXXXXXXXXXXXX

...por la larga contraseña que generamos en el paso 6 y que guardamos en un archivo de texto,

...y salimos del editor nano pulsando las teclas **Ctrl**+**x**, preguntará si guarda y pulsamos **S** y después **Enter** para salir.

Reiniciamos coturn: sudo /etc/init.d/coturn restart

Reiniciamos docker: sudo /etc/init.d/docker restart

Kurento: sudo docker restart kms

Tomcat-OpenMeetings: sudo /etc/init.d/tomcat3 restart

8)

------ Abrir puertos necesarios para los servidores------

Necesitamos abrir determinados puertos, tanto en el router como en el firewall, para que los servidores puedan ser accesibles.

Estos son:

3478 TCP UDP IN

5443 TCP IN

8888 TCP IN

49152:65535 UDP IN-OUT

...si tiene instalado gufw en Debian 10 (interfaz de ufw firewall) puede abrirlos directamente desde ahí añadiendo reglas.
En caso de que prefiera abrirlos (en el firewall) con IPTables, estos son los comandos:

sudo iptables -A INPUT -p tcp -m tcp --dport 3478 -j ACCEPT

sudo iptables -A INPUT -p udp -m udp --dport 3478 -j ACCEPT

sudo iptables -A INPUT -p tcp -m tcp --dport 5443 -j ACCEPT

sudo iptables -A INPUT -p tcp -m tcp --dport 8888 -j ACCEPT

sudo iptables -A INPUT -p udp --match multiport --dports 49152:65535 -j ACCEPT

sudo iptables -A OUT -p udp --match multiport --dports 49152:65535 -j ACCEPT

...tras haber lanzado los comandos guardamos los cambios:

sudo service iptables save

...y reiniciamos IPTables:

sudo service iptables restart

Y con esto concluimos.

Si tuviera alguna duda o pregunta, por favor planteela en los foros de Apache OpenMeetings:

https://openmeetings.apache.org/mailing-lists.html

Gracias.

Alvaro Bustos (PMC y Committer en Apache OpenMeetings)