Creating Let's Encrypt SSL Certificates and Installing Coturn on CentOS 7 for OpenMeetings 5.0.0-M4

This tutorial is basically made for those who have OpenMeetings 5.0.0-M4 installed on your computer behind a NAT router. It's tested and working properly.

If you didn't have it installed you can directly download the installation guide from here:

Download Installation OpenMeetings 5.0.0-M4 on CentOS 7

I thank Maxim Solodovnik and Carlos Heras, without whose collaboration in the trials practices could not have confirmed the proper functioning and thus be able to publish the present tutorial.

I also thank all those who have contributed such as Marcus Schulz and Daniel Baker.
Thanks to all them.

Starting…

1)

------ Creating SSL Let´s Encrypt certificates ------

Creating SSL certificates of your domain:

sudo yum install git

Download git from the official site to clone Let´s Encrypt at /opt:

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

... let's go to the new directory:

cd /opt/letsencrypt

It is important that your pc-server does not have port 80 in use with some web server or some other. If so, stop it and continue with this step. When the certificates are completed, you can throw it again. This port must be open in the router and the firewall.

Let´s Encrypt validate “SSL Certificate Authority (CA)” of your domain.

We'll run it with the --standalone parameter, so you can add each domain at the end requires a certificate, for exemple: -d newexemple.com
Change "exemple.com" to the true domain of your server:

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

You will be asked for an admin email address. Put a real one to get you keep you informed about certificates:

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): ...here your mail address and press **Enter**

Ask if you agree:

- -
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: ...type... a ...and press **Enter**

Ask if you want to share your email address:

- -
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: ...type... n ...and press **Enter**

when you finish making the certificates successfully, it will show the following:

IMPORTANT NOTES:
- Congratulations! Your certificate and chain have been saved at:
  /etc/letsencrypt/live/your_domain/fullchain.pem
  Your key file has been saved at:
  /etc/letsencrypt/live/your_domain/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)

------ Checking domain certificates ------

We´ll see where are stored the certificates we just create, that in our case will be at /etc/letsencrypt/live:

sudo ls /etc/letsencrypt/live

...will show your domain name: your_domain

All domains that you specified in the previous step will be located in the same certificate. We can verify this:

cd /opt/letsencrypt

sudo ./certbot-auto certificates ...attention to the point before ./cerbot-auto

and it will show something similar to the following:

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

3)

------ Renewing the SSL certificate ------

The Let's Encrypt certificate has an drawback, and is that it is valid only 90 days, so we're going to have to renew it.
We can do this manually (always connected to Internet):

cd /opt/letsencrypt

sudo ./letsencrypt-auto renew ...attention to the point in ./letsencrypt-auto

...or we can do it automatically by adding the bottom line to the cron so that every Sunday check if the certificate needs to be renewed and do so if necessary:

sudo crontab -e

...it will show several editors to choose from:

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]: ...press **Enter** to select nano editor

...and at the end of the file we paste the line below:

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

...exit the nano editor by pressing the **Ctrl+x** keys, ask if you save and press **Y** and then **Enter** to exit.

4)

------ Configuring Tomcat-OpenMeetings with SSL certificates ------

This configuration we will now make is only for the 5.xx (not 4) openMeetings series.
This step number 4 must be repeated every 80 days, after updating the certificates, as it is 90 Let´s Encrypt's valid days.

I followed the OM installation path that show the OpenMeetings tutorials that are found on their official wiki site. I mean **/opt/open504.**

If you had done the installation on a different path, modify what you indicate below.
We already made the letsencrypt certificates for our domain in step 1.
Now let's create a PKCS12 that contains the full chain and the private one. It is necessary to have installed openssl. We install it if not:

sudo yum install openssl

Now run the following command:

(Only one line with space between each of them)

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

…replace example.com with your true domain (the same as when we made letsencrypt certificates)
...will ask for a password. Type one that you likes and paste in a text file (will need now)

And now convert that PKCS12 to JKS file using java keytool:

(Only one line with space between each of them)

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

…replace example.com with your true domain (twice), and samplePassword (three times) with the password you just choosed (it you pasted in a text file).

Copy the generated example.com.jks file to the Tomcat-OpenMeetings installation directory:

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

…replace example.com with your true domain.

 # Configure Tomcat with the Java Keystore that we generated..

For that edit server.xml file:

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

...let's go to the block:

```
<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>
```

...and we modified it by leaving it like this:

```
<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>
```

...replace example.com with your true domain, and samplePassword with the password that you've just chosen (the one you just saved to a text file)

...exit the nano editor by pressing the **Ctrl+x** keys, ask if you save and press **Y** and then **Enter** to exit.

5)

------ Coturn installation and configuration of Turn server------

Install Coturn (Turn server):

```
sudo yum install coturn
```

Configuration of turn server.

First we create a password that we'll need to put it in the configuration file of the turn server and later in an OpnMeetings file. We created it:

```
sudo openssl rand -hex 32
```

...will generate something similar to this:
751c45cae60a2839711a94c8d6bf0089e78b2149ca602fdXXXXXXXXXXXXXXXXX

…copy that long password and paste it into a text file by saving it.

Now edit the turn file configuration:

```
sudo nano /etc/coturn/turnserver.conf
```

...in this file we will have to uncomment (delete #) only the following lines:

use-auth-secret

static-auth-secret=751c45cae60a2839711a94c8d6bf0089e78b2149ca602fdXXXXXXXXXXXXX

(on the above line put the long password we just saved in a text file)

user=kurento:a_new_password

(a new password this for **kurento**, to your liking, you have to save to a text file that we will later need it too))

realm=your_real_domaing ...change company.org to your real domain

stale-nonce=0 ...change 600 to 0 (zero)

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

...exit the nano editor by pressing the **Ctrl+x** keys, ask if you save and press **Y** and then **Enter** to exit.

6)

------ Setting Up OpenMeetings 5.0.0-M4 with Kurento media server------

Edit the applicationContext.xml file of OpenMeetings:

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

...and at the end of the file, in the <-- Kurento --> section we modify it like this:

```
<!-- 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="Public IP of your server:3478"
            p:turnUser="kurento:here the password you choose for kurento in step 6"
    p:turnSecret="751c45cae60a2839711a94c8d6bf0089e78b2149ca602fdXXXXXXXXXXXXX"
            p:turnMode="rest"
            p:turnTtl="60"
            p:objCheckTimeout="200"
            p:flowoutTimeout="5"
            />
```

...above, in:

p:turnSecret="751c45cae60a2839711a94c8d6bf0089e78b2149ca602fdXXXXXXXXXXXXX"

...replace the line: 751c45cae60a2839711a94c8d6bf0089e78b2149ca602fdXXXXXXXXXXXXX

...by the long password that we generated in step 5 and that we save in a text file

Exit the nano editor by pressing the **Ctrl+x** keys, ask if you save and press **Y** and then **Enter** to exit.

And all we have to do is modify the Tomcat-OpenMeetings run script so that it's the kurento user who launch it.

To do this we edited the aforementioned script (which we would already have after installing OpenMeeting 5.0.0-M4 following the tutorial found on the wki of the official OM site):

sudo nano /etc/init.d/tomcat3

...and modify the line:

$CATALINA_HOME/bin/startup.sh -u root -Dcatalina.base$CATALINA_BASE

...to

$CATALINA_HOME/bin/startup.sh -u kurento -Dcatalina.base$CATALINA_BASE
.

...exit the nano editor by pressing the **Ctrl+x** keys, ask if you save and press **Y** and then **Enter** to exit.

IS IMPORTANT...we must reboot the machine, and after continue in the next stepm7:

sudo reboot

7)

------ Run the servers after rebooted the machine ------

Run any server related with OpenMeetings:

MariaDB: sudo systemctl start mariadb.service

Docker: sudo systemctl start docker.service

Kurento: sudo docker start kms

Coturn: sudo systemctl start coturn.service

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

8)

------ Open ports required for servers------

We need open some ports in the router and the firewall for the servers access. These are:

3478 TCP-UDP IN

5443 TCP IN

8888 TCP IN

49152:65535 UDP IN-OUT

To open them (the firewall) with IPTables, these are the commands:

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

...after launching the commands we save the changes:

sudo service iptables save

...and restart IPTables:

sudo service iptables restart

And with this we conclude.

If you have some doubt or question, please raise it in the Apache OpenMeetings forums:

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

OpenMeetings

Thank you .

Alvaro Bustos (PMC and Committer at Apache OpenMeetings)