<https://access.redhat.com/documentation/pt-br/red_hat_openstack_platform/9/html/director_installation_and_usage/appe-ssltls_certificate_configuration>

The /etc/pki/CA/index.txt file stores records of all signed certificates. Check if this file exists. If it does not exist, create an empty file:

**$ sudo touch /etc/pki/CA/index.txt**

The /etc/pki/CA/serial file identifies the next serial number to use for the next certificate to sign. Check if this file exists. If it does not exist, create a new file with a new starting value:

**$ sudo echo '1000' | sudo tee /etc/pki/CA/serial**

## A.2. Creating a Certificate Authority

Normally you sign your SSL/TLS certificates with an external certificate authority. In some situations, you might aim to use your own certificate authority. For example, you might aim to have an internal-only certificate authority.

For example, generate a key and certificate pair to act as the certificate authority:

**$ cd** /etc/pki/CA/

**$ openssl genrsa -out ca.key.pem 4096**

**$ openssl req -key ca.key.pem -new -x509 -days 7300 -extensions v3\_ca -out ca.crt.pem**

The openssl req command asks for certain details about your authority. Enter these details.

This creates a certificate authority file called ca.crt.pem.

## A.3. Adding the Certificate Authority to Clients

For any external clients aiming to communicate using SSL/TLS, copy the certificate authority file to each client that requires access your Red Hat OpenStack Platform environment. Once copied to the client, run the following command on the client to add it to the certificate authority trust bundle:

**$ sudo cp ca.crt.pem /etc/pki/ca-trust/source/anchors/**

**$ sudo update-ca-trust extract**

## A.4. Creating an SSL/TLS Key

Run the following commands to generate the SSL/TLS key (server.key.pem), which we use at different points to generate our undercloud or overcloud certificates:

**$ openssl genrsa -out server.key.pem 2048**

## A.5. Creating an SSL/TLS Certificate Signing Request

This next procedure creates a certificate signing request for either the Undercloud or Overcloud.

Copy the default OpenSSL configuration file for customization.

~~$ cp /etc/pki/tls/openssl.cnf . //Não é necessário~~

~~Edit the custom openssl.cnf file and set SSL parameters to use for the director. An example of the types of parameters to modify include:~~

~~[req]~~

~~distinguished\_name = req\_distinguished\_name~~

~~req\_extensions = v3\_req~~

~~[req\_distinguished\_name]~~

~~countryName = Country Name (2 letter code)~~

~~countryName\_default = AU~~

~~stateOrProvinceName = State or Province Name (full name)~~

~~stateOrProvinceName\_default = Queensland~~

~~localityName = Locality Name (eg, city)~~

~~localityName\_default = Brisbane~~

~~organizationalUnitName = Organizational Unit Name (eg, section)~~

~~organizationalUnitName\_default = Red Hat~~

~~commonName = Common Name~~

~~commonName\_default = 192.168.0.1~~

~~commonName\_max = 64~~

~~[ v3\_req ]~~

~~# Extensions to add to a certificate request~~

~~basicConstraints = CA:FALSE~~

~~keyUsage = nonRepudiation, digitalSignature, keyEncipherment~~

~~subjectAltName = @alt\_names~~

~~[alt\_names]~~

~~IP.1 = 192.168.0.1~~

~~DNS.1 = instack.localdomain~~

~~DNS.2 = vip.localdomain~~

~~DNS.3 = 192.168.0.1~~

~~Set the commonName\_default to the IP address, or fully qualified domain name if using one, of the Public API:~~

* ~~For the Undercloud, use the undercloud\_public\_vip parameter in undercloud.conf. If using a fully qualified domain name for this IP address, use the domain name instead.~~
* ~~For the Overcloud, use the IP address for the Public API, which is the first address for the ExternalAllocationPools parameter in your network isolation environment file. If using a fully qualified domain name for this IP address, use the domain name instead.~~

~~Edit the alt\_names section to include the following entries:~~

* ~~IP - A list of IP addresses for clients to access the director over SSL.~~
* ~~DNS - A list of domain names for clients to access the director over SSL. Also include the Public API IP address as a DNS entry at the end of the alt\_names section.~~

~~For more information about openssl.cnf, run man openssl.cnf.~~

Run the following command to generate certificate signing request (server.csr.pem):

**$ openssl req ~~-config openssl.cnf~~ -key server.key.pem -new -out server.csr.pem**

Make sure to include the SSL/TLS key you created in [Section A.4, “Creating an SSL/TLS Key”](https://access.redhat.com/documentation/pt-br/red_hat_openstack_platform/9/html/director_installation_and_usage/appe-SSLTLS_Certificate_Configuration" \l "Creating_an_SSL_TLS_Key" \o "A.4. Creating an SSL/TLS Key) for the -key option.

**Important**

The openssl req command asks for several details for the certificate, including the Common Name. Make sure the Common Name is set to the IP address of the Public API for the Undercloud or Overcloud (depending on which certificate set you are creating). The openssl.cnf file should use this IP address as a default value.

Use the server.csr.pem file to create the SSL/TLS certificate in the next section.

## A.6. Creating the SSL/TLS Certificate

The following command creates a certificate for your undercloud or overcloud:

**$ sudo openssl ca ~~-config openssl.cnf~~ -extensions v3\_req -days 3650 -in server.csr.pem -out server.crt.pem -cert ca.crt.pem -keyfile ca.key.pem**

This command uses:

* The configuration file specifying the v3 extensions. Include this as the -config option.
* The certificate signing request from [Section A.5, “Creating an SSL/TLS Certificate Signing Request”](https://access.redhat.com/documentation/pt-br/red_hat_openstack_platform/9/html/director_installation_and_usage/appe-SSLTLS_Certificate_Configuration" \l "Creating_an_SSL_TLS_Certificate_Signing_Request" \o "A.5. Creating an SSL/TLS Certificate Signing Request) to generate the certificate and sign it throught a certificate authority. Include this as the -in option.
* The certificate authority you created in [Section A.2, “Creating a Certificate Authority”](https://access.redhat.com/documentation/pt-br/red_hat_openstack_platform/9/html/director_installation_and_usage/appe-SSLTLS_Certificate_Configuration" \l "Creating_a_Certificate_Authority" \o "A.2. Creating a Certificate Authority), which signs the certificate. Include this as the -cert option.
* The certificate authority private key you created in [Section A.2, “Creating a Certificate Authority”](https://access.redhat.com/documentation/pt-br/red_hat_openstack_platform/9/html/director_installation_and_usage/appe-SSLTLS_Certificate_Configuration" \l "Creating_a_Certificate_Authority" \o "A.2. Creating a Certificate Authority). Include this as the -keyfile option.

This results in a certificate named server.crt.pem. Use this certificate in conjunction with the SSL/TLS key from [Section A.4, “Creating an SSL/TLS Key”](https://access.redhat.com/documentation/pt-br/red_hat_openstack_platform/9/html/director_installation_and_usage/appe-SSLTLS_Certificate_Configuration" \l "Creating_an_SSL_TLS_Key" \o "A.4. Creating an SSL/TLS Key) to enable SSL/TLS.

## ~~A.7. Using the Certificate with the Undercloud~~

~~Run the following command to combine the certificate and key together:~~

**~~$ cat server.crt.pem server.key.pem > undercloud.pem~~**

This creates a undercloud.pem file. You specify the location of this file for the undercloud\_service\_certificate option in your undercloud.conf file. This file also requires a special SELinux context so that the HAProxy tool can read it. Use the following example as a guide:

**~~$ sudo mkdir /etc/pki/instack-certs~~**

**~~$ sudo cp ~/undercloud.pem /etc/pki/instack-certs/.~~**

**~~$ sudo semanage fcontext -a -t etc\_t "/etc/pki/instack-certs(/.\*)?"~~**

**~~$ sudo restorecon -R /etc/pki/instack-certs~~**

**$ sudo semanage fcontext -a -t etc\_t "/etc/pki/CA(/.\*)?"**

**$ sudo restorecon -R /etc/pki/CA**

~~Add the undercloud.pem file location to the undercloud\_service\_certificate option in the undercloud.conf file. For example:~~

~~undercloud\_service\_certificate = /etc/pki/instack-certs/undercloud.pem~~

~~In addition, make sure to add your certificate authority from [Section A.2, “Creating a Certificate Authority”](https://access.redhat.com/documentation/pt-br/red_hat_openstack_platform/9/html/director_installation_and_usage/appe-SSLTLS_Certificate_Configuration" \l "Creating_a_Certificate_Authority" \o "A.2. Creating a Certificate Authority) to the Undercloud’s list of trusted Certificate Authorities so that different services within the Undercloud have access to the certificate authority:~~

~~$ sudo cp ca.crt.pem /etc/pki/ca-trust/source/anchors/~~

~~$ sudo update-ca-trust extract~~

~~Continue installing the Undercloud as per the instructions in [Section 4.6, “Configuring the Director”](https://access.redhat.com/documentation/pt-br/red_hat_openstack_platform/9/html/director_installation_and_usage/chap-Installing_the_Undercloud" \l "sect-Configuring_the_Director" \o "4.6. Configuring the Director).~~

## ~~A.8. Using the Certificate with the Overcloud~~

~~Include the contents of your certificate files in the enable-tls.yaml environment file from [Section 6.10, “Enabling SSL/TLS on the Overcloud”](https://access.redhat.com/documentation/pt-br/red_hat_openstack_platform/9/html/director_installation_and_usage/chap-Configuring_Advanced_Customizations_for_the_Overcloud" \l "sect-Enabling_SSLTLS_on_the_Overcloud" \o "6.10. Enabling SSL/TLS on the Overcloud). The Overcloud deployment process takes the parameters from enable-tls.yaml and automatically integrates them on each node in the Overcloud.~~

Configuração do apache ssl HTTPd

<https://www.redhat.com/sysadmin/webserver-use-https>

# dnf install httpd mod\_ssl

# mkdir /var/www/https

# echo secure content > /var/www/https/index.html

## SELinux security

Verify that the correct SELinux context is set:

[root@webserver ~]# ll -Z /var/www

Also verify the SELinux context is correct for your **index.html** file:

[root@webserver ~]# ll -Z /var/www/https/index.html

-rw-r--r--. 1 root root unconfined\_u:object\_r:httpd\_sys\_content\_t:s0 15 Oct  1 15:07 /var/www/https/index.html

[root@webserver ~]#

[root@webserver ~]# vi /etc/httpd/conf.d/ssl.conf

[root@webserver ~]# grep -e httpd.crt -e httpd.key /etc/httpd/conf.d/ssl.conf -B1

**#**SSLCertificateFile /etc/pki/tls/certs/localhost.crt SSLCertificateFile /etc/pki/CA/**server.crt.pem**

**#**SSLCertificateKeyFile /etc/pki/tls/private/localhost.key SSLCertificateKeyFile /etc/pki/CA/**server.key.pem**

[root@webserver ~]#

[root@webserver ~]# grep DocumentRoot /etc/httpd/conf.d/ssl.conf DocumentRoot "/var/www/https" **#**DocumentRoot "/var/www/html" [root@webserver ~]#

## Firewall settings

The httpd service has now been configured, but we need to make sure that traffic is allowed through the firewall. We'll enable port 443, and then reload the firewall:

[root@webserver ~]# firewall-cmd --permanent --add-port=443/tcp

success

[root@webserver ~]# firewall-cmd --reload

success

## Final configuration and testing

Enable the httpd service to start at boot and then restart the httpd service:

[root@webserver ~]# systemctl enable httpd

[root@webserver ~]# systemctl restart httpd

[root@webserver ~]# systemctl status httpd