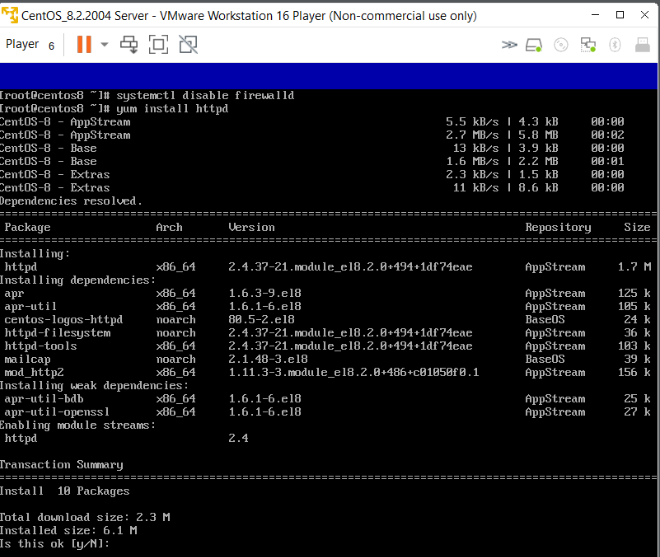
# **Report on VPN (Task 3)**

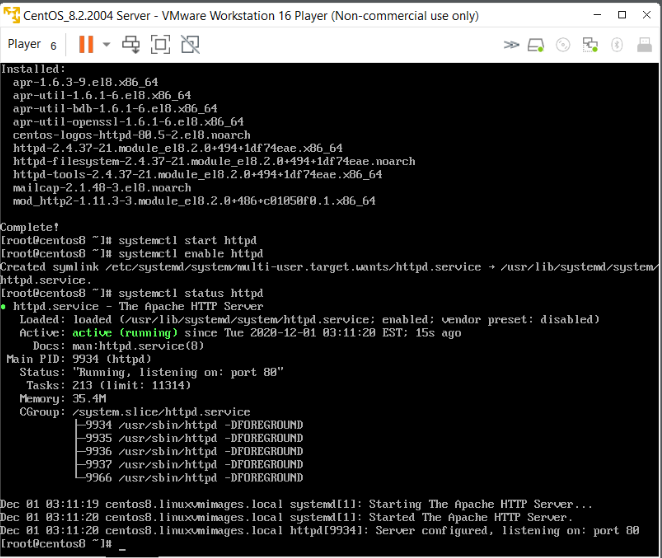
**Name:** Aravindh JAWAHAR

In this lab, we will create a PKI. We will see how to create a root CA and a client certificate. In this lab, we will need a CentOS 8 VM with internet that can be accessed from your computer web browser

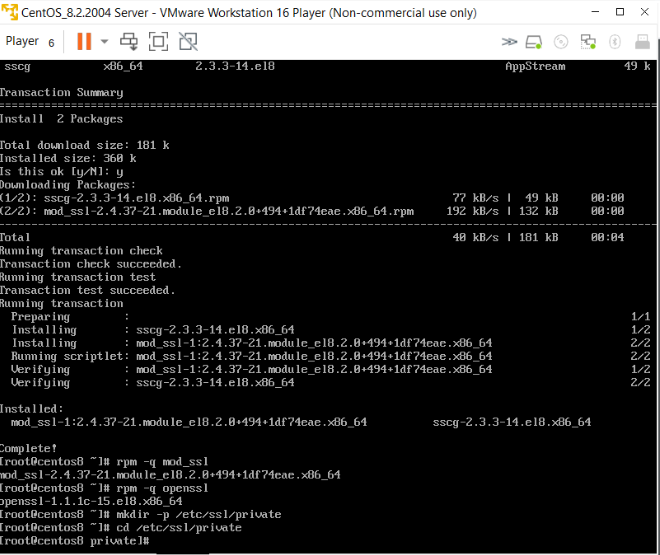
Apache server has been installed using the yum install httpd command



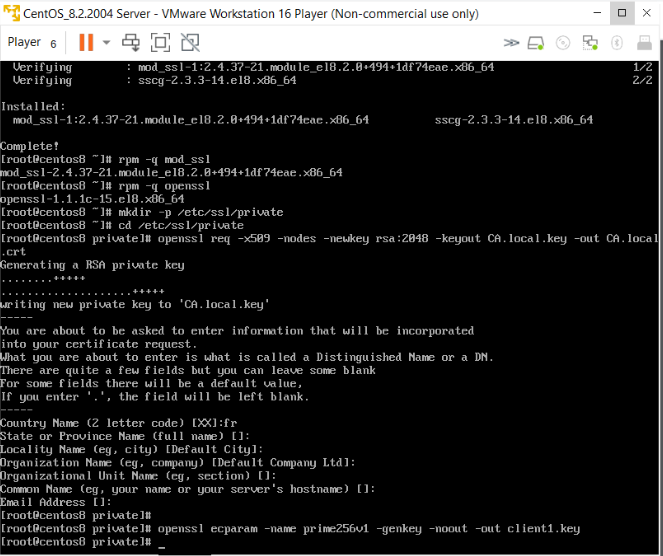
HTTPD service has been started to use the apache server on the external host using its physical IP address of the server with HTTP format without encryption. And the service should be enabled to achieve the usage of httpd on the browser without any load failure.



Now we are going to create https certificate for httpd server by installing SSL module using rpm command and we check the installation using the rpm command for both mod\_ssl and openssl.



Now we create the directory for the Certificate authority (CA) . In the private directory we create certificate authority using the **OPENSSL** command along with **RSA** encryption with certificate and key for the encryption. The OpenSSL command will generate a 2048-bit RSA private key

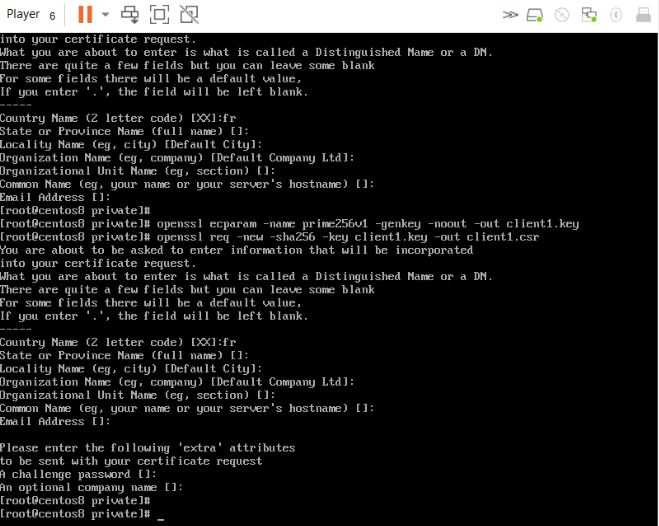


**7 .** using openssl command and ecparameters we generate key with the group name prime256v1 to client1.key file

**8 . CSR** stands for Certificate Signing Request. A **CSR contains** information such your organization's name, your domain name, and your location, and is filled out and submitted to a certificate authority

openssl req -new -sha256 -key client1.key -out client1.csr

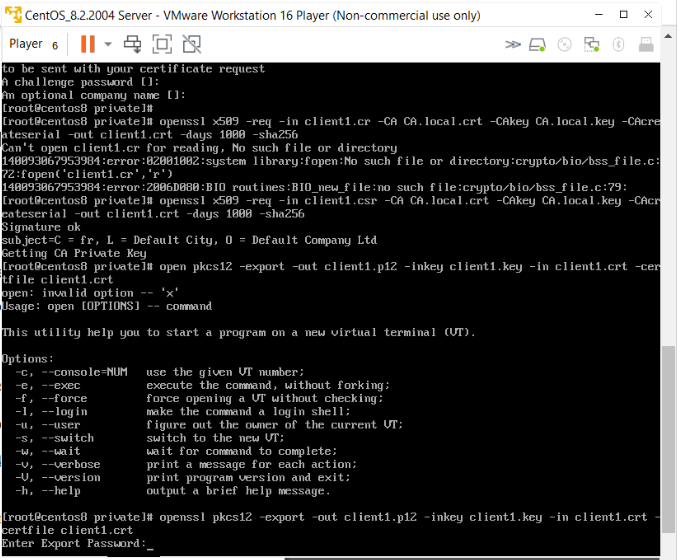
The above command generates the new key called client1.key and generate the certificate signing request.



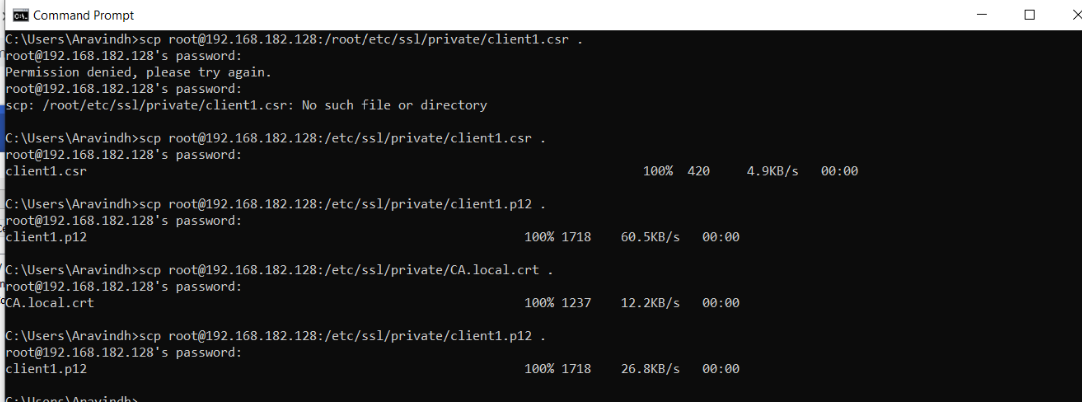
9 . The command

openssl x509 -req -in client1.csr -CA CA.local.crt -CAkey CA.local.key -CAcreateserial -out client1.crt -days 1000 -sha256

It ha the parameters that it required the client1.csr and then the local certificate key which will be valid for 1000 days under sha256 encryption method. And while exporting the password has been setup for the keys and certificates.



Now we copy the certificates from Linux VM to window using command prompt on host windows machine and downloaded to the location to import in the chrome



12 . Go back to your web server and configure it for user authentication.

Add those lines in the https virtual host in /etc/httpd/conf.d/ssl.conf :

SSLOptions +StdEnvVars

SSLVerifyClient require

SSLCACertificateFile /etc/ssl/private/CA.local.crt

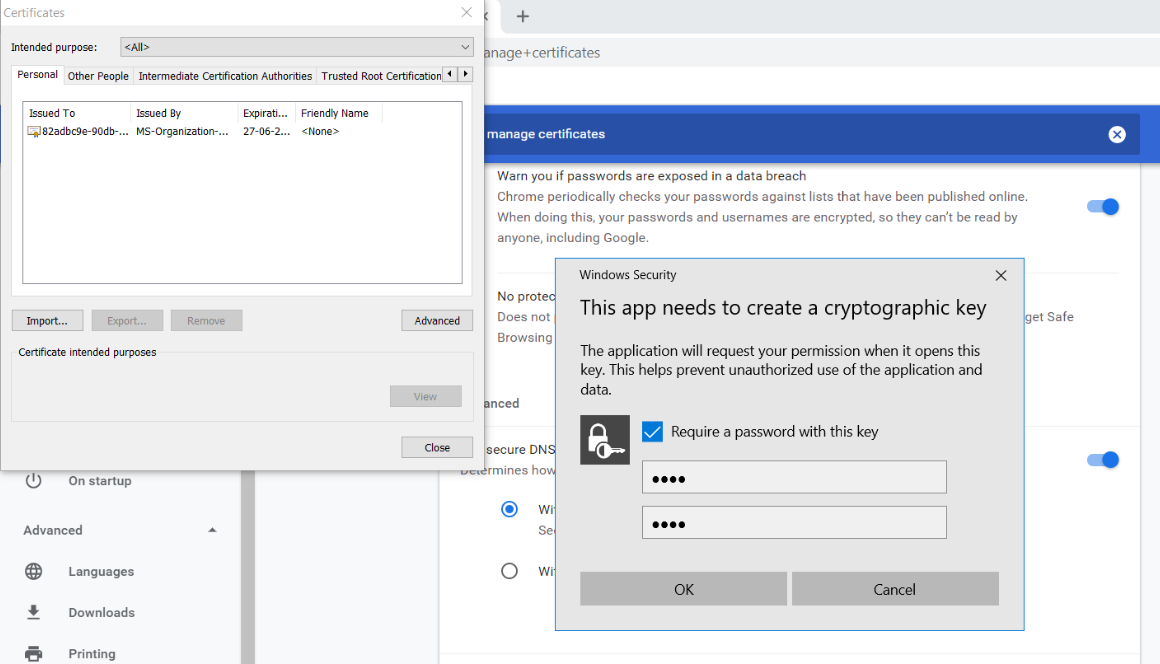
Now we configure the ssl.conf and opting to use our certificate for signing when used by the client to access the server in browser.

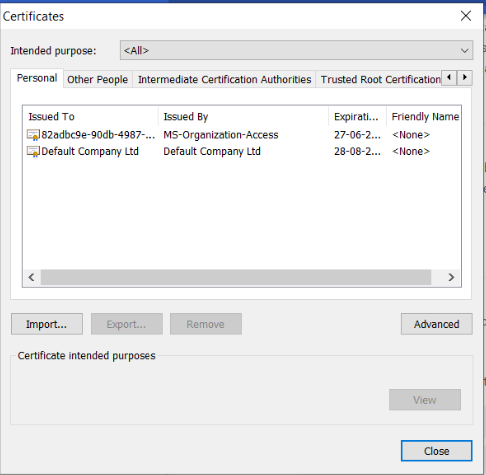
We set environment variables in SSLOption +StdEnvVars

Client verification is required and achieved by SSLVerifyClient require

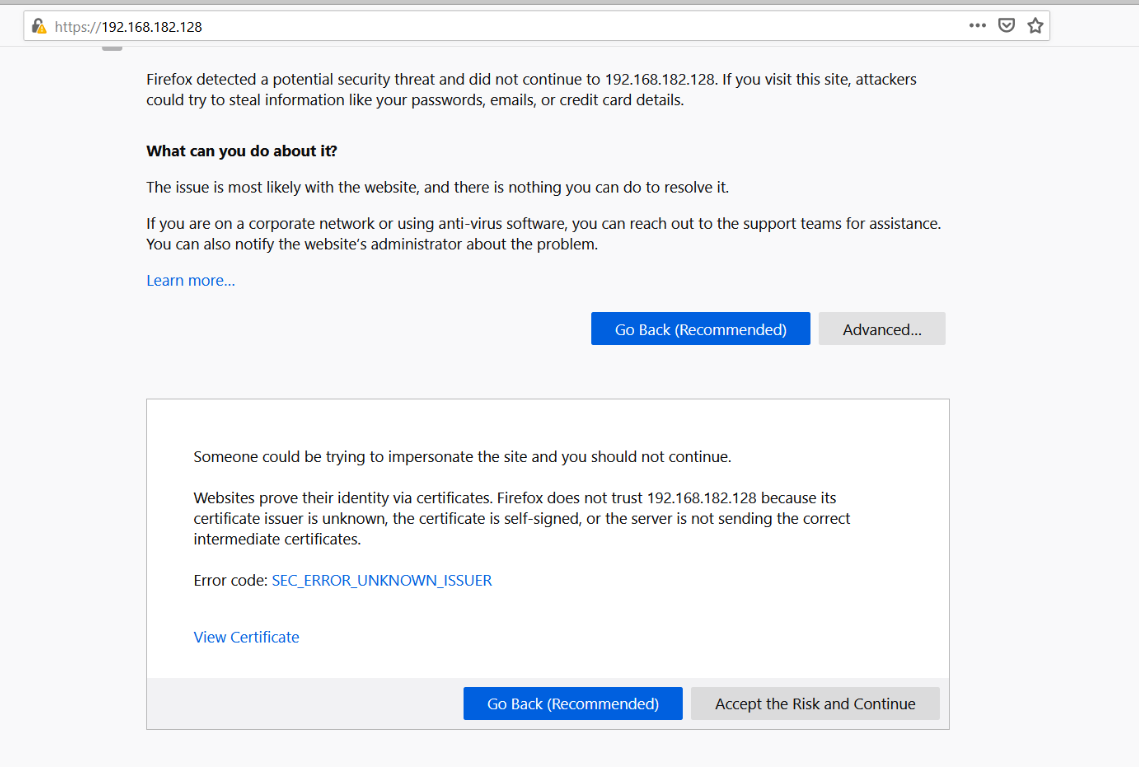
Now we append the crt file or key to the ssl in order to make user sign with the corresponding generated certifiace.

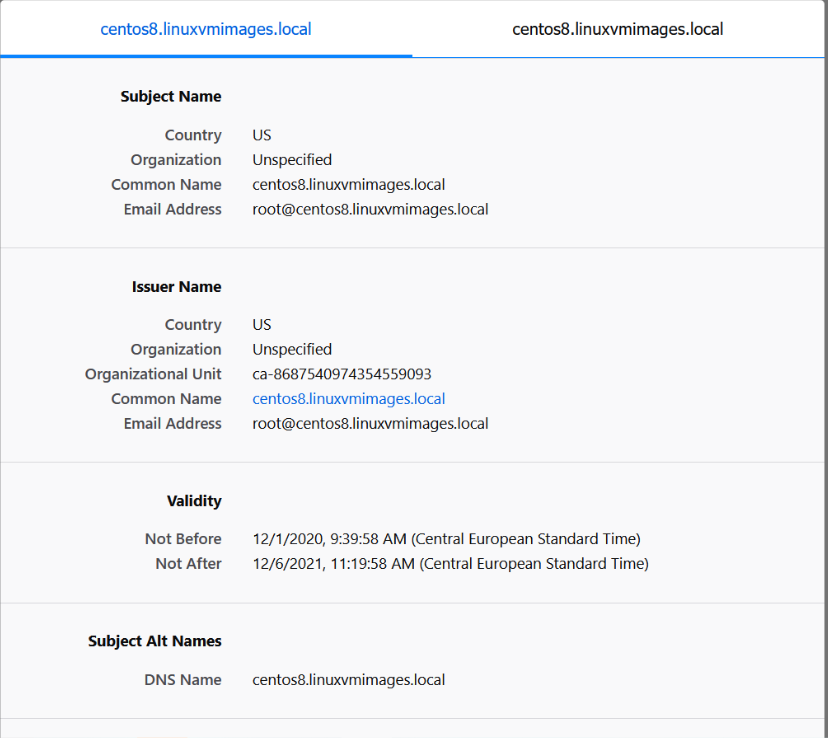
Now we import the key and certificate in the chrome browser where the screenshots are below:





14 . If we open in other browser than chrome in my case it will prompt for certificate signing request and you need to accept in order to open the server in the web browser.





15 . You fill out the appropriate forms add your public keys (they are just numbers) and send it/them to the certificate authority. (this is a **certificate Request**)The certificate authority does some checks ( depends on authority), and sends you back the keys enclosed in a **certificate**.The certificate is**signed** by the**Issuing Certificate authority**, and this guarantees the keys.Now when someone wants your public keys, you send them the certificate, they **verify the signature** on the certificate, and if it verifies, then they can **trust your keys**.

16 . **Certificate**-based **authentication** is the use of a Digital **Certificate** to identify a user, machine, or device before granting access to a resource, network, application, etc. In the case of user **authentication**, it is often deployed in coordination with traditional methods such as username and password.