mod17_ex04: Creating Client Keystore

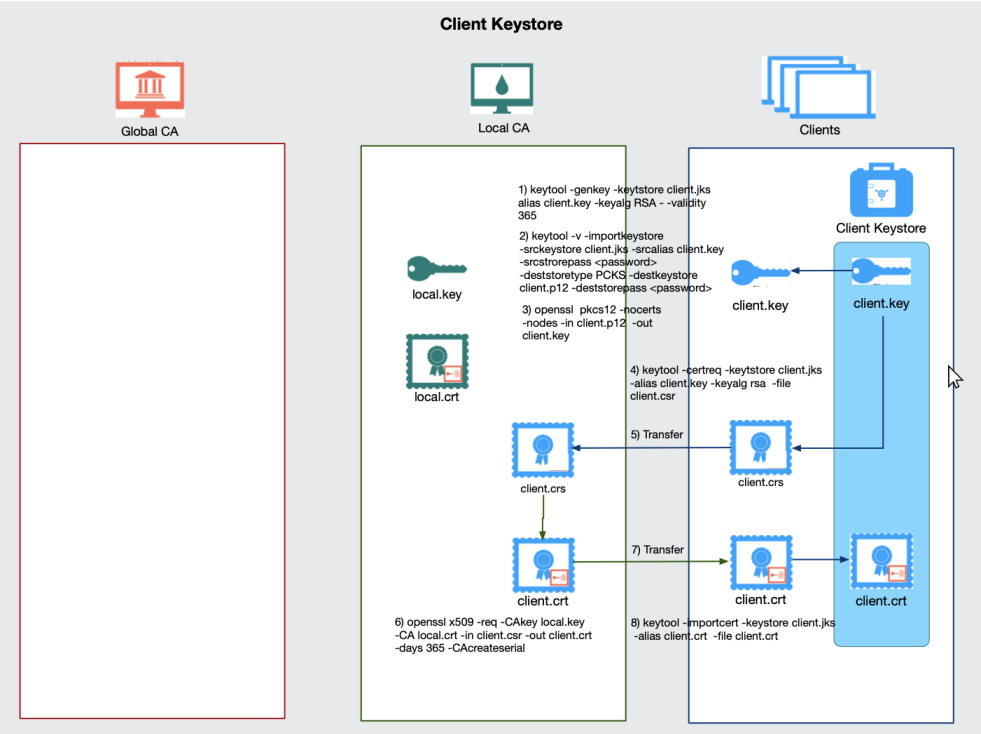
The purpose of this exercise is to create a host keystore for client. This is a repeat of Creating a Local Keystore with additional keytool commands.

Reference Information

The following documents provide information related to this exercise.

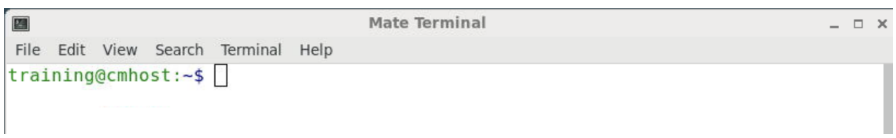
- TBD

1. Overview creating client keystore



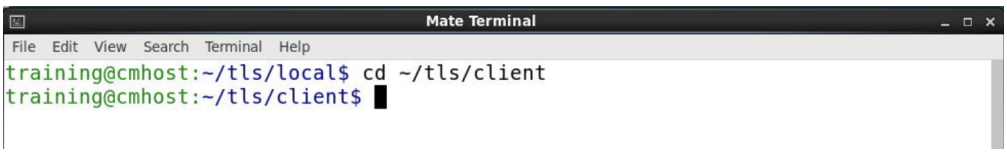
1. The keytool command will create a client keystore with a private key.
2. The keytool command does not have a function for exporting the private key. The practice is to change the keystore format to PKCS12.
3. The openssl command will convert the PKCS12 private key into the PEM format.
4. The keytool command will create a certificate signing request (.csr).
5. Transfer the .csr file to the CA.
6. The CA uses the openssl command to sign the csr by creating the certificate file.
7. Transfer the .crt file to the client.
8. The keytool command imports the signed certificate file into the client keystore.

2. Open a Mate terminal as the user training.



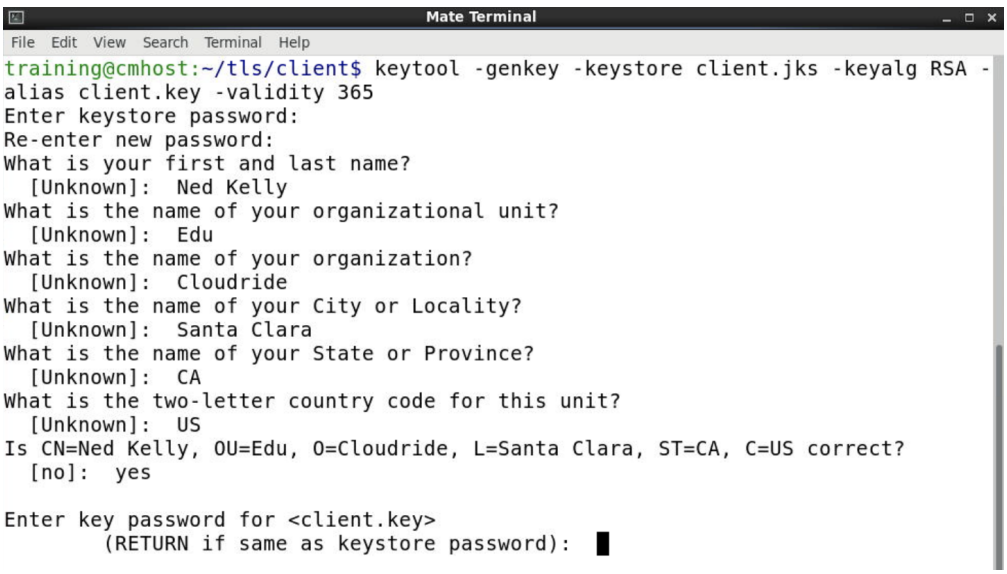
3. Create the client keystore.

3.1 Change directory to client.




```
$ cd ~/tls/client
```

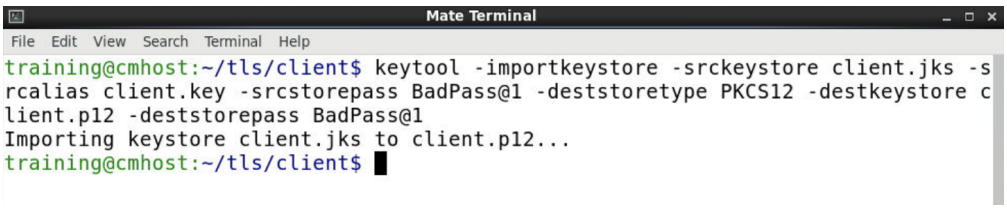
3.2 Create the local keystore with only a private key. Use password <password>. Use the information provided.




```
$ keytool –genkey –keystore client.jks  –keyalg RSA –alias client.key –
validity 365
```

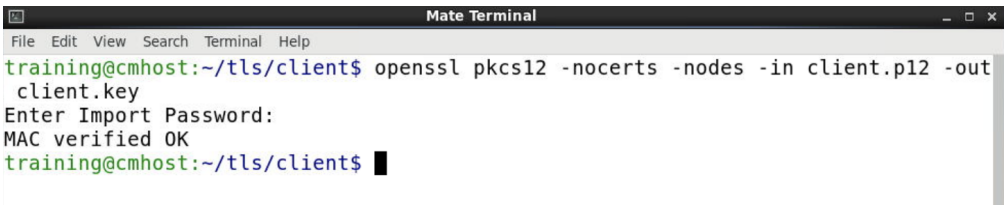
4. Export the private key from the local keystore.

4.1 Convert the keystore into a PKCS12 format.




```
$ keytool –importkeystore –srckeystore client.jks –srcalias client.key –
srcstorepass <password> –deststoretype PKCS12 –destkeystore client.p12 –
deststorepass <password>
```

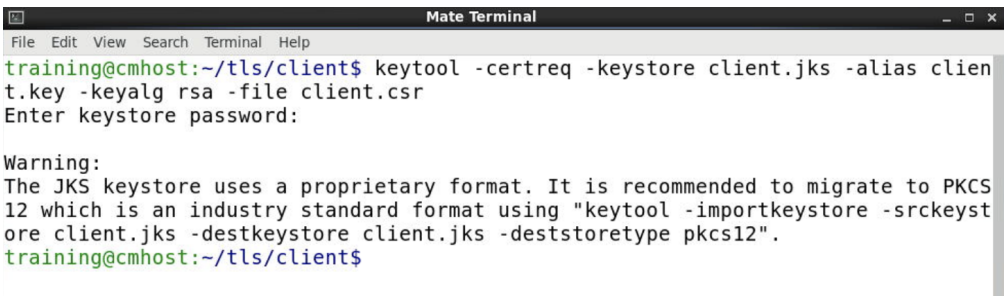
4.2 Convert PKCS12 format to PEM format.




```
$ openssl pkcs12 -nocerts -nodes -in client.p12 -out client.key
```

5. Create a Certificate Signing Request.

5.1 Use the keytool to create a signing request.




```
$ keytool -certreq -keystore client.jks  -alias client.key -keyalg rsa -file
client.csr
```

5.2 Transfer the signing request to the CA.

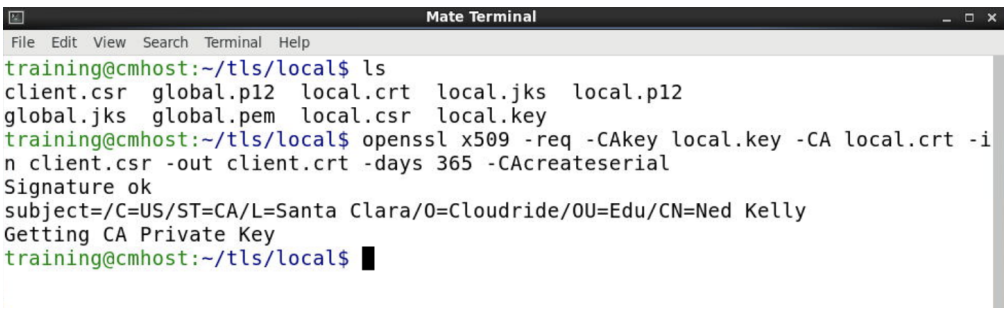

```
$ cp client.csr ~/tls/local/client.csr
```

6. Sign a Certificate Signing Request.

6.1 Change directory to local.


```
$ cd ~/tls/local
```

6.2 Use the opensssl command and the CA private key and CA public certificate to sign the certificate signing request.



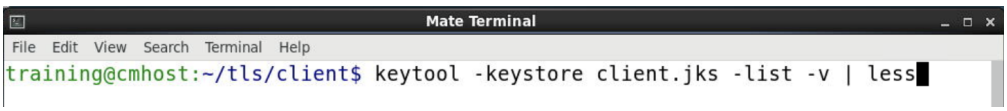

```
$ openssl x509 -req -CAkey local.key -CA local.crt -in client.csr -out
client.crt -days 365 -CAcreateserial
```

6.3 Transfer the signed certificates back to the client.


```
$ cp client.crt ~/tls/client/client.crt
```

7. Validate the signed certificate.

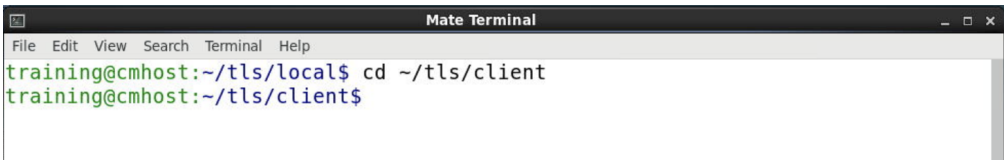
7.1 List the contents of the keystore. Compare alias local to local.crt.




```
% keytool -list -v -keystore client.jks  | less
```

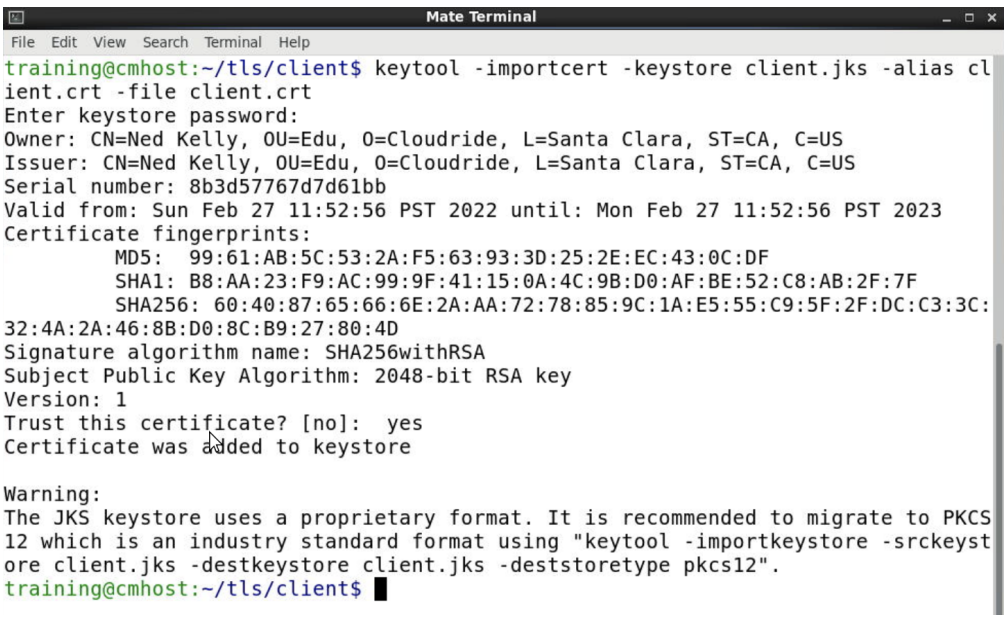
8. Import the signed certificate into the keystore.

8.1 Change directories to local.




```
$ cd ~/tls/client
```

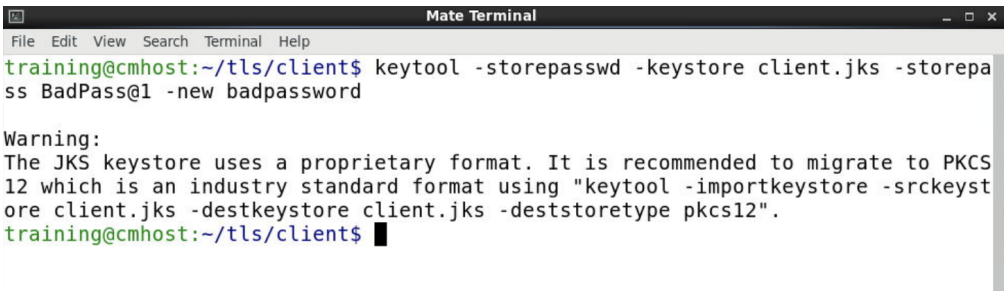
8.2 Import the signed certiticate in the local keystore.




```
$ keytool -importcert -keystore client.jks  -alias client.crt -file
client.crt
```

9. Managing Certificates in Keystore

9.1 Change the keystore password to badpassword.

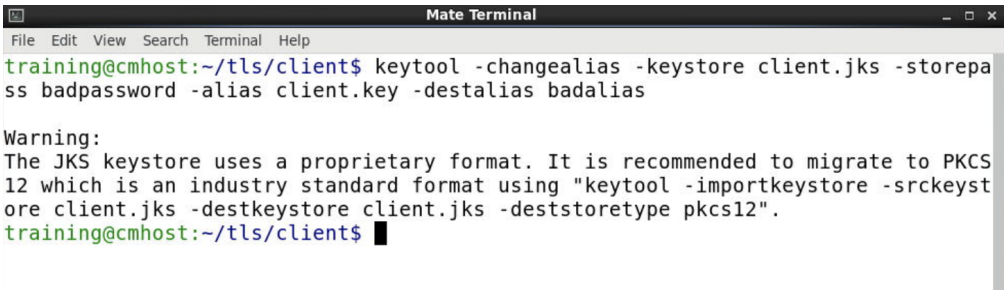



```
% keytool -storepasswd -keystore client.jks -storepass <password> -new
badpassword
```

9.2 Change the password for the private key to badpassword.

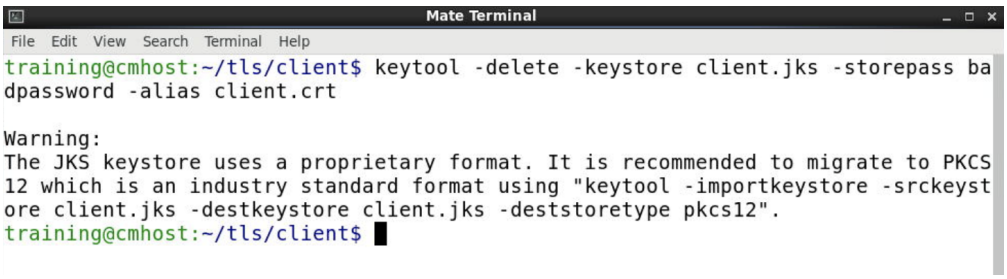

```
% keytool -keypasswd -keystore client.jks -storepass badpassword -keypass
BadPass@1 -alias client.key
```

9.3 Change the alias for the private key.




```
% keytool -changealias -keystore client.jks -storepass badpassword -alias
client.key -destalias badalias
```

9.4 Delete the public certificate, alias client.crt




```
% keytool –delete –keystore client.jks –storepass badpassword –alias
client.crt
```