Rapport de TP : Public Key Infrastructure (PKI) Lab

Réalisé par : Chater Ahmed et Daouda David Coulibaly

---

1. Introduction

Ce rapport présente les étapes et les observations du TP sur l'Infrastructure à Clé Publique (PKI). L'objectif était de comprendre les bases de PKI, la génération de certificats numériques, la configuration d'un serveur HTTPS, et la mise en œuvre d'attaques de type Man-In-The-Middle (MITM).

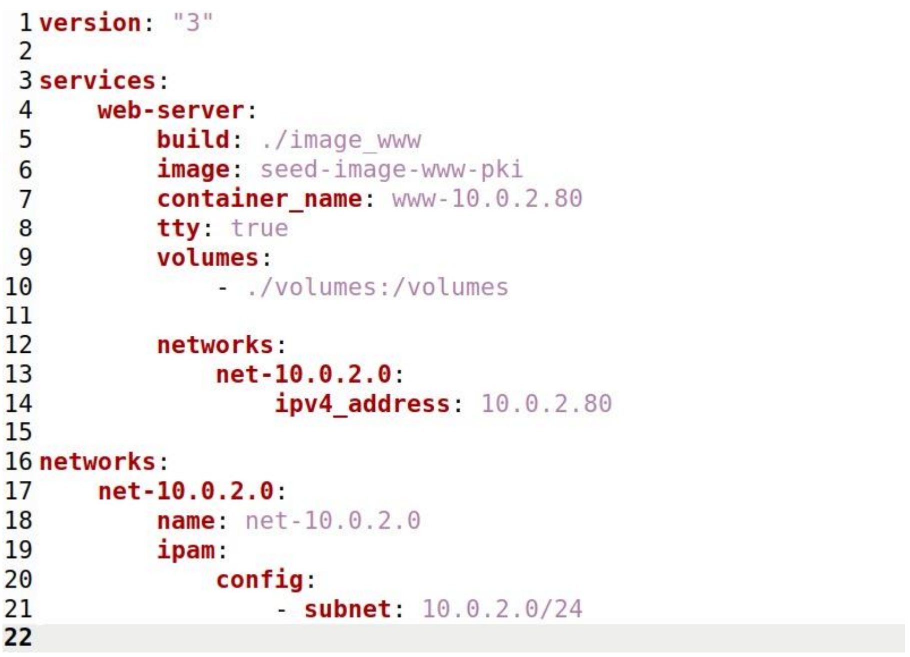
---

2. Environnement de Travail

- **Machine Virtuelle** : SEED Ubuntu 20.04.
- **Outils Utilisés** : OpenSSL, Apache, Docker.
- **Fichiers Importants** : openssl.cnf pour la configuration des certificats.
- **DNS Local** : Fichier `/etc/hosts` modifié pour mapper les noms de domaine.

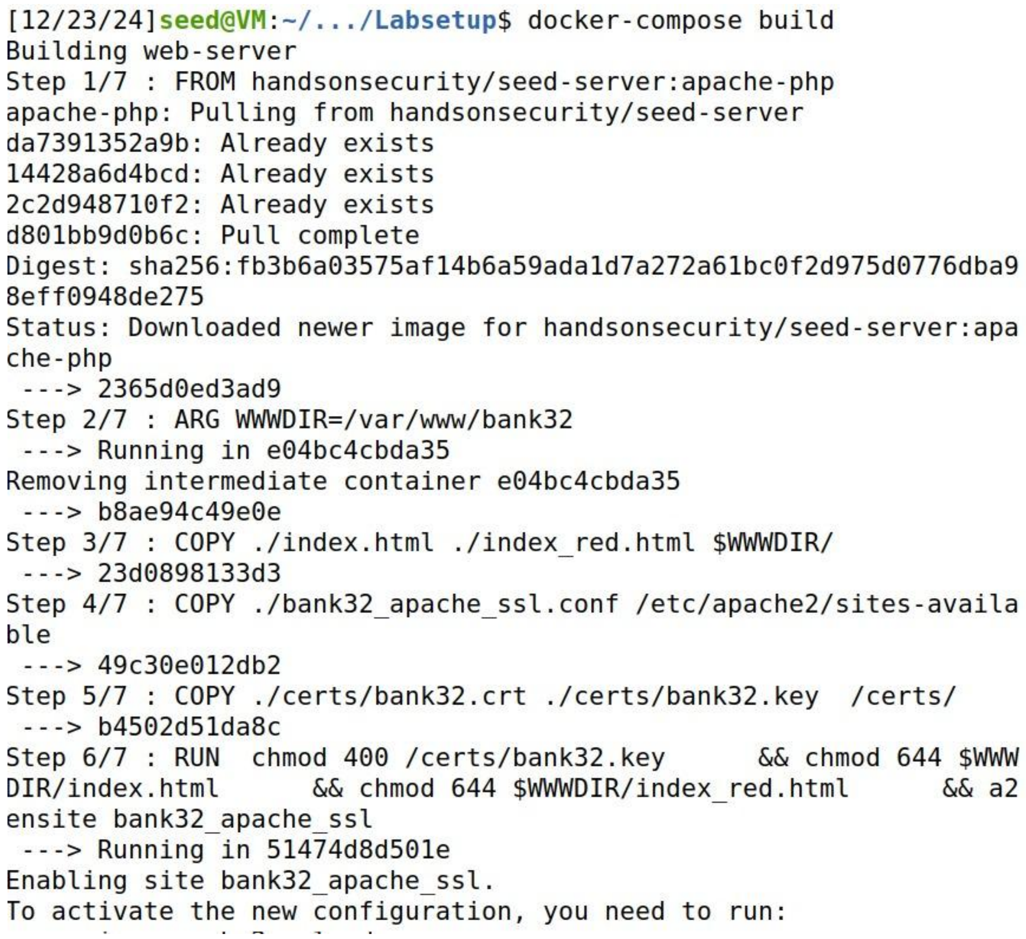
Configuration des Conteneurs

1. **Setup des conteneurs Docker** :
   - Télécharger et décompresser le fichier `Labsetup.zip` depuis le site web du TP.
   - Accéder au répertoire `Labsetup` et utiliser le fichier `docker-compose.yml`



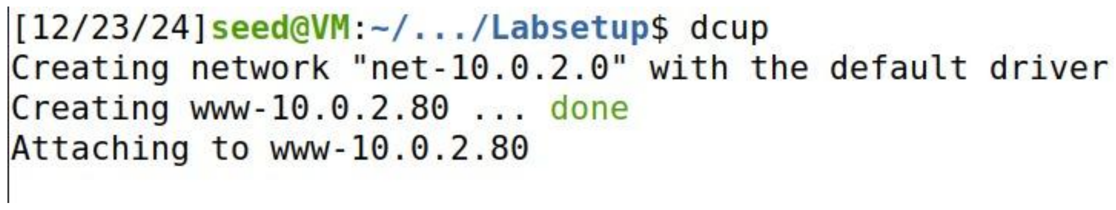
```yaml
version: "3"

services:
    web-server:
        build: ./image_www
        image: seed-image-www-pki
        container_name: www-10.0.2.80
        tty: true
        volumes:
            - ./volumes:/volumes

        networks:
            net-10.0.2.0:
                ipv4_address: 10.0.2.80

networks:
    net-10.0.2.0:
        name: net-10.0.2.0
        ipam:
            config:
                - subnet: 10.0.2.0/24

```

- pour créer l’environnement :
- `$ docker-compose build  # Construction de l'image du conteneur`



```
[12/23/24]seed@VM:~/.../Labsetup$ docker-compose build
Building web-server
Step 1/7 : FROM handsonsecurity/seed-server:apache-php
apache-php: Pulling from handsonsecurity/seed-server
da7391352a9b: Already exists
14428a6d4bcd: Already exists
2c2d948710f2: Already exists
d801bb9d0b6c: Pull complete
Digest: sha256:fb3b6a03575af14b6a59ada1d7a272a61bc0f2d975d0776dba9
8eff0948de275
Status: Downloaded newer image for handsonsecurity/seed-server:apa
che-php
 ---> 2365d0ed3ad9
Step 2/7 : ARG WWWDIR=/var/www/bank32
 ---> Running in e04bc4cbda35
Removing intermediate container e04bc4cbda35
 ---> b8ae94c49e0e
Step 3/7 : COPY ./index.html ./index_red.html $WWWDIR/
 ---> 23d0898133d3
Step 4/7 : COPY ./bank32_apache_ssl.conf /etc/apache2/sites-availa
ble
 ---> 49c30e012db2
Step 5/7 : COPY ./certs/bank32.crt ./certs/bank32.key  /certs/
 ---> b4502d51da8c
Step 6/7 : RUN  chmod 400 /certs/bank32.key      && chmod 644 $WWW
DIR/index.html      && chmod 644 $WWWDIR/index_red.html      && a2
ensite bank32_apache_ssl
 ---> Running in 51474d8d501e
Enabling site bank32_apache_ssl.
To activate the new configuration, you need to run:
```

- `$ docker-compose up     # Lancement du conteneur`



```
[12/23/24]seed@VM:~/.../Labsetup$ dcup
Creating network "net-10.0.2.0" with the default driver
Creating www-10.0.2.80 ... done
Attaching to www-10.0.2.80
```

2. **Interaction avec les conteneurs** :
   - Lister les conteneurs actifs :
   - `$ docker ps`

   Alias :

   ```
   $ dockps  # Alias pour : docker ps --format "{{.ID}}
   {{.Names}}"
   ```

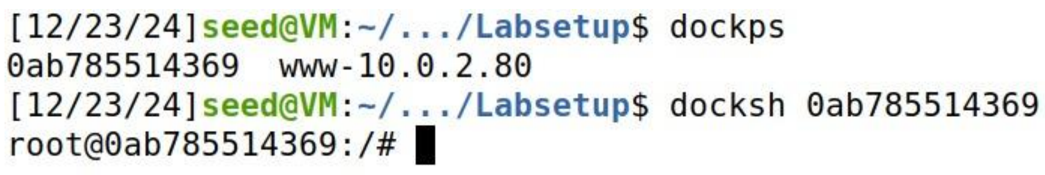
- Accéder à un shell dans un conteneur :
- `$ docker exec -it <ID_du_conteneur> /bin/bash`

  Alias :

```
$ docksh <ID_du_conteneur>
```

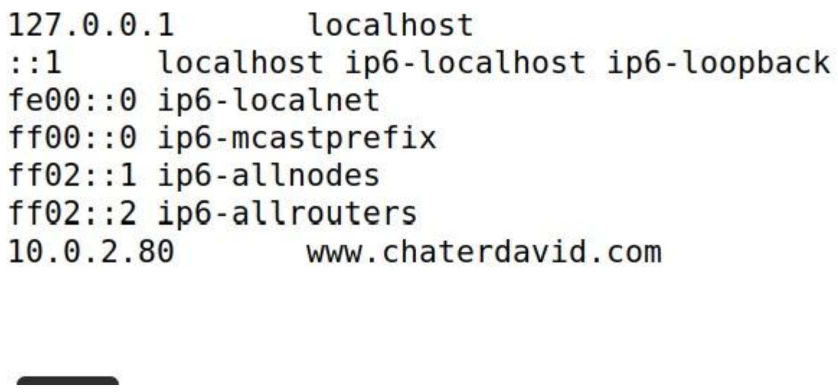
  Exemple :

```
$ dockps
b1004832e275 hostA-10.9.0.5
0af4ea7a3e2e hostB-10.9.0.6
9652715c8e0a hostC-10.9.0.7

$ docksh 965
root@9652715c8e0a:/#
```



```
[12/23/24]seed@VM:~/.../Labsetup$ dockps
0ab785514369  www-10.0.2.80
[12/23/24]seed@VM:~/.../Labsetup$ docksh 0ab785514369
root@0ab785514369:/#
```

3. **Remarque** :
   - Si une commande Docker requiert un ID de conteneur, taper les premiers caractères de l’ID suffit, tant qu’ils sont uniques.
4. **Problèmes courants** :
   - En cas de problèmes lors du setup, consulter la section « Common Problems » du manuel SEED Labs.
5. **Configuration DNS** :

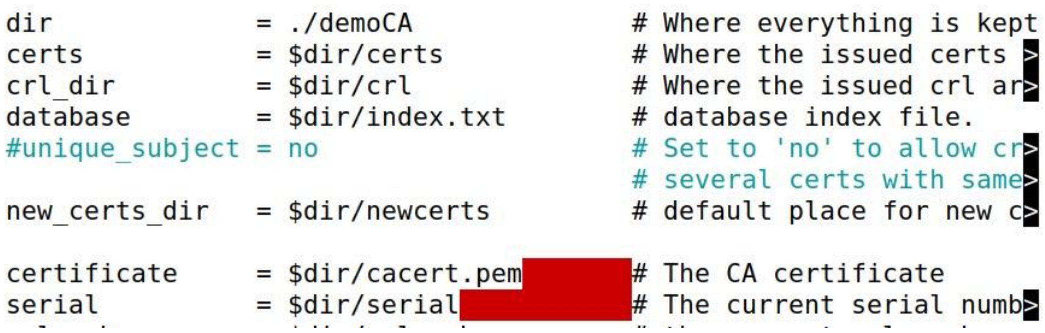


```
127.0.0.1       localhost
::1     localhost ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
10.0.2.80       www.chaterdavid.com
```

3. Tâches Réalisées

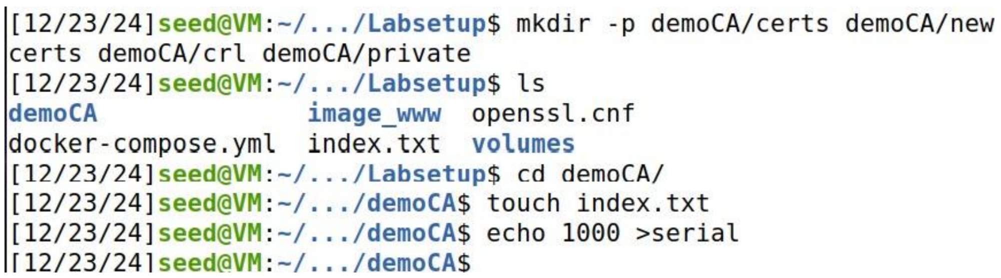
3.1. Devenir une Autorité de Certification (CA)

1. **Objectif** : Créer une CA racine et générer un certificat auto-signé.
2. **Procédure** :
   - Configuration du fichier `openssl.cnf`.



```
dir             = ./demoCA          # Where everything is kept
certs           = $dir/certs        # Where the issued certs >
crl_dir         = $dir/crl          # Where the issued crl ar>
database        = $dir/index.txt    # database index file.
#unique_subject = no                # Set to 'no' to allow cr>
                                    # several certs with same>
new_certs_dir   = $dir/newcerts     # default place for new c>

certificate     = $dir/cacert.pem[redacted]# The CA certificate
serial          = $dir/serial[redacted]# The current serial numb>
```

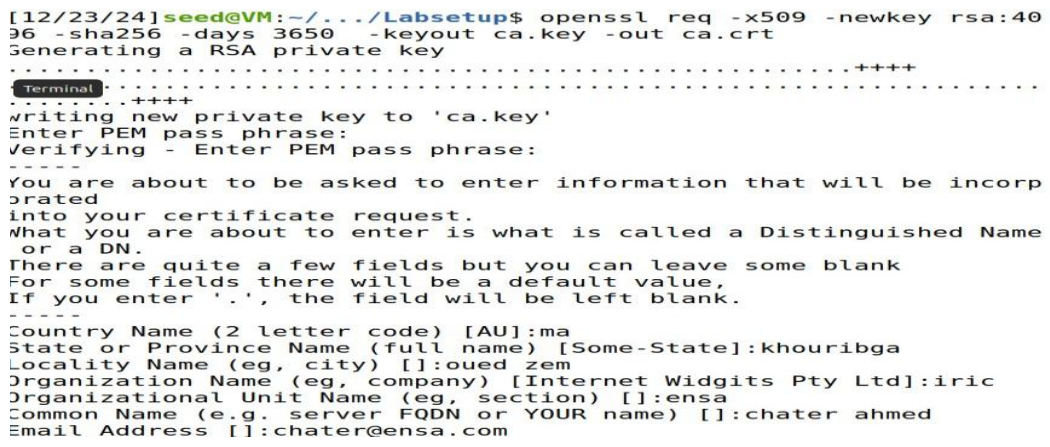
   - Création des répertoires requis (à partir de `demoCA`).



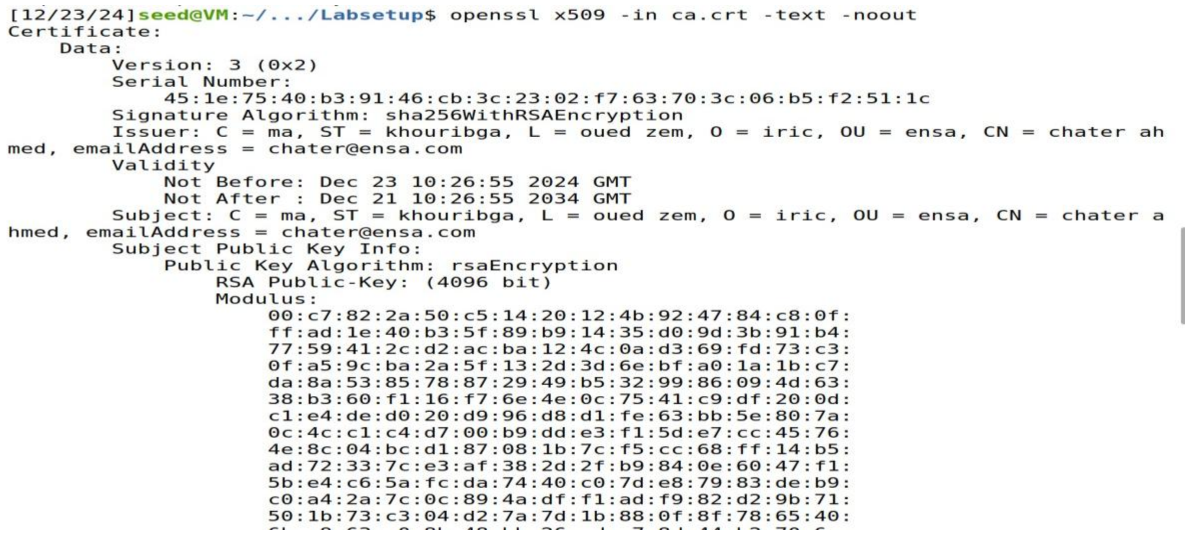
```
[12/23/24]seed@VM:~/.../Labsetup$ mkdir -p demoCA/certs demoCA/new
certs demoCA/crl demoCA/private
[12/23/24]seed@VM:~/.../Labsetup$ ls
demoCA              image_www  openssl.cnf
docker-compose.yml  index.txt  volumes
[12/23/24]seed@VM:~/.../Labsetup$ cd demoCA/
[12/23/24]seed@VM:~/.../demoCA$ touch index.txt
[12/23/24]seed@VM:~/.../demoCA$ echo 1000 >serial
[12/23/24]seed@VM:~/.../demoCA$
```

   - Commande pour générer un certificat auto-signé :
   - `openssl req -x509 -newkey rsa:4096 -sha256 -days 3650 \`
   - `-keyout ca.key -out ca.crt -subj "/CN=www.modelCA.com/O=Model CA LTD./C=US" \`
   - `-passout pass:dees`

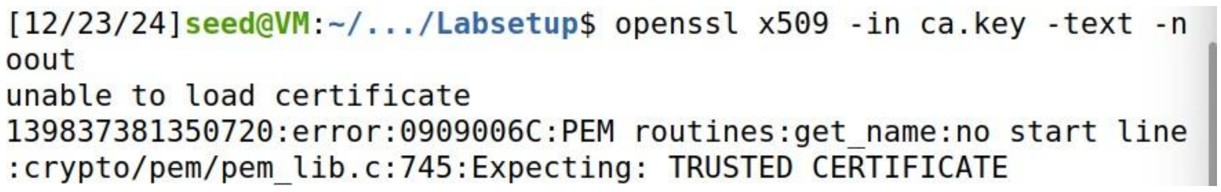
3. **Résultats** :



```
[12/23/24]seed@VM:~/.../Labsetup$ openssl req -x509 -newkey rsa:40
96 -sha256 -days 3650  -keyout ca.key -out ca.crt
Generating a RSA private key
...................................................++++
.................................................................
.......++++
writing new private key to 'ca.key'
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
You are about to be asked to enter information that will be incorp
orated
into your certificate request.
What you are about to enter is what is called a Distinguished Name
 or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:ma
State or Province Name (full name) [Some-State]:khouribga
Locality Name (eg, city) []:oued zem
Organization Name (eg, company) [Internet Widgits Pty Ltd]:iric
Organizational Unit Name (eg, section) []:ensa
Common Name (e.g. server FQDN or YOUR name) []:chater ahmed
Email Address []:chater@ensa.com
```



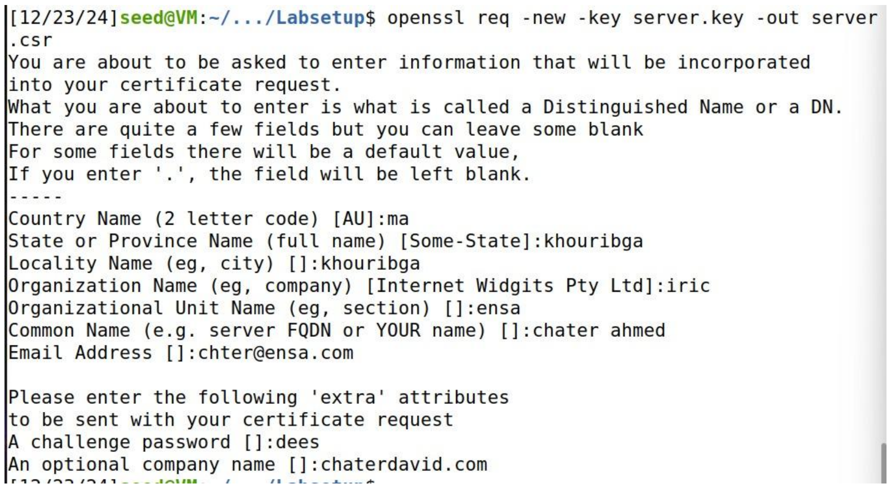
```
[12/23/24]seed@VM:~/.../Labsetup$ openssl x509 -in ca.crt -text -noout
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            45:1e:75:40:b3:91:46:cb:3c:23:02:f7:63:70:3c:06:b5:f2:51:1c
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C = ma, ST = khouribga, L = oued zem, O = iric, OU = ensa, CN = chater ah
med, emailAddress = chater@ensa.com
        Validity
            Not Before: Dec 23 10:26:55 2024 GMT
            Not After : Dec 21 10:26:55 2034 GMT
        Subject: C = ma, ST = khouribga, L = oued zem, O = iric, OU = ensa, CN = chater a
hmed, emailAddress = chater@ensa.com
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                RSA Public-Key: (4096 bit)
                Modulus:
                    00:c7:82:2a:50:c5:14:20:12:4b:92:47:84:c8:0f:
                    ff:ad:1e:40:b3:5f:89:b9:14:35:d0:9d:3b:91:b4:
                    77:59:41:2c:d2:ac:ba:12:4c:0a:d3:69:fd:73:c3:
                    0f:a5:9c:ba:2a:5f:13:2d:3d:6e:bf:a0:1a:1b:c7:
                    da:8a:53:85:78:87:29:49:b5:32:99:86:09:4d:63:
                    38:b3:60:f1:16:f7:6e:4e:0c:75:41:c9:df:20:0d:
                    c1:e4:de:d0:20:d9:96:d8:d1:fe:63:bb:5e:80:7a:
                    0c:4c:c1:c4:d7:00:b9:dd:e3:f1:5d:e7:cc:45:76:
                    4e:8c:04:bc:d1:87:08:1b:7c:f5:cc:68:ff:14:b5:
                    ad:72:33:7c:e3:af:38:2d:2f:b9:84:0e:60:47:f1:
                    5b:e4:c6:5a:fc:da:74:40:c0:7d:e8:79:83:de:b9:
                    c0:a4:2a:7c:0c:89:4a:df:f1:ad:f9:82:d2:9b:71:
                    50:1b:73:c3:04:d2:7a:7d:1b:88:0f:8f:78:65:40:
```



```
[12/23/24]seed@VM:~/.../Labsetup$ openssl x509 -in ca.key -text -n
oout
unable to load certificate
139837381350720:error:0909006C:PEM routines:get_name:no start line
:crypto/pem/pem_lib.c:745:Expecting: TRUSTED CERTIFICATE
```

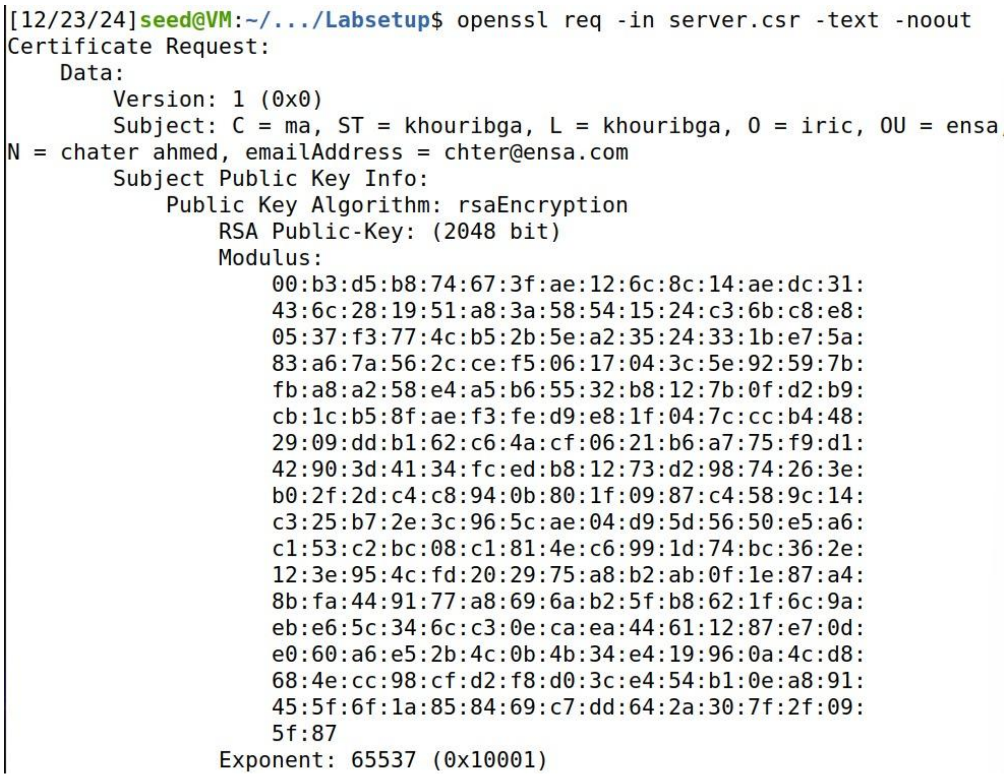
3.2. Génération d'une Requête de Signature de Certificat (CSR)

1. **Objectif** : Générer une CSR pour un serveur web (ex. www.chater2024.com).
2. **Procédure** :
   - Commande :
   - `openssl req -newkey rsa:2048 -sha256 \`
   - `-keyout server.key -out server.csr \`
   - `-subj "/CN=www.chater2024.com/O=Chater Inc./C=US" \`
   - `-passout pass:dees`
   - Ajout de noms alternatifs via `-addext`.

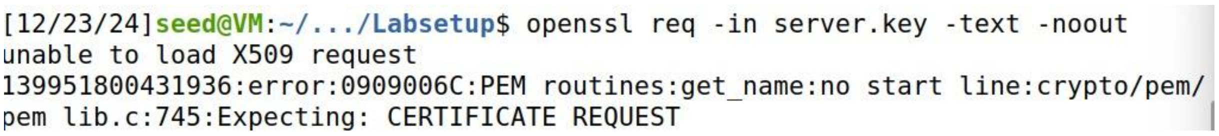


```
[12/23/24]seed@VM:~/.../Labsetup$ openssl req -new -key server.key -out server
.csr
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:ma
State or Province Name (full name) [Some-State]:khouribga
Locality Name (eg, city) []:khouribga
Organization Name (eg, company) [Internet Widgits Pty Ltd]:iric
Organizational Unit Name (eg, section) []:ensa
Common Name (e.g. server FQDN or YOUR name) []:chater ahmed
Email Address []:chter@ensa.com

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:dees
An optional company name []:chaterdavid.com
```

3. **Résultats** : Capture d'écran des fichiers `server.key` et `server.csr`.



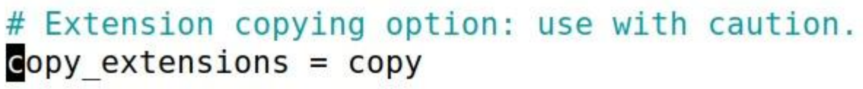
```
[12/23/24]seed@VM:~/.../Labsetup$ openssl req -in server.csr -text -noout
Certificate Request:
    Data:
        Version: 1 (0x0)
        Subject: C = ma, ST = khouribga, L = khouribga, O = iric, OU = ensa
N = chater ahmed, emailAddress = chter@ensa.com
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                RSA Public-Key: (2048 bit)
                Modulus:
                    00:b3:d5:b8:74:67:3f:ae:12:6c:8c:14:ae:dc:31:
                    43:6c:28:19:51:a8:3a:58:54:15:24:c3:6b:c8:e8:
                    05:37:f3:77:4c:b5:2b:5e:a2:35:24:33:1b:e7:5a:
                    83:a6:7a:56:2c:ce:f5:06:17:04:3c:5e:92:59:7b:
                    fb:a8:a2:58:e4:a5:b6:55:32:b8:12:7b:0f:d2:b9:
                    cb:1c:b5:8f:ae:f3:fe:d9:e8:1f:04:7c:cc:b4:48:
                    29:09:dd:b1:62:c6:4a:cf:06:21:b6:a7:75:f9:d1:
                    42:90:3d:41:34:fc:ed:b8:12:73:d2:98:74:26:3e:
                    b0:2f:2d:c4:c8:94:0b:80:1f:09:87:c4:58:9c:14:
                    c3:25:b7:2e:3c:96:5c:ae:04:d9:5d:56:50:e5:a6:
                    c1:53:c2:bc:08:c1:81:4e:c6:99:1d:74:bc:36:2e:
                    12:3e:95:4c:fd:20:29:75:a8:b2:ab:0f:1e:87:a4:
                    8b:fa:44:91:77:a8:69:6a:b2:5f:b8:62:1f:6c:9a:
                    eb:e6:5c:34:6c:c3:0e:ca:ea:44:61:12:87:e7:0d:
                    e0:60:a6:e5:2b:4c:0b:4b:34:e4:19:96:0a:4c:d8:
                    68:4e:cc:98:cf:d2:f8:d0:3c:e4:54:b1:0e:a8:91:
                    45:5f:6f:1a:85:84:69:c7:dd:64:2a:30:7f:2f:09:
                    5f:87
                Exponent: 65537 (0x10001)
```



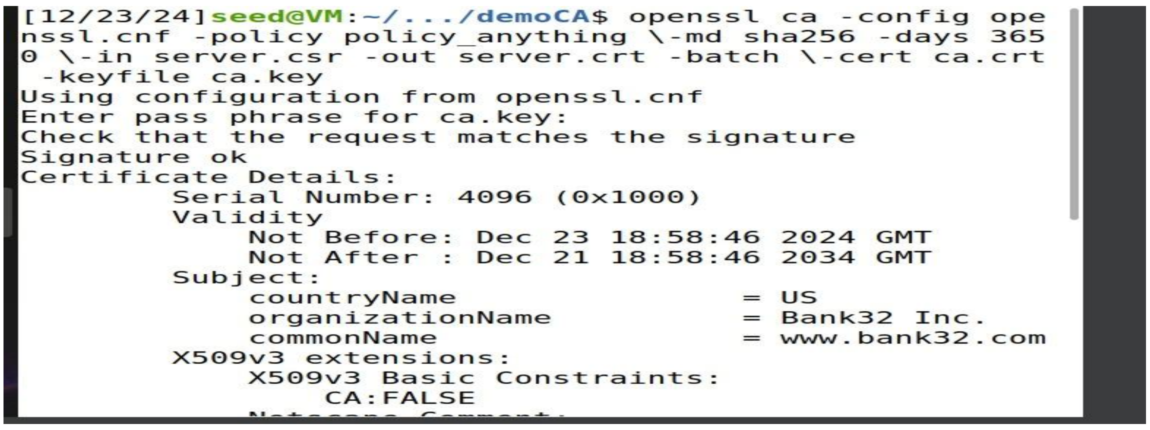
```
[12/23/24]seed@VM:~/.../Labsetup$ openssl req -in server.key -text -noout
unable to load X509 request
139951800431936:error:0909006C:PEM routines:get_name:no start line:crypto/pem/
pem lib.c:745:Expecting: CERTIFICATE REQUEST
```

3.3. Génération d’un Certificat pour le Serveur

1. **Objectif** : Signer la CSR avec la CA pour obtenir un certificat X509.
2. **Procédure** :
   - Modification du fichier `openssl.cnf` pour copier les extensions.
   - Commande pour signer :
   - `openssl ca -config openssl.cnf -policy policy_anything \`
   - `-md sha256 -days 3650 -in server.csr -out server.crt -batch \`
   - `-cert ca.crt -keyfile ca.key`

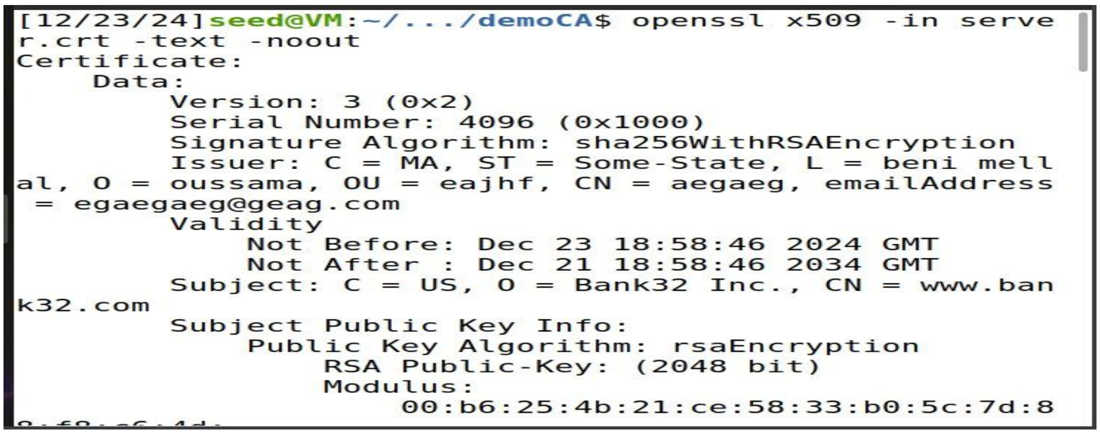


```
# Extension copying option: use with caution.
copy_extensions = copy
```



```
[12/23/24]seed@VM:~/.../demoCA$ openssl ca -config ope
nssl.cnf -policy policy_anything \-md sha256 -days 365
0 \-in server.csr -out server.crt -batch \-cert ca.crt
 -keyfile ca.key
Using configuration from openssl.cnf
Enter pass phrase for ca.key:
Check that the request matches the signature
Signature ok
Certificate Details:
        Serial Number: 4096 (0x1000)
        Validity
            Not Before: Dec 23 18:58:46 2024 GMT
            Not After : Dec 21 18:58:46 2034 GMT
        Subject:
            countryName               = US
            organizationName          = Bank32 Inc.
            commonName                = www.bank32.com
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
```

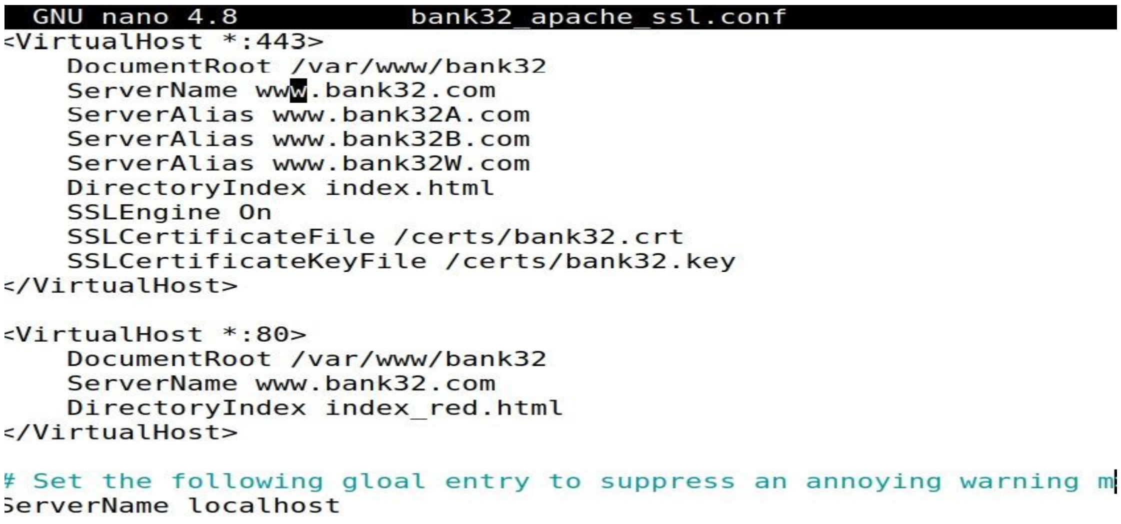
3. **Résultats** : Validation des noms alternatifs via `openssl x509 -text`



```
[12/23/24]seed@VM:~/.../demoCA$ openssl x509 -in serve
r.crt -text -noout
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 4096 (0x1000)
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C = MA, ST = Some-State, L = beni mell
al, O = oussama, OU = eajhf, CN = aegaeg, emailAddress
 = egaegaeg@geag.com
        Validity
            Not Before: Dec 23 18:58:46 2024 GMT
            Not After : Dec 21 18:58:46 2034 GMT
        Subject: C = US, O = Bank32 Inc., CN = www.ban
k32.com
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                RSA Public-Key: (2048 bit)
                Modulus:
                    00:b6:25:4b:21:ce:58:33:b0:5c:7d:8
```

3.4. Déploiement d'un Site HTTPS

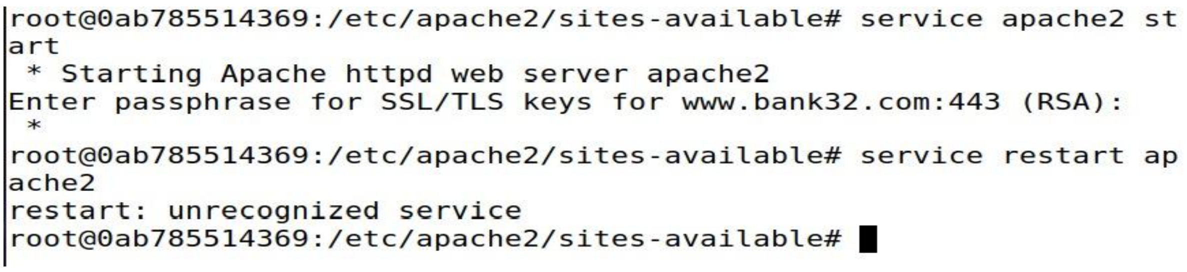
1. **Objectif** : Configurer un site web HTTPS avec Apache.
2. **Procédure** :
    - Configuration de `bank32_apache_ssl.conf` avec :
    - `<VirtualHost *:443>`
    - `DocumentRoot /var/www/chater2024`
    - `ServerName www.chater2024.com`
    - `SSLEngine On`
    - `SSLCertificateFile /certs/server.crt`
    - `SSLCertificateKeyFile /certs/server.key`
    - `</VirtualHost>`
    - Activation des modules SSL : `a2enmod ssl`.
    - Lancement du serveur : `service apache2 start`.
3. **Résultats** : Validation de la connexion HTTPS avec un navigateur.



```
  GNU nano 4.8              bank32_apache_ssl.conf
<VirtualHost *:443>
    DocumentRoot /var/www/bank32
    ServerName www.bank32.com
    ServerAlias www.bank32A.com
    ServerAlias www.bank32B.com
    ServerAlias www.bank32W.com
    DirectoryIndex index.html
    SSLEngine On
    SSLCertificateFile /certs/bank32.crt
    SSLCertificateKeyFile /certs/bank32.key
</VirtualHost>

<VirtualHost *:80>
    DocumentRoot /var/www/bank32
    ServerName www.bank32.com
    DirectoryIndex index_red.html
</VirtualHost>

# Set the following gloal entry to suppress an annoying warning m
ServerName localhost
```

3.5. Attaque Man-In-The-Middle (MITM)

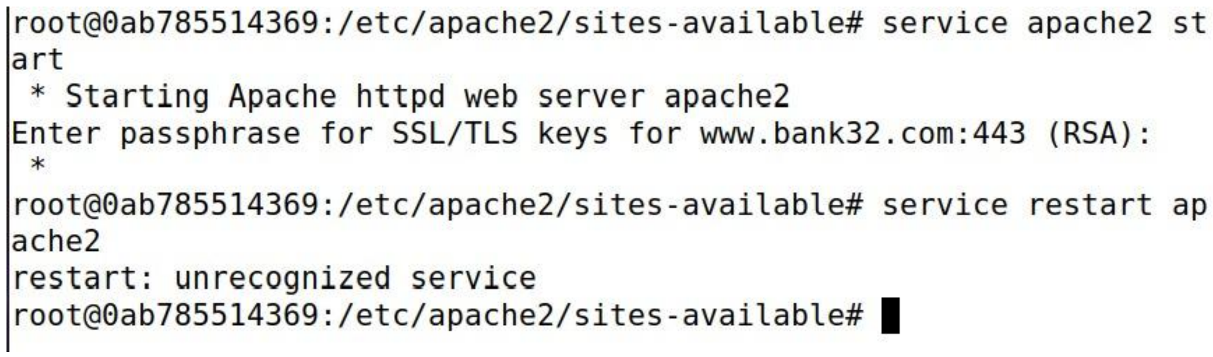
1. **Objectif** : Simuler une attaque MITM sur un site HTTPS.
2. **Procédure** :
    - Modification du fichier `/etc/hosts` pour rediriger www.example.com vers le serveur malveillant.
    - Configuration d'un faux site web.
3. **Résultats** : Analyse des réactions du navigateur face au certificat non valide.



```
root@0ab785514369:/etc/apache2/sites-available# service apache2 st
art
 * Starting Apache httpd web server apache2
Enter passphrase for SSL/TLS keys for www.bank32.com:443 (RSA):
 *
root@0ab785514369:/etc/apache2/sites-available# service restart ap
ache2
restart: unrecognized service
root@0ab785514369:/etc/apache2/sites-available# █
```

3.6. Attaque MITM avec une CA Compromise

1. **Objectif** : Démontrer l’impact d’une CA compromise.
2. **Procédure** :
   - Utilisation de la clé privée de la CA pour générer un certificat pour un faux site.
   - Test de connexion avec le faux certificat ajouté comme émetteur approuvé.
3. **Résultats** : Succès de l’attaque sans avertissement du navigateur.



```
root@0ab785514369:/etc/apache2/sites-available# service apache2 st
art
 * Starting Apache httpd web server apache2
Enter passphrase for SSL/TLS keys for www.bank32.com:443 (RSA):
 *
root@0ab785514369:/etc/apache2/sites-available# service restart ap
ache2
restart: unrecognized service
root@0ab785514369:/etc/apache2/sites-available# 
```

---

4. Conclusion

Ce TP nous a permis de comprendre les concepts fondamentaux de PKI et leur mise en œuvre pratique. Les tests ont montré comment les certificats protègent les communications tout en mettant en évidence les vulnérabilités possibles en cas de compromis de la CA.

.