Pr. Saiida LAZAAR
ENSA de Tanger
Département Mathématiques & Informatique

TP: Création de clés de chiffrement et de certificats avec openSSL
Signature de documents et sécurisation de sites web

1. Brève introduction sur openssl

openSSL est une boite à outils cryptographiques implémentant les protocoles SSL et TLS avec une bibliothèque de programmation en **C** qui réalise des applications client/serveur sécurisées à l'aide de la commande openssl ; cette commande permet de créer des clés de chiffrement comme RSA, des certificats (X509), des empreintes MD5, SHA, etc. **openssl** permet le chiffrement et déchiffrement (DES, IDEA, RC4, etc.). Pour plus de détails, vous êtes invités à visiter le site https://www.**openssl**.org/

Plan du TP : On utilisera openssl pour réaliser les tests suivants :

- Chiffrement et déchiffrement d'un fichier donné.
- Génération de clés de cryptage.
- Signature des documents avec des fonctions de hachage. Familiarisation avec ces notions.
- Création de certificats.
- Sécurisation d'un site web avec le protocole https.

Ci-dessous, un guide et quelques questions pour vous aider à réaliser ce tp.

Tout d'abord, la commande **man openssl** permet de connaitre toutes les fonctionnalités de openssl.

Chiffrement RSA avec openSSL

Génération d'une paire de clés

On peut générer une paire de clés RSA avec la commande **genrsa** de openSSL.

```
$ openssl genrsa -out <fichier> <taille>
```

Par exemple, pour générer une paire de clés de 1024 bits, stockée dans le fichier **maCle.pem**, on tape la commande

```
$ openssl genrsa -out maCle.pem 1024
```

Le fichier obtenu est un fichier au format PEM (Privacy Enhanced Mail, format en base 64), dont voici un exemple

```
$ cat maCle.pem
-----BEGIN RSA PRIVATE KEY-----
MIICXAIBAAKBgQCveVjLltevTC5kSAiTYjHMVuAR80DHMLWCp3BOVZ49eXwraXxO
7AfKWpA5g0wFZgZNERIfFYaCnvaQDQA+9BRIfsSSr3oSw0My5SD6eg15v0VmJmvP
d8LgBypJHbr6f5MXWqntvzp0Qvg6ddeNpUIrqkkh4uDfHFDWqyrkQUCvKwIDAQAB
AoGANchUrfnq28DWy0fE0R+cscvC292Z8jN8vrIBWxEk8iSlKU0om6v+a0g8wlP6
3gC6V66uxjY7xxdf7SD+/UykVl4PGFymhLtywSdGlgec3tLgBtV3ytJFilAVDBij
LzQwUegCO4zt1JWYc6vvaVdNyQSaGIIeYGsNDWEYlOtDSlkCQQDVRn9JS15G8p+H
4Z0PbU9ZQg2L1u9/SD/kELVe3Kx1fdHulxH0v8V2AgPdXA29Nhi+TxUtC+V8CMc2
KXmAvFsHAkEA0qBDmjHMDPwcGaqbQ2lymYQIGlZ5TLQFA98Dey2uE+CB6pmS/e/Z
ilu1IaasuE3vBzXfB/JU7DUkV++JQ7TtvQJBAL2s5dUch2sXqlOhjhpDP/eE7CE6
9WLAsbm2Nmd4YJRZYtQLXPfLeeSapC9BCCMHsnfGQ3H9i4mFEQ6VUi7w1Q8CQAQa
pVaS09QI8Y86eM4GdvowzWud9b0d4N8jcFDtIfA3NrDYjzmte8KraMsgEUuCET9F
uHPSL/9uRagE/dq44s0CQCMQU4PMqkMtwzCFsV8ZqLmkDPn1binIAwRLYFcsQRDt
gTi6rycz3Pk1hCVzBfyMd8zwqpwKmR5FoOXuJEv+mVg=
-----END RSA PRIVATE KEY-----
```

Visualisation des clés RSA

La commande **rsa** permet de visualiser le contenu d'un fichier au format PEM contenant une paire de clés RSA.

```
$ openssl rsa -in <fichier> -text -noout
```

L'option **-text** demande l'affichage décodé de la paire de clés. L'option **-noout** supprime la sortie normalement produite par la commande rsa.

Par exemple

```
$ openssl rsa -in maCle.pem -text -noout
Private-Key: (1024 bit)
modulus:
00:af:79:58:cb:96:d7:af:4c:2e:64:48:08:93:62:
31:cc:56:e0:11:f3:40:c7:30:b5:82:a7:70:4e:55:
9e:3d:79:7c:2b:69:7c:4e:ec:07:ca:5a:90:39:83:
4c:05:66:06:4d:11:12:1f:15:86:82:9e:f6:90:0d:
00:3e:f4:14:48:7e:c4:92:af:7a:12:c3:43:32:e5:
20:fa:7a:0d:79:bf:45:66:26:6b:cf:77:c2:e0:07:
2a:49:1d:ba:fa:7f:93:17:5a:a9:ed:bf:3a:74:42:
f8:3a:75:d7:8d:a5:42:2b:aa:49:21:e2:e0:df:1c:
50:d6:ab:2a:e4:41:40:af:2b
publicExponent: 65537 (0x10001)
privateExponent:
35:c8:54:ad:f9:ea:db:c0:d6:cb:47:c4:d1:1f:9c:
```

```
b1:cb:c2:db:dd:99:f2:33:7c:be:b2:01:5b:11:24:
f2:24:a5:29:4d:28:9b:ab:fe:6b:48:3c:c2:53:fa:
de:00:ba:57:ae:ae:c6:36:3b:c7:17:5f:ed:20:fe:
fd:4c:a4:56:5e:0f:18:5c:a6:84:bb:72:c1:27:46:
96:07:9c:de:d2:e0:06:d5:77:ca:d2:45:8a:50:15:
0c:18:a3:2f:34:30:51:e8:02:3b:8c:ed:d4:95:98:
73:ab:ef:69:57:4d:c9:04:9a:18:82:1e:60:6b:0d
0d:61:18:94:eb:43:4a:59
prime1:
00:d5:46:7f:49:4b:5e:46:f2:9f:87:e1:9d:0f:6d:
4f:59:42:0d:8b:d6:ef:7f:48:3f:e4:10:b5:5e:dc:
ac:75:7d:d1:ee:97:11:f4:bf:c5:76:02:03:dd:5c:
0d:bd:36:18:be:4f:15:2d:0b:e5:7c:08:c7:36:29:
79:80:bc:5b:07
prime2:
00:d2:a0:43:9a:31:cc:0c:fc:1c:19:aa:9b:43:69:
72:99:84:08:1a:56:79:4c:b4:05:03:df:03:7b:2d:
ae:13:e0:81:ea:99:92:fd:ef:d9:8a:5b:b5:21:a6:
ac:b8:4d:ef:07:35:df:07:f2:54:ec:35:24:57:ef:
89:43:b4:ed:bd
exponent1:
00:bd:ac:e5:d5:1c:87:6b:17:aa:53:a1:8e:1a:43:
3f:f7:84:ec:21:3a:f5:62:c0:b1:b9:b6:36:67:78:
60:94:59:62:d4:0b:5c:f7:cb:79:e4:9a:a4:2f:41:
08:23:07:b2:77:c6:43:71:fd:8b:89:85:11:0e:95:
52:2e:f0:d5:0f
exponent2:
04:1a:a5:56:92:d3:d4:08:f1:8f:3a:78:ce:06:76:
fa:30:cd:6b:9d:f5:bd:1d:e0:df:23:70:50:ed:21:
f0:37:36:b0:d8:8f:39:ad:7b:c2:ab:68:cb:20:11:
4b:82:11:3f:45:b8:73:d2:2f:ff:6e:45:a8:04:fd:
da:b8:e2:cd
coefficient:
23:10:53:83:cc:aa:43:2d:c3:30:85:b1:5f:19:a8:
b9:a4:0c:f9:f5:6e:29:c8:03:04:4b:60:57:2c:41:
10:ed:81:38:ba:af:27:33:dc:f9:35:84:25:73:05:
fc:8c:77:cc:f0:aa:9c:0a:99:1e:45:a0:e5:ee:24:
4b:fe:99:58
```

Les différents éléments de la clé sont affichés en hexadécimal. On peut distinguer le modulus, l'exposant public (qui par défaut est toujours 65537[1]), l'exposant privé, les nombres premiers facteurs du modulus, plus trois autres nombres qui servent à optimiser l'algorithme de déchiffrement.

Exercice 1 : Donnez une explication du choix de la valeur **65537** pour exposant public par défaut.

Chiffrement d'un fichier de clés RSA

Trois options sont possibles qui précisent l'algorithme de chiffrement symétrique à utiliser : **-des, -des3 et -idea**.

```
$ openssl rsa -in maCle.pem -des3 -out maCle.pem
```

```
writing RSA key
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
```

Une phrase de passe est demandée deux fois pour générer une clé symétrique protégeant l'accès à la clé.

Exercice 2 : Avec la commande **cat** observez le contenu du fichier maCle.pem.
Utilisez à nouveau la commande rsa pour visualiser le contenu de la clé.

Exportation de la partie publique

La partie publique d'une paire de clés RSA est publique. Le fichier maCle.pem contient la partie privée de la clé. Avec l'option –**pubout** on peut exporter la partie publique de la clé.

```
$ openssl rsa -in maCle.pem -pubout -out maClePublique.pem
```

Exercice 3 :

Q 1 . Notez le contenu du fichier maClePublique.pem. Remarquez les marqueurs de début et de fin.
Q 2 . Avec la commande rsa visualisez la clé publique. Vous devez préciser l'option **-pubin**, puisque seule la partie publique figure dans le fichier maClePublique.pem.

Chiffrement/déchiffrement de données avec RSA

On peut chiffrer des données avec une clé RSA. Pour cela on utilise la commande **rsautl**

```
$ openssl rsautl -encrypt -in <fichier entree> -inkey <cle> -out <fic
hier sortie>
```

où
- fichier entree est le fichier des données à chiffrer. Le fichier des données à chiffrer ne doit pas avoir une taille excessive (ne doit pas dépasser 116 octets pour une clé de 1024 bits).
- cle est le fichier contenant la clé RSA. Si ce fichier ne contient que la parte publique de la clé, il faut rajouter l'option -pubin.
- fichier sortie est le fichier de données chiffré.

Pour déchiffrer on remplace l'option -encrypt par -decrypt. Le fichier contenant la clé doit obligatoirement contenir la partie privée.

Exercice 4 : Chiffrez le fichier de votre choix avec le système symétrique de votre choix. Chiffrez la clé ou le mot de passe avec la clé publique de votre destinataire. Envoyez-lui le mot de passe chiffré ainsi que le fichier chiffré.

Signature de fichiers

Il n'est possible de signer que de petits documents. Pour signer un gros document on calcule d'abord une empreinte de ce document. La commande **dgst** permet de le faire.

```
$ openssl dgst <hachage>  -out <empreinte> <fichier entree>
```

où hachage est une fonction de hachage.

Signer un document revient à signer son empreinte. Pour cela, on utilise l'option -sign de la commande rsautl

```
$ openssl rsautl –sign–in <empreinte> –inkey <cle>  –out <signature>
```

et pour vérifier la signature

```
$ openssl rsautl –verify   –in <signature> –pubin –inkey <cle>
–out <empreinte>
```

il reste ensuite à vérifier que l'empreinte ainsi produite est la même que celle que l'on peut calculer. L'option -pubin indique que la clé utilisée pour la vérification est la partie publique de la clé utilisée pour la signature.

Exercice 5 : Signez le fichier de votre choix, puis vérifiez la signature.

Certificats

Vous allez élaborer un certificat pour votre clé publique. Puis, vous verrez comment utiliser les clés certifiées pour signer et/ou chiffrer des courriers électoniques.

Génération de la paire de clés

Exercice 6 : Générez votre paire de clés RSA d'une taille de 1024 bits, protégée par un mot de passe. Dans la suite, on suppose nommé maCle.pem le fichier contenant votre paire de clés RSA. Ce fichier est protégé par le mot de passe fourni lors de la génération.

Exercice 7 : Créez un fichier ne contenant que la partie publique de votre clé RSA. Dans la suite, on suppose ce fichier nommé **maClePublique.pem**.

Création d'une requête de certificats

Maintenant que vous disposez d'une clé RSA, vous allez établir une requête pour obtenir un certificat. Outre la clé publique du sujet, un certificat contient un certain nombre d'informations concernant l'identité de son propriétaire et qui sont demandées lors de la création de la requête. Un fichier de configuration (req.cnf) peut-être défini qui propose les informations à apporter dans le certificat avec des valeurs par défaut.

On établit une requête avec la commande req de openssl.

```
$ openssl req –config req.cnf –new –key maCle.pem –out maRequete.pem
```

Le fichier produit maRequete.pem est aussi au format PEM.

```
$ cat maRequete.pem
```

On peut consulter les informations contenues dans la requête avec la commande

Exercice 8 : Expliquez cette requête. La privée du sujet y figure–t–elle ?

Demande de signature de certificat

Une fois que vous avez établi une requête de certificat, il reste à contacter une autorité de certification qui vous donnera un certificat signé.

Un certificat produit par openssl est un fichier au format PEM.

```
$ cat unCertif.pem
-----BEGIN CERTIFICATE-----

-----END CERTIFICATE-----
```

Pour visualiser le contenu d'un certificat

```
$ openssl x509 -in unCertif.pem -text -noout
```

Exercice 9 : Après avoir récupéré le certificat de l'autorité, ainsi que sa paire de clés RSA, cherchez quelle est la date d'expiration du certificat et la taille de la clé.

Création d'un certificat : Pour créer et signer un certificat à partir d'une requête maRequete.pem, l'autorité invoque la commande x509

Vérification de certificats : On peut vérifier la validité d'un certificat avec la commande **verify**. Pour vérifier la validité d'un certificat, il est nécessaire de disposer du certificat de l'autorité qui l'a émis.