

LAB Cryptographie

U7 CYBER

SPINELLI Dylan
BTS SIO SISR | PARIS YNOV CAMPUS

Table des matières

1-	Introduction à OpenSSL	2
2-	Cryptographie Asymétrique.....	5
3-	Chiffrement d'un fichier volumineux	7
4-	Signature électronique.....	8

Nous allons désormais générer une paire de clés de cryptage pour Alice :

```
(root@kali)-[~/home/kali/Documents/LAB1/Alice]
└─# openssl genrsa -out AliceKeyPair

(root@kali)-[~/home/kali/Documents/LAB1/Alice]
└─# ls
AliceDocument  AliceKeyPair

(root@kali)-[~/home/kali/Documents/LAB1/Alice]
└─#
```

On sépare la clé publique en la renommant AlicePublicKey :

```
(root@kali)-[~/home/kali/Documents/LAB1/Alice]
└─# openssl rsa -in AliceKeyPair -pubout -out AlicePublicKey
writing RSA key

(root@kali)-[~/home/kali/Documents/LAB1/Alice]
└─# ls
AliceDocument  AliceKeyPair  AlicePublicKey

(root@kali)-[~/home/kali/Documents/LAB1/Alice]
└─#
```

On renomme la clé privée AlicePrivateKey :

```
(root@kali)-[~/home/kali/Documents/LAB1/Alice]
└─# mv AliceKeyPair AlicePrivateKey
```

Voici la clé publique de Alice :

```
root@kali: /home/kali/Documents/LAB1/Alice
Session Actions Edit View Help
GNU nano 8.7.1 AlicePublicKey
-----BEGIN PUBLIC KEY-----
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAz+K3a9MW/nyxQ2W9fkSi
pL3cdBb80km6P4ekCMqF4r4vVeNR8CXuPsPR3im96KBagJ6PLV7Ffli9VGwl8I+6
aSjopLAVijKNwyxFUu+A85qZdWJBsqd9GKEj8Loi/HjfmoeLdIQYLT01kKqf2iQ
WK0qlTJCEn8+nE8t5iyRCFux/+CBKztnVxrOC7ZnrNb+cE9PoIQVfmxB/A72XjWb
vp5u+5eANo5LuULxUUQOrMLL+mgUNKQIn+8TiumFSfFBvGoEBx14RrcDK58zcoDH
a7X/T/3sRGIDQNmg97dyTh0Z/dsTapjAs4k2twCt+rp5i3H0KDtPEGj4Ccnfp9KM
+wIDAQAB
-----END PUBLIC KEY-----
```

Voici la clé privée de Alice :

```
root@kali: /home/kali/Documents/LAB1/Alice
Session Actions Edit View Help
GNU nano 8.7.1 AlicePrivateKey
-----BEGIN PRIVATE KEY-----
MIIEvAIBADANBgkqhkiG9w0BAQEFAASCbKYYggSiAgEAAoIBAQDP4rdr0xb+FLFD
Zb1+RKKkvdX0FvW6Sbo/h6QIyoXivi9V41HwJe4+w9HeKb3ooFqAno8tXsXGWL1U
bCXwj7ppK0iksBWKMo3DLEVS74DzmpL1YkGyqQP0YoSPwuiL8eN+ah4t0hBiVPTW
Qqp/aJBYSrSqVMkISfz6cTy3mLJEIW7H/4IErO2dXGs4Ltmes1v5Wt0+ghBUWbEH8
DvZeNZu+nm77L4A2jku5QvFRSo6sWSv6aBQ0pAif7xOK6YVJ8UG8agQHHXhGtwMr
nzNygMdrftf9P/exEYgNA2aD3t3JOHRn92xNqmMCziTa3AK36unmLcc4o008QaPgJ
yd+n0oz7AgMBAAEcggEACJyHPv7/8gucR2k/6XeFltyk9Mu9Ww5ezSic5QUh33vd
Xh0MJRSNKZBwpL4gUP8DgLx3Iq8zjIna6rpbWZTtz3/Gubu8yK+Onv0gfTtqoUZQ
NEQ26hdSQ9D1vA0cIN/xeo61MIfsVQZyT5KnZjiHC9IjwQmgVXMFiiTfIKyYvW3S
QTxJZzuzG1ogiutKCuWbr0vkZYFC0hQTVcrpB99qMNNUh4chrqQLKXLYRb0L/cX9
5b9I3c+HJY0n1RiFu+H14L3/dje0qBYK7b7qh4V1MLG94PpxudpUsJbRgEJDDU7C
JNZS9gxsqTOAF8WQfsi4gIGk1829TlhK3fB9ZonagQKBgQDrRBoN69/FfL+AsyRA
Jds9Qsogbtq9lkp7V7EHdoSZRCr7nwKphEKww6Kym9mMRLt4Wx4KwBm7bJekia2l
QKpYs1jvTYckbXUJFWAzj5Cr7Kl4sgc2jws93QyVlewJsx+a7U9oNUTfx/CK2L8y
neXBKXME9xPscCueRCr49XQWd4QKBgQDiNODLGl1V0zL6XATrKlfrVeun/WpM9ssW
5usJ1wLMqe2SG0hPa1L6p1IBBi1yEzwEs1uEuzn7jMhsQ+z2viGqbazDm8S3FwDm
RD2KUmKRICwT/vwTL6gn01he3blxn4j0pwnlaVZtm18tsNYLStZfLKWHPe0wnujt
walcwrUuWwKBgFvkZ8DURr0ecrZUQm6D2tRcyBJ0+kVe8gK0R8NAeZ5bz+YkP6fy
rxHX+C6HTTvlJ2MLQ+CzE9nvGnP8CsUnMuIeezhNMeH6mWv7222HTCGa+L7Z2pNm
398z/fxXaLcJNGkRFBkBS/rDk6LH8jZVEi6WgASKDU+XGse+SHwzX0RBAoGAfER2
Vb1nFvYR9nFE1ncpCqTJ8CCcXCmcCjnTVzWCOTME6yS5wp+AF2kB+MQPBUMhzqdF
mJZ0VS3TEP8D1809q2XS6ibRm46rnhfRAwMdWQKmlFmY22Vr6G0HeB4weQYeXwCO
xDPPi8TSIOEMTP6qBBb4v4ynYo+983FH8qGu08CgYBXA3m8VbrL4cWHqYlBoIiK
6yTDfYY3Z7QDuI+MmhWZNR6Ap2JmXwLntLPttN5REU7Xl38WW1xmNPOFmP6Mm939
qdGsFIROJkwFLqAV0ggr/19Lh7JTxsNkuWEJ8q2f4BN1oXjBvDhWRh3nRm6SBGG
7rm0he3G7dB27qqcc1nlqA=
-----END PRIVATE KEY-----
```


Nous allons désormais décrypter AliceDocumentEncrypted grâce à la clé Privée de Bob. Pour cela, on copie AliceDocumentEncrypted dans le dossier de Bob :

```
(root@kali)-[/home/kali/Documents/LAB1/Alice]
└─# cp /home/kali/Documents/LAB1/Alice/AliceDocumentEncrypted /home/kali/Documents/LAB1/Bob

(root@kali)-[/home/kali/Documents/LAB1/Alice]
└─# cd ..

(root@kali)-[/home/kali/Documents/LAB1]
└─# cd Bob

(root@kali)-[/home/kali/Documents/LAB1/Bob]
└─# ls
AliceDocumentEncrypted BobPrivateKey BobPublicKey

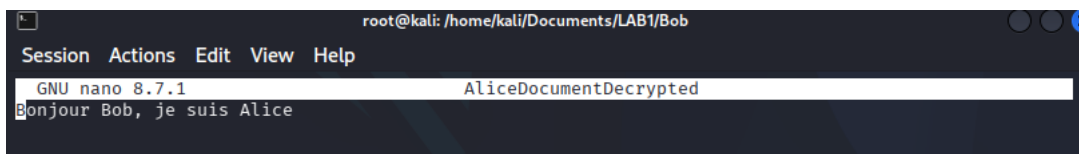
(root@kali)-[/home/kali/Documents/LAB1/Bob]
```

Désormais, nous allons décrypter AliceDocumentEncrypted avec BobPrivateKey :

```
(root@kali)-[/home/kali/Documents/LAB1/Bob]
└─# openssl rsautl -decrypt -in AliceDocumentEncrypted -inkey BobPrivateKey -out AliceDocumentDecrypted
The command rsautl was deprecated in version 3.0. Use 'pkeyutl' instead.

(root@kali)-[/home/kali/Documents/LAB1/Bob]
└─# ls
AliceDocumentDecrypted AliceDocumentEncrypted BobPrivateKey BobPublicKey
```

Vérifions le contenu du fichier décrypté :



```
root@kali: /home/kali/Documents/LAB1/Bob
Session Actions Edit View Help
GNU nano 8.7.1 AliceDocumentDecrypted
Bonjour Bob, je suis Alice
```

On retrouve bien le texte d'origine.

3-Chiffrement d'un fichier volumineux

Nous allons désormais chiffrer un fichier volumineux. On commence par créer le fichier LargeFile dans le dossier Alice :

```
(root@kali)-[~/home/kali/Documents/LAB1/Alice]
└─# openssl rand -out LargeFile -base64 $((2**30 * 3/4))
```

On encrypte ce fichier avec la clé publique de Bob :

```
(root@kali)-[~/home/kali/Documents/LAB1/Alice]
└─# openssl rand -out LargeFile -base64 $((2**30 * 3/4))

(root@kali)-[~/home/kali/Documents/LAB1/Alice]
└─# openssl rsautl -encrypt -in LargeFile -inkey BobPublicKey -out LargeFileEncrypted
The command rsautl was deprecated in version 3.0. Use 'pkeyutl' instead.
Could not find private key from BobPublicKey
40478EE48C7F0000:error:1608010C:STORE routines:ossl_store_handle_load_result:unsupported:../crypto/store/store_result.c:160:provider=default

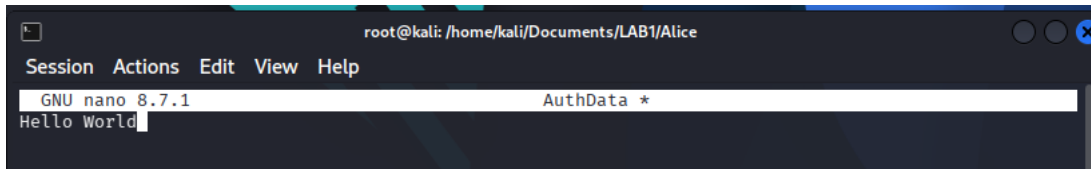
(root@kali)-[~/home/kali/Documents/LAB1/Alice]
└─# ls
AliceDocument  AliceDocumentEncrypted  AlicePrivateKey  AlicePublicKey  BobPublicKey  LargeFile
```

Nous obtenons un message d'erreur. Cela s'explique par le fait que le chiffrement asymétrique n'est pas possible sur les grands fichiers. Pour des fichiers volumineux, on préférera utiliser le chiffrement symétrique.

4-Signature électronique

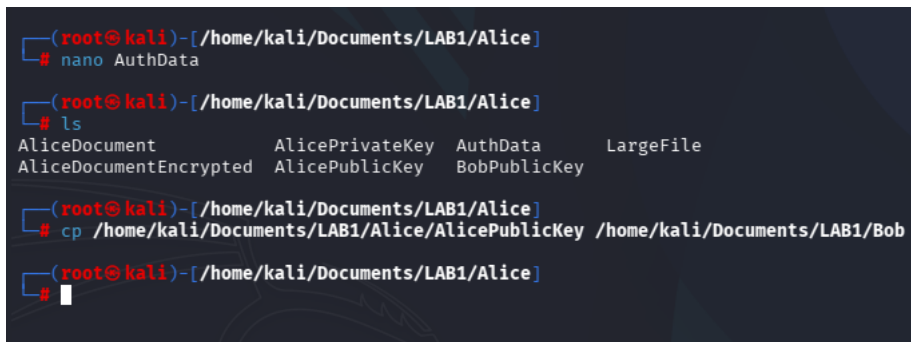
Nous allons désormais générer une signature électronique pour Alice. La signature permet à Alice de s'authentifier auprès de Bob et de s'assurer de l'intégrité des données.

On commence par créer un fichier texte AuthData dans le dossier Alice :



```
root@kali: /home/kali/Documents/LAB1/Alice
Session Actions Edit View Help
GNU nano 8.7.1 AuthData *
Hello World
```

On copie également la clé publique de Alice dans le dossier de Bob :



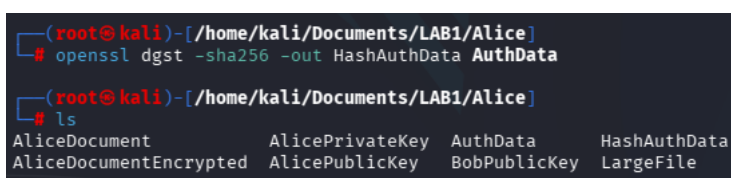
```
(root@kali)-[/home/kali/Documents/LAB1/Alice]
# nano AuthData

(root@kali)-[/home/kali/Documents/LAB1/Alice]
# ls
AliceDocument      AlicePrivateKey  AuthData          LargeFile
AliceDocumentEncrypted AlicePublicKey  BobPublicKey

(root@kali)-[/home/kali/Documents/LAB1/Alice]
# cp /home/kali/Documents/LAB1/Alice/AlicePublicKey /home/kali/Documents/LAB1/Bob

(root@kali)-[/home/kali/Documents/LAB1/Alice]
#
```

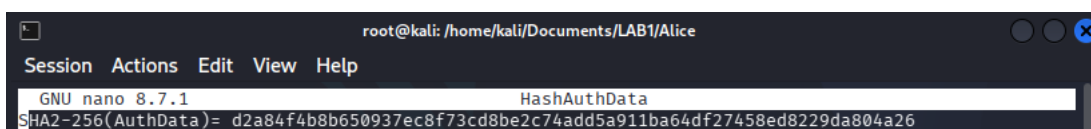
On applique désormais la fonction de hachage SHA256 sur le fichier AuthData pour générer son hachage HashAuthData :



```
(root@kali)-[/home/kali/Documents/LAB1/Alice]
# openssl dgst -sha256 -out HashAuthData AuthData

(root@kali)-[/home/kali/Documents/LAB1/Alice]
# ls
AliceDocument      AlicePrivateKey  AuthData          HashAuthData
AliceDocumentEncrypted AlicePublicKey  BobPublicKey      LargeFile
```

Vérifions le contenu de HashAuthData :



```
root@kali: /home/kali/Documents/LAB1/Alice
Session Actions Edit View Help
GNU nano 8.7.1 HashAuthData
SHA2-256(AuthData)= d2a84f4b8b650937ec8f73cd8be2c74add5a911ba64df27458ed8229da804a26
```

HashAuthData contient bien le hachage de AuthData

Nous allons maintenant procéder à la signature de HashAuthData avec la clé Privée de Alice en la nommant AliceSignature :

```
(root@kali)-[~/home/kali/Documents/LAB1/Alice]
└─# openssl rsautl -sign -in HashAuthData -inkey AlicePrivateKey -out AliceSignature
The command rsautl was deprecated in version 3.0. Use 'pkeyutl' instead.

(root@kali)-[~/home/kali/Documents/LAB1/Alice]
└─# ls
AliceDocument      AlicePrivateKey  AliceSignature  BobPublicKey  LargeFile
AliceDocumentEncrypted  AlicePublicKey  AuthData       HashAuthData
```

Vérifions le contenu de AliceSignature :

```
root@kali: /home/kali/Documents/LAB1/Alice
Session Actions Edit View Help
GNU nano 8.7.1 AliceSignature
8TtE*D|..c/'pF
u^C9HVw^$xcHeg@R^W^B^]T/^K^U^!^?eVD|^O^D^_y^W4h+i-XXL+u^'\T^F~M^L^m
r^;^U+k )^#^Cb%^^^^^B\Z^O^W^Ou^?
Hp^Ok^b^Bng^X^g^a^E^Z^J^\##^W0^
^("^O^e%^O^FKX^C+^\<^I^j^
^5^/;G
```

On copie ensuite AliceSignature et AuthData dans le dossier de Bob :

```
(root@kali)-[~/home/kali/Documents/LAB1/Alice]
└─# cp /home/kali/Documents/LAB1/Alice/AliceSignature /home/kali/Documents/LAB1/Bob

(root@kali)-[~/home/kali/Documents/LAB1/Alice]
└─# cp /home/kali/Documents/LAB1/Alice/AuthData /home/kali/Documents/LAB1/Bob

(root@kali)-[~/home/kali/Documents/LAB1/Bob]
└─# ls
AliceDocumentDecrypted  AlicePublicKey  AuthData      BobPublicKey
AliceDocumentEncrypted  AliceSignature  BobPrivateKey
```

Ensuite, nous allons vérifier AliceSignature avec la clé Publique de Alice.

Pour cela, nous allons récupérer le HashAuthData de AliceSignature :

```
(root@kali)-[~/home/kali/Documents/LAB1/Bob]
└─# openssl rsautl -verify -in AliceSignature -pubin -inkey AlicePublicKey -out HashAuthData
The command rsautl was deprecated in version 3.0. Use 'pkeyutl' instead.

(root@kali)-[~/home/kali/Documents/LAB1/Bob]
└─# ls
AliceDocumentDecrypted  AlicePublicKey  AuthData      BobPublicKey
AliceDocumentEncrypted  AliceSignature  BobPrivateKey  HashAuthData
```

On vérifie le contenu de HashAuthData :

```
root@kali: /home/kali/Documents/LAB1/Bob
Session Actions Edit View Help
GNU nano 8.7.1 HashAuthData
SHA2-256(AuthData)= d2a84f4b8b650937ec8f73cd8be2c74add5a911ba64df27458ed8229da804a26
```

Nous allons maintenant créer un nouveau hachage HashBob sur le fichier AuthData et le comparer avec HashAuthData pour vérifier que les données n'ont pas été modifiées.

Calculons HashBob sur AuthData :

```
(root@kali)-[~/home/kali/Documents/LAB1/Bob]
# openssl dgst -sha256 -out HashBob AuthData

(root@kali)-[~/home/kali/Documents/LAB1/Bob]
# ls
AliceDocumentDecrypted  AlicePublicKey  AuthData        BobPublicKey  HashBob
AliceDocumentEncrypted  AliceSignature  BobPrivateKey   HashAuthData
```

Vérifions le contenu de HashBob :

```
root@kali: /home/kali/Documents/LAB1/Bob
Session Actions Edit View Help
GNU nano 8.7.1 HashBob
SHA2-256(AuthData)= d2a84f4b8b650937ec8f73cd8be2c74add5a911ba64df27458ed8229da804a26
```

Comparons désormais HashBob et HashAuthData afin de vérifier que le fichier reçu par Bob n'a pas été modifié :

```
(root@kali)-[~/home/kali/Documents/LAB1/Bob]
# diff HashBob HashAuthData

(root@kali)-[~/home/kali/Documents/LAB1/Bob]
#
```

La commande ne renvoie aucun résultat, ce qui signifie que les deux hash sont identiques. Le fichier AuthData reçu par Bob est donc identique à celui de Alice

Si Modifions HashBob, la fonction « diff » devrait nous indiquer une différence entre les deux hash :

```
(root@kali)-[~/home/kali/Documents/LAB1/Bob]
# diff HashBob HashAuthData

1c1
< SHA2-256(AuthData)= d2a84f4b8b650937ec8f73cd8be2c74add5a911ba64df27458ed8229da804a2
> SHA2-256(AuthData)= d2a84f4b8b650937ec8f73cd8be2c74add5a911ba64df27458ed8229da804a26
```

Nous voyons ici que les deux hash sont différents.

Dans cette situation, si les deux hash sont différents, on peut supposer que le fichier a été modifié lors de l'envoi par un acteur malveillant.