

TP Sécurité - Manipulation des grands principes de la cryptographie

3 Les consignes

3.1 Utilisation d'un algorithme de hachage, le SHA1

3.1.1 Rappels théoriques

Une fonction de hachage (One copy hash function) est une fonction de prise d'empreinte numérique. Il n'y a pas de "clef" au sens propre et l'algorithme est connu de tous.

Le principe consiste à réaliser un condensat (appelé "empreinte" ou "hash") de longueur fixe et représentatif de la donnée d'entrée d'une taille variable. Il n'est cependant pas possible de retrouver le plaintext à partir du hash.

Néanmoins, il est possible de casser cette protection sans connaître le mot de passe réel via linux (/etc/shadow) en utilisant une attaque par force brute et/ou une attaque par dictionnaire.

Afin de renforcer l'entropie du hachage, on ajoute un aléas appelé salage (salting).

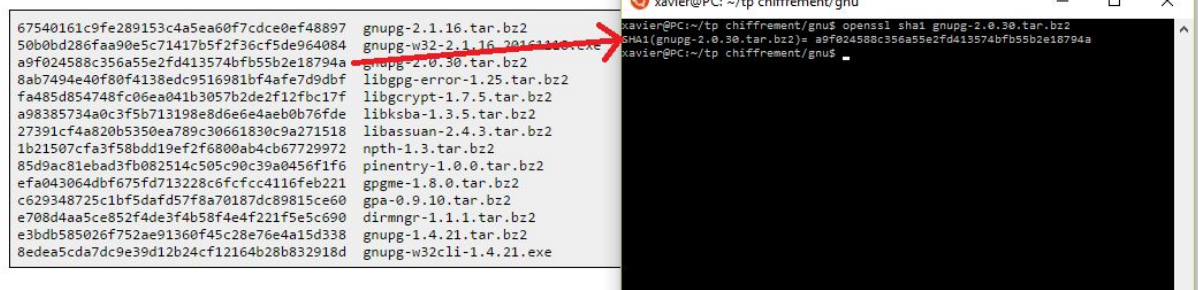
Dans le cas du téléchargement d'un fichier sur un site internet, la fonction de hachage permet de prévenir l'intégrité du fichier, confirmant ainsi si le fichier est authentique ou non.

3.1.2 Contrôle d'intégrité d'un téléchargement

Condensats uniques des archives GNUPG générés par SHA1 (Xavier)

LIST OF SHA-1 CHECK-SUMS

For your convenience, all SHA-1 check-sums available for software that can be downloaded from [our site](#), have been gathered below.



Condensats uniques des archives GNUPG générés par SHA1 (Charles)

LIST OF SHA-1 CHECK-SUMS

For your convenience, all SHA-1 check-sums available have been gathered below.

67540161c9fe289153c4a5ea60f7cdce0ef48897 50b0bd286faa90e5c71417b5f2f36cf5de964084 a9f024588c356a55e2fd413574bfb55b2e18794a 8ab7494e40f80f4138edc9516981bf4afe7d9dbf fa485d854748fc06ea041b3057b2de2f12fbc17f a98385734a0c3f5b713198e8d6e6e4aeb0b76fde 27391cf4a820b5350ea789c30661830c9a271518 1b21507cfa3f58bdd19ef2f6800ab4cb67729972 85d9ac81ebad3fb082514c505c90c39a0456f1f6 efa043064dbf675fd713228c6fcfcc4116feb221 c629348725c1bf5dafd57f8a70187dc89815ce60 e708d4aa5ce852f4de3f4b58f4e4f221f5e5c690 e3bdb585026f752ae91360f45c28e76e4a15d338 8edea5cda7dc9e39d12b24cf12164b28b832918d	gnupg-2.1.18 gnupg-w32c-2.1.18 gnupg-2.0.30.tar.bz2 libgpg-error-1.18 libgcrypt-1.8.5 libksba-1.3.5 libassuan-2.0.5 nptl-1.3.3 pinentry-1.1.0 gpgme-1.8.0 gpa-0.9.10 dirnmgr-1.1.0 gnupg-1.4.23 gnupg-w32c-1.4.23
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Afin de vérifier que le fichier n'a pas été altéré pendant le téléchargement, nous avons comparés les condensats uniques du fichier générés avec SHA1 (versions officielle et manuelle). Avec l'image ci-dessus, nous pouvons confirmer que l'archive GNUPG n'a pas été altéré pendant le téléchargement (a9f024588c356a55e2fd413574bfb55b2e18794a). L'intégrité du fichier a bien été contrôlée.

3.2 Chiffrement de fichier à l'aide du chiffrement symétrique AES

3.2.1 Rappels théoriques

Le principe du chiffrement symétrique consiste à utiliser une clef partagée pour le chiffrement et le déchiffrement (même clef utilisée, les mécanismes étant identiques).

Il existe deux types de chiffrement symétrique :

- Chiffrement par flux (stream cipher)
 - Le message est chiffré bit par bit (ou octet par octet),
 - Un XOR est effectué entre la clef et le message clair,
 - RC4
- Chiffrement par blocs (bloc cipher)
 - On découpe le message clair en blocs de taille fixe. La taille des blocs est définie par l'algorithme de chiffrement utilisé (Exemple : DES/64 bits, AES/128 bits),
 - On chiffre chacun des blocs
 - La manière dont les blocs vont être manipulés, chiffrés et le message reconstitué s'appelle le mode d'opération.

Les applications du chiffrement symétrique peuvent fournir les services de confidentialité et d'authentification.

Cependant le chiffrement symétrique contient deux grandes faiblesses :

- L'échange de la clef est la principale vulnérabilité ;

- Le nombre de clefs augmente avec le nombre d'échanges différents, c'est donc difficile à gérer.

3.2.2 Chiffrement de fichier

Chiffrement du fichier textCrypted.txt (Xavier)

```
xavier@PC: ~/tp chiffrement/gnu
-camellia-128-cfb1      -camellia-128-cfb8      -camellia-128-ecb
-camellia-128-ofb      -camellia-192-cbc       -camellia-192-cfb
-camellia-192-cfb1     -camellia-192-cfb8      -camellia-192-ecb
-camellia-192-ofb      -camellia-256-cbc       -camellia-256-cfb
-camellia-256-cfb1     -camellia-256-cfb8      -camellia-256-ecb
-camellia-256-ofb      -camellia128            -camellia192
-camellia256          -cast                    -cast-cbc
-cast5-cbc             -cast5-cfb              -cast5-ecb
-cast5-ofb             -des                     -des-cbc
-des-cfb               -des-cfb1               -des-cfb8
-des-ecb               -des-edc                -des-edc-cbc
-des-edc-cfb          -des-edc-ofb            -des-edc3
-des-edc3-cbc          -des-edc3-cfb           -des-edc3-cfb1
-des-edc3-cfb8         -des-edc3-ofb           -des-ofb
-des3                  -desx                    -desx-cbc
-id-aes128-GCM          -id-aes192-GCM          -id-aes256-GCM
-rc2                   -rc2-40-cbc             -rc2-64-cbc
-rc2-cbc               -rc2-cfb                -rc2-ecb
-rc2-ofb               -rc4                     -rc4-40
-rc4-hmac-md5          -seed                    -seed-cbc
-seed-cfb              -seed-ecb               -seed-ofb

xavier@PC:~/tp chiffrement/gnu$ openssl aes-256-cbc -in text -out textCrypted
enter aes-256-cbc encryption password:epsi
Verifying - enter aes-256-cbc encryption password:epsi
```

Déchiffrement du fichier textCrypted.txt (Alexis)

	Nom	Modifié le	Type	Taille
récents	epsi_2016-2017_i5_SECU-INFO_TP_1.pdf	15/12/2016 14:31	Adobe Acrobat D...	172 Ko
	textCrypted.txt	15/12/2016 14:50	Document texte	1 Ko
	textDecrypted.txt	15/12/2016 14:53	Document texte	1 Ko

ts

textCrypted.txt - Bloc-notes

Fichier Edition Format Affichage ?

Salted__ Ð>“W*«ã-ŒLAáÔŦ ;iðěžsü_]μ→ž~ö %ÚVÿf•6J f,,^êU«5èpø*x>+

C:\Windows\system32\cmd.exe

```
D:\I5\Sécurité de l'information et az - Thierry Meyer\TP>openssl enc -d -aes-256-cbc -in textCrypted.txt -out textDecrypted.txt
enter aes-256-cbc decryption password:

D:\I5\Sécurité de l'information et az - Thierry Meyer\TP>
```


textDecrypted.txt - Bloc-notes

Fichier Edition Format Affichage ?

Ce texte est normalement chiffré !

Déchiffrement du fichier textCrypted.txt (Charles)

```
charlit@charlit-VirtualBox: ~/Téléchargements
charlit@charlit-VirtualBox:~/Téléchargements$ openssl aes-256-cbc -d -in textCrypted.txt -out test.txt
enter aes-256-cbc decryption password:
test.txt (~/.Téléchargements) - gedit
```

Ouvrir

Ce texte est normalement chiffré !

Téléchargements

Dossier personnel Téléchargements

Récents Dossier personnel Bureau Documents Images Musique Téléchargements Vidéos

textcrypteeed.txt xavier.pub testpublic.pub test.key

test.txt textCrypted.txt

3.3 Manipulation des protocoles de chiffrement asymétrique

3.3.1 Rappels théoriques

Le principe de la cryptographie asymétrique (ou à clef publique) est que chaque participant à l'échange dispose d'une biclef (paire de clefs, indissociables) :

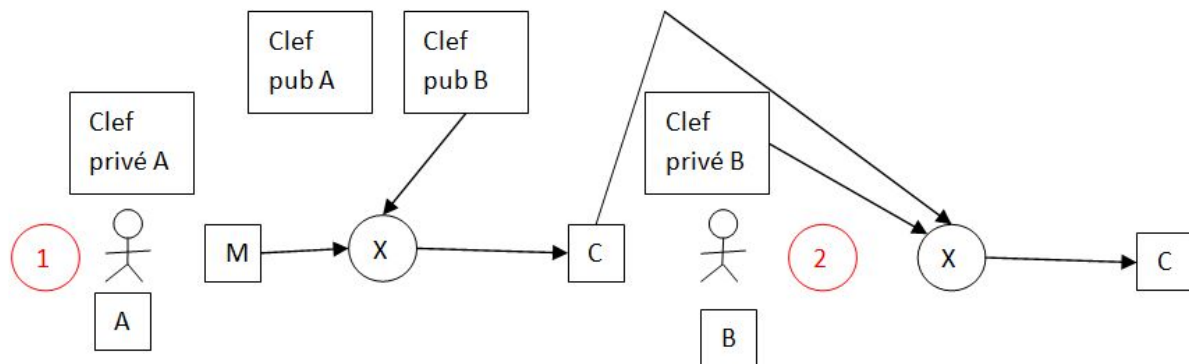
- une clef privée, qui sert à déchiffrer : il conserve sa clef privée en sécurité,
- une clef publique, qui sert à chiffrer : celle-ci est distribuée avec les participants à l'échange.

Il est important de protéger sa clef privée, en la chiffrant avec une passphrase et/ou n'affecter des droits de lecture qu'au propriétaire du fichier contenant la clef privée (strict mode sur un serveur ssh).

L'algorithme utilisé pour chiffrer et déchiffrer est le même.

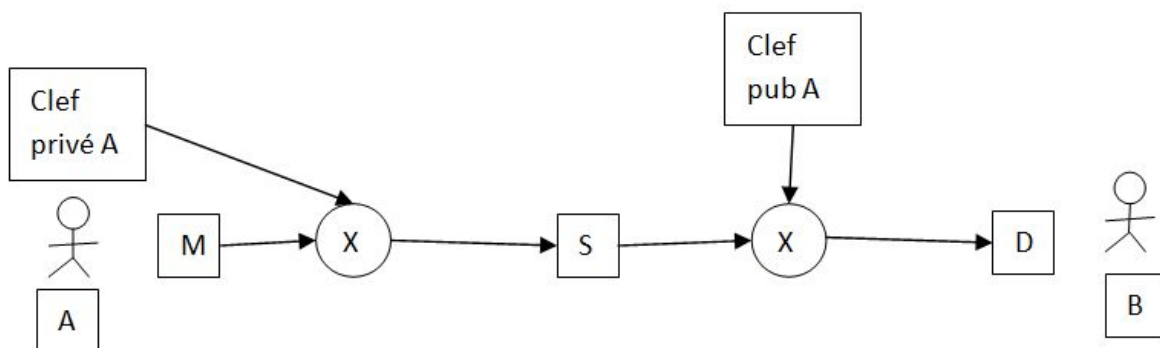
Les deux applications principales de la cryptographie asymétrique sont :

- Le chiffrement :



Ici, A chiffre son message avec la clef publique de B et l'envoi à B. B reçoit un message crypté et le déchiffre avec sa clef privée (B). B peut à présent lire le contenu du message de A.

- La signature électronique :



Ici, A signe son message avec sa clef privée (A) et l'envoi à B. B utilise la clef publique de A pour vérifier l'authenticité et l'intégrité du message reçu de A. Si la signature du message reçu et du message vérifié son identique, alors le message provient bien de A.

3.3.2 Préparation de la bi clef

Génération clef privée et clef publique (Alexis)

	Nom	Modifié le	Type	Taille
	3.2.2.png	15/12/2016 14:55	Image PNG	30 Ko
its récents	epsi_2016-2017_i5_SECU-INFO_TP_1.pdf	15/12/2016 14:31	Adobe Acrobat D...	172 Ko
	rsa.key	15/12/2016 14:57	Fichier KEY	1 Ko
ients	rsa.pub	15/12/2016 15:00	Fichier PUB	1 Ko
	textCrypted.txt	15/12/2016 14:50	Document texte	1 Ko
	textDecrypted.txt	15/12/2016 14:53	Document texte	1 Ko

```

C:\Windows\system32\cmd.exe

D:\I5\Sécurité de l'information et az - Thierry Meyer\IP>openssl genrsa -out rsa
.key
Loading 'screen' into random state - done
Generating RSA private key, 512 bit long modulus
...+++++
...+++++
e is 65537 (0x10001)

D:\I5\Sécurité de l'information et az - Thierry Meyer\IP>ls -l rsa.key
ls: l: No such file or directory
rsa.key

D:\I5\Sécurité de l'information et az - Thierry Meyer\IP>ls -l rsa.key
-rw-r--r--  1 Mikuraya Administ  493 Dec 15 13:57 rsa.key

D:\I5\Sécurité de l'information et az - Thierry Meyer\IP>cat rsa.key
-----BEGIN RSA PRIVATE KEY-----
MIIBOgIBAAJBAMHJz0UGcUdXO6MJMw3LhLT0e0JGmaPQR8u0t2gIrA0FWosbljn1
h7Fk20ZH0zp+9PH8sQsfEZYKhjCPHJpiyncCAwEAAQJAK10q4aUz7AUW9XpSWmc+
eKylLJpQhh5HQ7scKTMUIbgyszsUUIRjpXLSUXN/FAQHJNczUGcAF03sCU60U00
4QIhAP10xU3rUIm0nBmti/D9Ielzfazjaltmh9GxHysgM7unaieAw7u6fLZyYtJN
EKkuQjOWateQF/h54U/qd52/0j8DFLECIQDyNLI307xHABUaLlhg/0U89iTPJrPo
jF6WgJCsgaUuWQIgSSdqpZ3S1phe8S68DkYUm80BL9RQSS1+CkLMNJkkX6ECIHeh
7Tdard2s0mXgua8XKU+yfkYb9DvQJ1o0iy8cA0lq
-----END RSA PRIVATE KEY-----

D:\I5\Sécurité de l'information et az - Thierry Meyer\IP>openssl rsa -in rsa.key
-pubout > rsa.pub
writing RSA key

D:\I5\Sécurité de l'information et az - Thierry Meyer\IP>

```

Génération clef privée et clef publique (Charles)

```

test.key - LibreOffice Writer

-----BEGIN RSA PRIVATE KEY-----
MIIEpAIBAAKCAQEAZyaaJZ2MK8pCQ7T2vJELAGn661pJiopci1aHsYlQ7ULKMDgo2
atvesug355CBKnmZhw13qYQU1mrTk5Puuh1H8UvGkW1jtNrQrestz0dykgWmG11ap
9/E0bjpBoYKh8VmdpyNFIDaKdQa1cMJPgEnVI1/3gYMYs2a+M0kjBCN8uu0yaM
HJp3LRQpW0m7kza2Qs0P510X6p6NxV80JknPQJvpZyYazjZ7m50CqP1usRwL16w
UmbUBdWOUHn0Lrdj33PKKEybfxtsxwXBE/tXf41eB1zWvsdLMD1wJ5U/NP9mHC
16SFLyFmJwXKtNXSuxs3f2GoGRbRn0h6h0iWIDAQABAQCO7w/JcUogEakH
Zah80QtXrFzuQ26H1RZFFI/1QctHogS5Nu0FgAfVIR0LV1FjZJrQxd/henzv954
SXjkm/opJgq1zuSEdczPQ89Y1+eCM+q2RTX9HN+wEnJ/woJ5T2yha0H8/tGPX4
/VL62uokPzbjyJe6yHs9qPPEv38t4WBrDpLwEkyIs0XoMFID39LjKtQm1xv31rZ6
BxgnPQAb+1s55x889E0OxavEDB0zsSSb0c9wWdHkFDVqF1ToxClZf4SmFjP//
x/EotQEYp206CLsVkv0w3081xQtetIXS1v1k4ZCZ0J0M0819K36IkwN2QzN6
Sz84wxtBA0G8A0ZIMVS2tYjGha/f8WjmkC3IR8vvKkt4N+Lr413WCMk0M1e/gzoa
gpcS8/MetSs+jf4FM1M87eKJKPvukUQtFegbeFQPLNU/RGo5Bb0qhPsvPddR-D1
bc1F2P5eEH1Fc12xt6E1ydzRbKJ83n101U3Yugc5YpJtQKPYVfEw7RAoGBAN4Z
x16K21K+0p1xAnd8a1MF4+ScMurDXT4pf1gMiskzFD1yuRS
ye0XNFjKbcD0/q1gbwtBHvJBAQ1eEE6YNRzbX+amp0wIITm
nW10mM10Rer1j0Mdh1hT17ASavY/Y4URaTAKnabAoGAdew
oN6U1rh79ktmjfbK9iyU03RrYpwhYRHs1F09JgLK6b7wa
oR0H0Ahd/b817nAHVYGEJmoqJ1Bzz/chWamfBafVt1x1+/24
wEKFccHA4dukP2LkxFrnDECgYBE0d6rRXSQcQ0Q0SMH/Gn
LBG9Nc0cNzrQXLMcr/shNMGS11fYgny8+vcqrJ0XCHX2640
UAZxsBdRe16fGd/Htgw7B91T5/qDL2FJUQqWn+WSe80/sf05
JK3mqQK8QDfRjxxYrYTR8y1zh57VYw+DX1SFAI3TSrqr
Hy4Q7yry8B8koyUmrJgXvGrDTM6NT0q1Aez+Rq06Sct4TFYSy7LtppxSuyJRWpZg
8DWGHRFS215m/8TnnY2T/ewKgr416UmerPwEQR3Z051aVwoeAwH2Q==
-----END RSA PRIVATE KEY-----

charlit@charlit-VirtualBox: ~
charlit@charlit-VirtualBox:~$ openssl genrsa -out test.key
Generating RSA private key, 2048 bit long modulus
.....+++++
.....+++++
e is 65537 (0x10001)
charlit@charlit-VirtualBox:~$

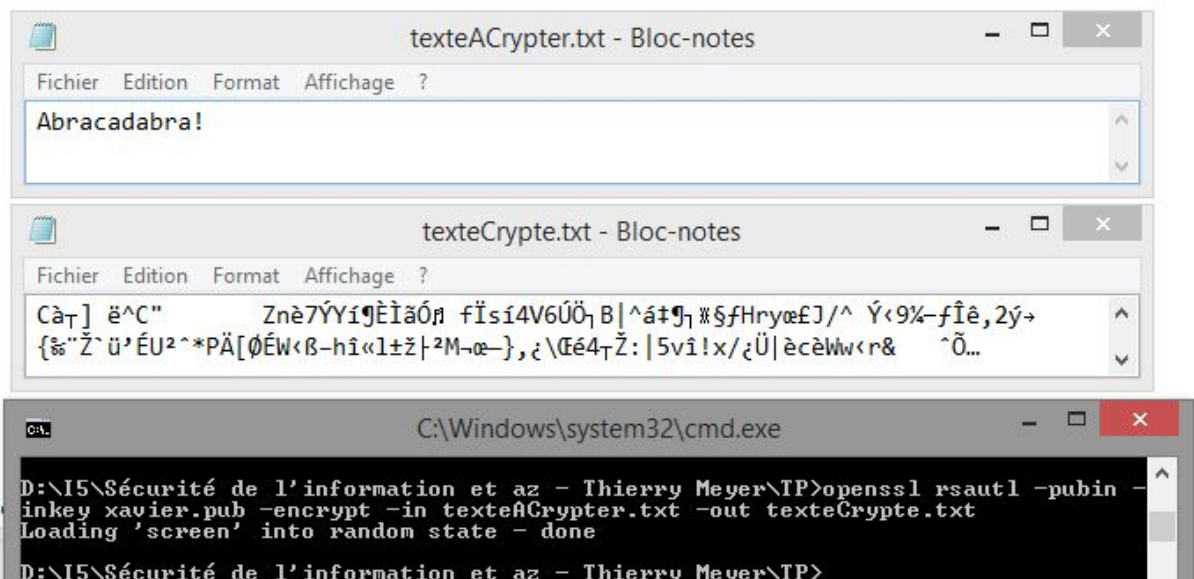
```

Bien évidemment, la clef a été partagée aux interlocuteurs et la clef publique.

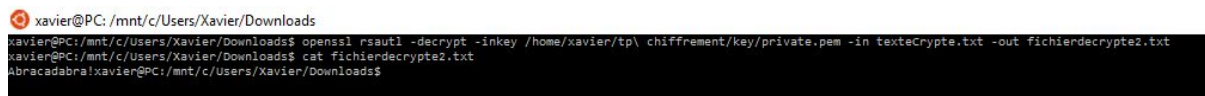
3.3.3 Chiffrement et déchiffrement de fichiers

Cryptage par clef publique (Alexis)

Nom	Modifié le	Type	Taille
3.2.2.png	15/12/2016 14:55	Image PNG	30 Ko
3.3.2.PNG	15/12/2016 15:01	Image PNG	43 Ko
epsi_2016-2017_i5_SECU-INFO_TP_1.pdf	15/12/2016 14:31	Adobe Acrobat D...	172 Ko
rsa.key	15/12/2016 14:57	Fichier KEY	1 Ko
rsa.pub	15/12/2016 15:00	Fichier PUB	1 Ko
textCrypted.txt	15/12/2016 14:50	Document texte	1 Ko
textDecrypted.txt	15/12/2016 14:53	Document texte	1 Ko
texteACrypter.txt	15/12/2016 15:09	Document texte	1 Ko
texteCrypte.txt	15/12/2016 15:09	Document texte	1 Ko
xavier.pub	15/12/2016 15:03	Fichier PUB	1 Ko



Décryptage par clé privée (Xavier)



Le contenu du fichier décrypté est bien conforme au texteACrypter.txt = Abracadabra!


3.3.4 Signature électronique

Signature (Xavier)

```
xavier@PC: ~/tp chiffrement/gnu
xavier@PC:~/tp chiffrement/gnu$ openssl dgst -sha512 -sign ../key/private.pem -out text.sha1 text
xavier@PC:~/tp chiffrement/gnu$ ll
total 4316
drwxrwxrwx 2 xavier xavier  0 janv.  2 10:07 ./
drwxrwxrwx 2 xavier xavier  0 déc. 15 14:54 ../
-rw-rw-rw- 1 xavier xavier 4414652 déc. 15 14:38 gnupg-2.0.30.tar.bz2
-rw-rw-rw- 1 xavier xavier 287 déc. 15 14:38 gnupg-2.0.30.tar.bz2.sig
-rw-rw-rw- 1 xavier xavier 36 déc. 15 14:44 text
-rw-rw-rw- 1 xavier xavier  0 janv.  2 09:59 textCrypted
-rw-rw-rw- 1 xavier xavier 256 janv.  2 10:05 textCrypted.sha1
-rw-rw-rw- 1 xavier xavier 256 janv.  2 10:07 text.sha1
xavier@PC:~/tp chiffrement/gnu$
```

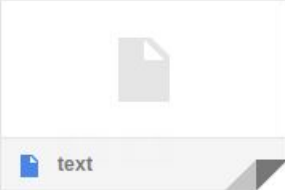

- Création d'une empreinte sha1
- Visualisation du contenu
- Crypter l'empreinte du fichier avec une clé privée (sign) = Signature

Vérification de la signature (Charles)


Xavier Laffargue
10:08 (Il y a 9 minutes)

À moi

2 pièces jointes

```
charlit@charlit-VirtualBox: ~/Téléchargements
charlit@charlit-VirtualBox:~$ cd Téléchargements/
charlit@charlit-VirtualBox:~/Téléchargements$ openssl dgst -sha512 -verify xavier.pub -signature text.sha1 text
Verified OK
charlit@charlit-VirtualBox:~/Téléchargements$
```

- Décrypter la signature avec une clé publique (verify)
- Comparer les deux empreintes pour vérifications

Les deux empreintes sont identiques, le fichier est donc bien authentique !