KRIJAH Iptissame

Projet : Chiffrement de fichiers

1.

Il est recommandé d'utiliser des blocs de 128 bits pour éviter les attaques comme la constitution de dictionnaires (cf RègleBlocSym-2. « Pour une utilisation au-delà de 2020, la taille minimale des blocs de mécanismes de chiffrement par bloc est de 128 bits ».). Il convient donc d'utiliser l'AES 128 bits.

Aussi, il est recommandé d'utiliser des clés de 128 bits. (cf RègleCléSym-2. « La taille minimale des clés symétriques devant être utilisées au-delà de 2020 est de 128 bits. ») L'emploi de clés de 128 bits permet de s'assurer que les attaques génériques par recherche exhaustive seront inopérantes car les clés de 56 bits sont clairement insuffisantes et la capacité actuelle à attaquer des clés de 64 bits est même aujourd'hui admise.

Le mode CBC est un mode sûr puisqu'il garantit la confidentialité des informations avec l'emploi de valeurs initiales (le chiffrement du même message deux fois de suite n'a qu'une chance infime d'utiliser la même valeur initiale). Ainsi, le message chiffré sera toujours différent. Le padding PKCS5 permet de compléter le message par un bit valant 1 suivi de bits nuls lors de la phase initiale pour permettre les opérations de chiffrement sur les blocs.

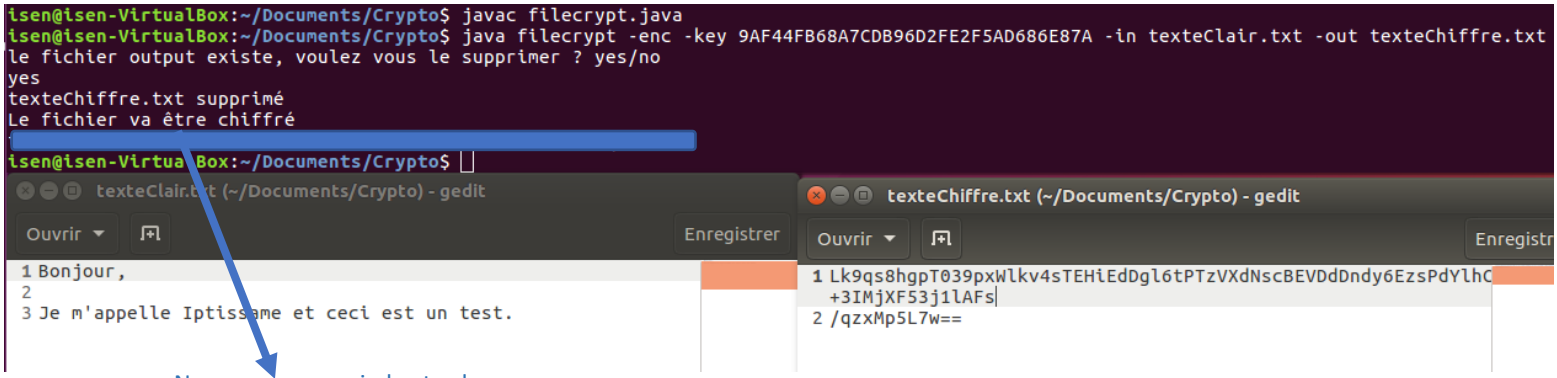
Pour toutes ces raisons, j'utiliserai le mode de chiffrement **AES 128bits / CBC / PKCS5.**

2.

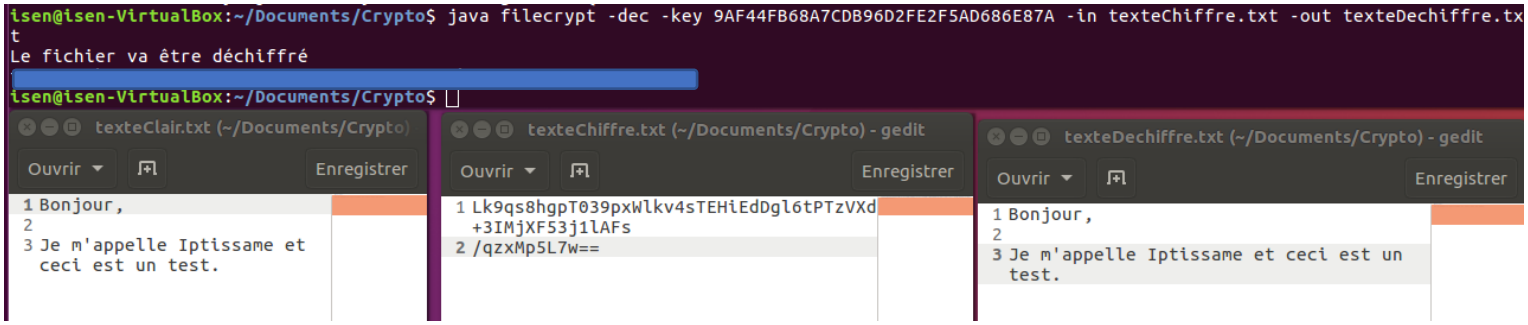
Pour l'utilisation de l'utilitaire, faites :

$ javac filecrypt.java

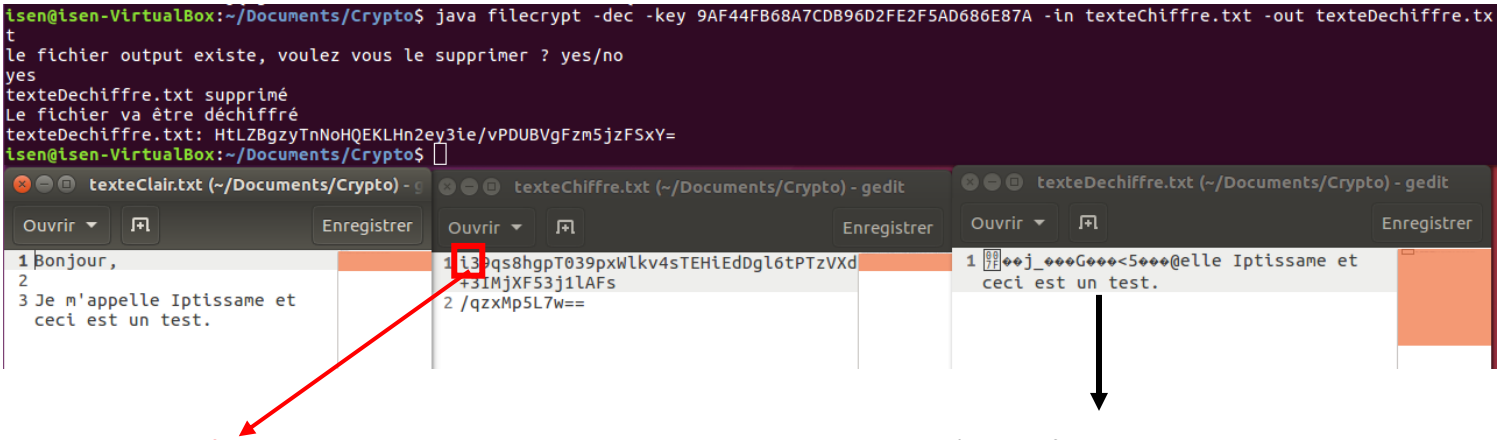
$ java filecrypt -enc -key FF……FF -in <votrefichierinput.txt> -out <votrefichieroutput.txt>



Nous verrons ceci plus tard.



Nous pouvons voir que le fichier a été déchiffré correctement. Mais si je modifie le fichier chiffré, j'obtiens ce résultat :



Modification

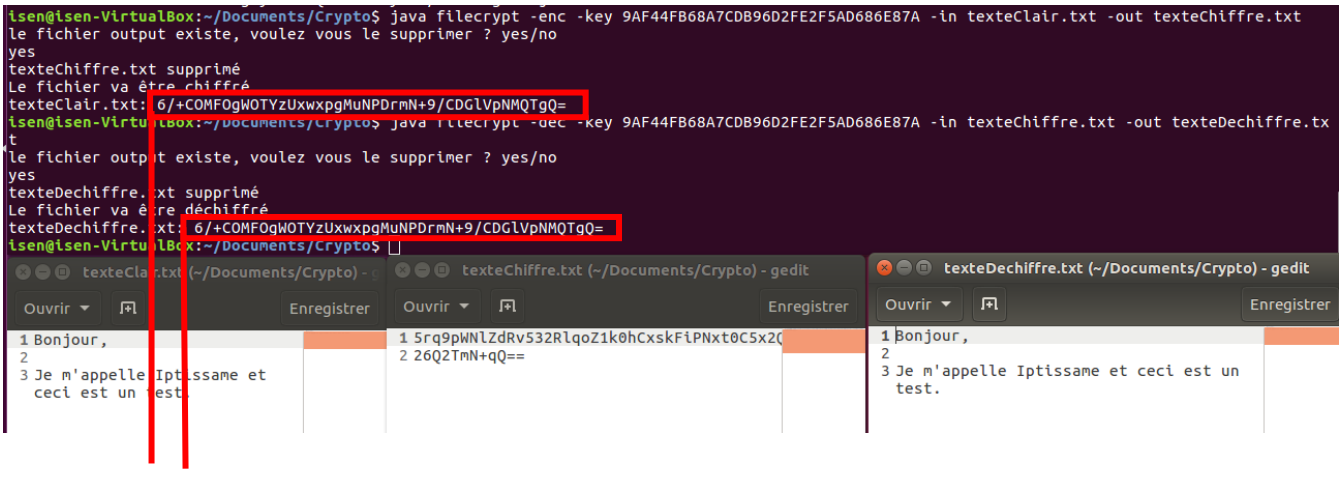
Le début du fichier est illisible.

Si le début du fichier est modifié, alors les blocs premiers blocs déchiffrés seront illisibles. Cependant, si deux blocs de suite restent inchangés, alors la suite du fichier peut-être normalement déchiffré. Le mode d'opération CBC a une propriété d'auto-synchronisation.

3.

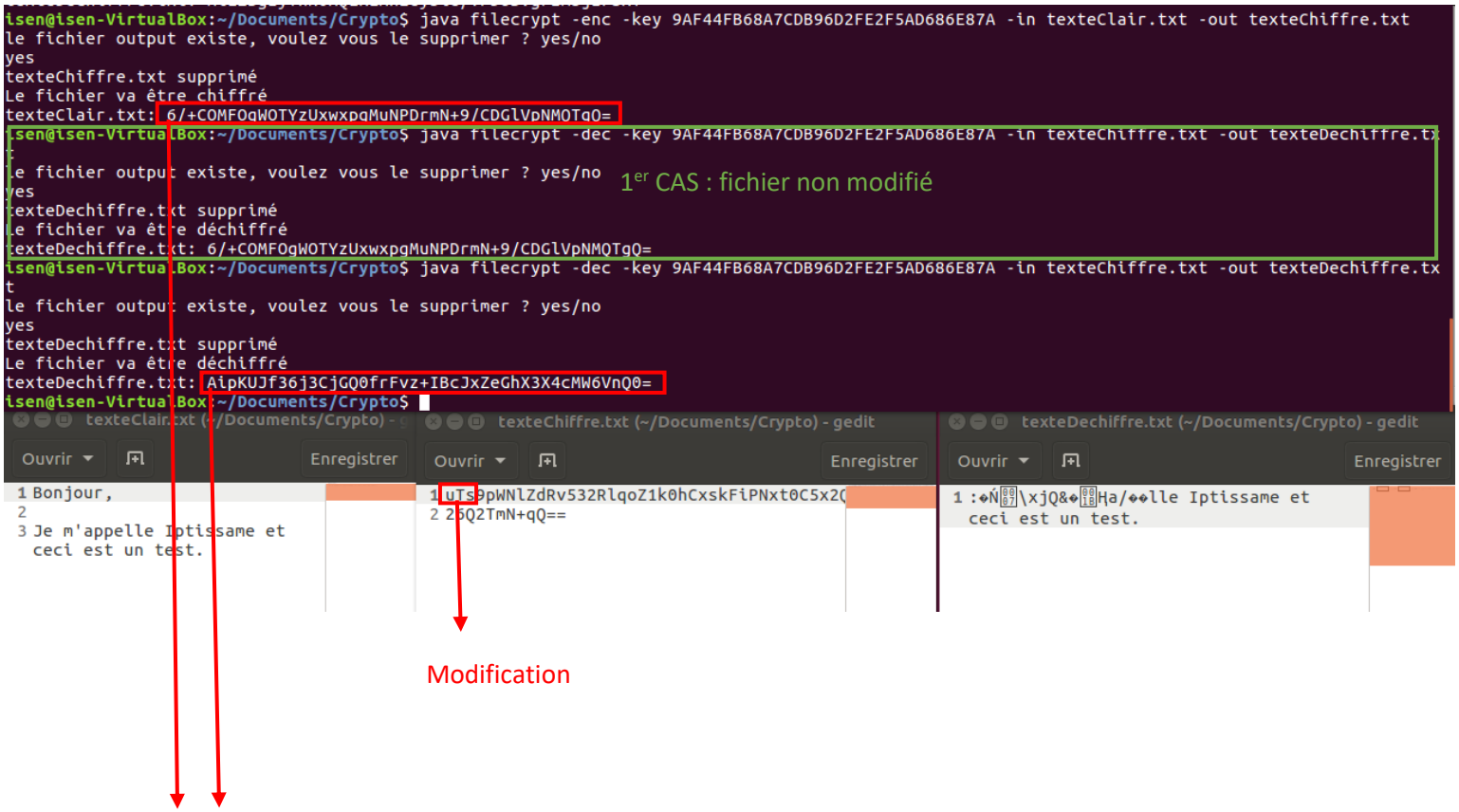
Un code d'authentification de message ou MAC permet de garantir qu'un message (un byte array) n'a pas été modifié pendant son transit. L'utilisation d'un MAC pour garantir la sécurité de la transmission des messages exige que les deux parties partagent une clé secrète pour pouvoir générer et vérifier le MAC.

- 1er Cas : cas où le fichier n'a pas été modifié pendant son transit



Nous pouvons voir qu'il y a exactement le même MAC. Donc le fichier n'a pas été modifié.

- 2nd Cas : cas où le fichier a été modifié pendant son transit



Les MAC sont complètement différents. Il y a donc eu une modification.

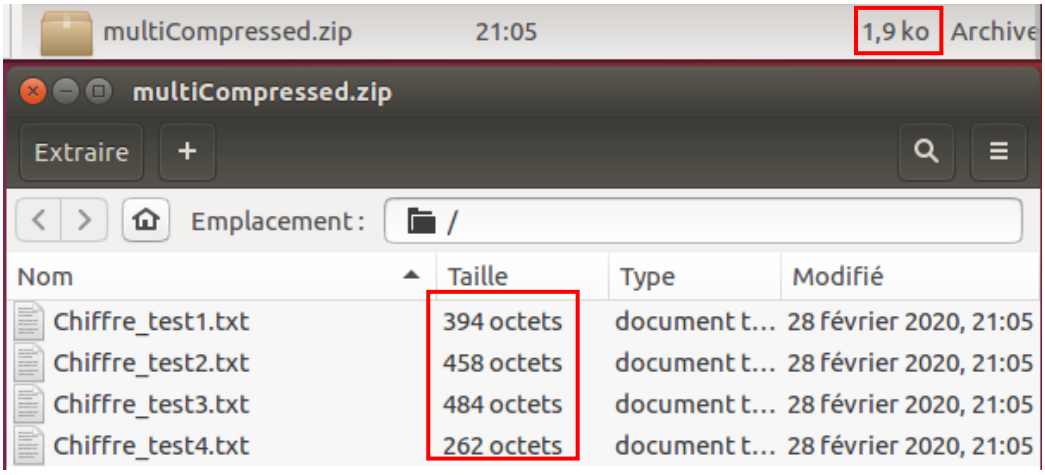
4.

<u>Remarques préliminaires :</u>

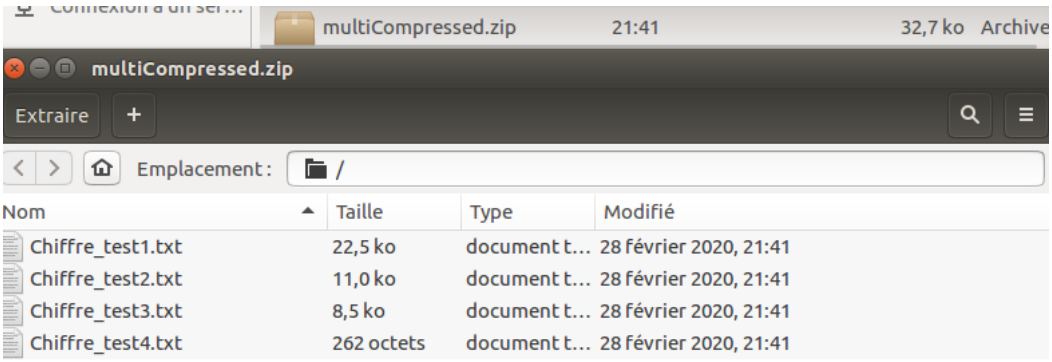
- Afin de répondre à la question, je n'ai fait que chiffrer les fichiers et compresser les fichiers chiffrés (il n'y a pas de déchiffrement possible).
- J'ai fait un nouvel utilitaire qui ne prend en compte que des inputs à chiffrer (pas d'output à expliciter) : $ java **filecryptV2** -enc -key FFF….FF -in <fichier1> <fichier2> …. <fichierN>
- Il ne faut pas chiffrer deux fois un même fichier **en même temps** car il risque d'y avoir des erreurs (le fichier chiffré ainsi que l'IV et le MAC pourraient être remplacés car il y a les mêmes nom de fichiers). '' $ java filencryptV2 -enc -key FFF….FF -in **fichier1.txt fichier1.txt** fichier2.txt'' est donc à proscrire.

L'utilitaire précédent (ainsi que celui de la question 2. à la question 3.) permet de chiffrer différemment un fichier au contenu identique : un IV aléatoire différent est généré à chaque chiffrement et la probabilité que l'on génère un même IV est infime. Cela permet donc d'avoir des fichiers chiffrés totalement différents.

5.



1er cas : test avec des fichiers peu volumineux. La somme des fichiers vaut 1,6 ko octets alors que le fichier zip vaut 1,9 ko. Je suppose que l'écart est dû aux informations pour la décompression. La compression est donc inutile pour les fichiers trop petits.



2nd cas : test avec des fichiers très volumineux. La somme des fichiers vaut 42,2 ko alors que le fichier zip vaut 32,7 ko. La compression est utile ici.

6. …

Krijah Iptissame – 28/02/2020