TP4 – Gestion des clés et certificats

Dans la première partie de ce TP nous allos découvrir la commande `keytool` qui permet de gérer un "magasin de clés" (angl. *keystore*) dans lequel on pourra garder nos clés

Exercice 1:

1. Créez un nouveau keystore et insérez une nouvelle clé à l'aide de la commande

   `keytool -v -genkey -alias nvllecle -keystore nvmagasin`

   Comme vous allez voir, `keytool` va vous demander un certain nombre d'informations à mettre dans le nouveau certificat qu'elle va créer pour la nouvelle clé, et, en particulier, le mot de passe protégeant la clé et celui protégeant le keystore lui même. L'algorithme par défaut est le "DSA" (Digital Signature Algorithm), et la taille de la clé privée est de 1024 bits. Enfin, l'alias de votre nouvelle clé est `nvllecle` (par défaut elle s'appelera `mykey`).

   Important : le certificat de votre nouvelle clé est *auto-signé !*
2. Affichez le contenu de votre keystore à l'aide de la commande

   `keytool -v -list -keystore nvmagasin -v`
3. Créez un fichier avec le certificat (auto-certifié !) de votre clé avec

   `keytool -v -export -alias nvllecle -keystore nvmagasin -file moncert`
4. Affichez le contenu du certificat avec

   `keytool -v -printcert -file moncertificat`
5. Échagez les certificats avec vos collègues, ensuite importez les noveaux certificats dans votre keystore :

   `keytool -v -import -file moncert -keystore autremag`

Important : une fois que vous avez inseré le certificat dans votre keystore, le certificat est considéré *de confiance !* (D'ailleurs, le `keytool` vous le demande explicitement lors de l'insertion du nouveau certificat).

Un fichier du nom de `cacerts` existe dans le répertoire `java.home/lib/security`, et il représente un keystore utilisable dans tout le système ; il est gérable par le `root` à l'aide de `keytool`.

Exercice 2: Écrire un programme `signature.java` qui lit une entrée dans le keystore (donc une paire clé publique/clé privée) pour générer une signature, et écrit la clé publique utilisée dans un fichier, utilisant le format de codage "X.509".

Écrire ensuite un deuxième programme `versig.java` qui vérifie la signature digitale fournie par `signature.java`, en utilisant la clé publique créée par celui-ci (et récuperée dans le fichier `clepublique`).
