Université de la Polynésie Française
Licence Sciences & Technologies
L3/S5 - option Informatique
Module Sécurité Informatique

Contrôle "Sécurité Informatique" (pratique)

jeudi 17 novembre 2011
durée : 3h
responsable : M.Munier

2 page(s)

Documents autorisés : Pour cette évaluation de travaux pratiques vous êtes autorisés à utiliser tous les documents pédagogiques qui vous ont été fournis : supports de cours "Sécurité Informatique" (éventuellement annotés) distribués par M.Munier, corrections des TD/TP, notes de TD et notes de TP, documentations mises à disposition. Tout autre document (documentations récupérées sur le web, photocopies de livres, etc...) est formellement interdit. Dans le même ordre d'idée, aucune assistance électronique (calculatrice, ordinateur de poche, téléphone portable, PDA,...) n'est autorisée. Toutes les notions vues en cours sont supposées acquises et, le cas échéant, doivent être mises en œuvre.

⋆ ⋆ ⋆

Les différents fichiers nécessaires à cette évaluation de vos compétences pratiques sont disponibles sur `Espadon`, cours `Sécurité Informatique`, dossier `exam`.

Afin que je puisse récupérer votre travail sur ma clé USB à la fin de l'évaluation vous développerez tous vos programmes dans un seul et unique répertoire dont le nom est `<nom>_<prenom>` en minuscules. Vous y mettrez les codes source de vos programmes ainsi que les réponses aux différentes questions (fichier texte, OpenOffice, LibreOffice, Word,...).

⋆ ⋆ ⋆

1. Contrôle d'une application dont vous avez le code

Pour cet exercice, récupérez le fichier `SecuJavaExo1.java` et répondez aux questions suivantes :

(a) Quelle est la commande à taper pour exécuter cette application (une fois compilée) en mode local ?

(b) Quelle est la commande à taper pour exécuter cette même application en mode sécurisé cette fois ? Quel est le résultat ?

(c) Fournir le fichier de politique de sécurité `exo1.policy` donnant les permissions **nécessaires et suffisantes** (i.e. l'ensemble minimal de permissions) pour pouvoir exécuter cette application en mode protégé.

2. Contrôle d'une application dont vous n'avez pas le code

Mêmes questions que précédemment, sauf que cette fois ci vous ne disposez pas du code source de l'application à exécuter... Récupérez pour cela le fichier `SecuJavaExo2.class`

(d) Question supplémentaire : quelle est la commande à taper pour afficher la liste des permissions nécessaires à l'exécution de cette application ?

3. Utilisation de l'API de cryptographie

Dans cet exercice il vous est demandé de déchiffrer le contenu d'un fichier. Un squelette de programme vous est fourni dans le fichier `SecuJavaExo3.java` avec notamment une méthode vous rappelant (si nécessaire...) comment lire le contenu d'un fichier en Java. Vous pourrez bien évidemment vous en inspirer pour, cette fois, non plus afficher les caractères lus, mais pour les déchiffrer et afficher le résultat.

Cet exercice est très similaire à ce que vous avez fait en TP. La différence est vous utiliserez des clés générées en dehors de votre programme. En effet, afin de chiffrer le fichier, je me suis généré une paire de clés publique/privée via l'algorithme RSA. J'ai chiffré le message avec ma clé privée (que vous ne connaissez donc pas). Vous trouverez ma clé publique dans le certificat `munier.cer` que je vous fournis. Le fichier à déchiffrer est `foo.txt.crypt`

Afin de vous aider (un peu), voici les principales étapes à réaliser :

(a) Importer le certificat `munier.cer` dans un *keystore* (ex : fichier `keystore` dans votre répertoire de travail et alias `munier`).

(b) Dans votre programme, vous devrez tout d'abord créer une instance de la classe `KeyStore` (format `jks`), puis charger le fichier `keystore` créé précédemment via un `FileInputStream`. NB : mettez `null` pour le mot de passe ; vous n'en aurez pas besoin puisque vous n'accéderez qu'à des informations publiques.

(c) Pour le `Cipher`, l'algorithme utilisé est `"RSA/ECB/PKCS1Padding"`. Pour obtenir ma clé publique, vous devrez tout d'abord utiliser la méthode `getCertificate` pour récupérer le certificat associé à mon alias dans le *keystore*. Sur ce certificat, vous utiliserez la méthode `getPublicKey` pour récupérer (enfin) ma clé publique.

(d) La suite est normalement identique à ce que vous avez fait en TP...

Document rédigé en LaTeX sous Linux.