

Université Marie and Louis Pasteur, ISIFC, Cryptographie. TP sur le chiffrement Asymétrique RSA.

Jean-François Couchot

October, 9th 2026

1 Prise en main du chiffrement et du déchiffrement

L'objectif de cet exercice est de comprendre l'intérêt d'un chiffrement avec des clés publiques et privées.

Vous avez accès à un dossier partagé partagé appelé contenant un sous dossier nommé **annuaire** et un sous dossier nommé **canal**. Le premier sous-dossier représente un annuaire tandis que le second représente un canal où les informations circulent.

1. Chaque étudiant · e utilise le script Python du cours pour générer sa paire de clés (2048 bits) selon RSA. La clé privée `mPrivKey.pem` est gardée tandis que la clé public `mPubKey.pem` est d'abord renommée `prenom_nom_pubKey.pem` pour être déposée ensuite dans l'**annuaire**.
2. Chaque étudiant · e construit un message secret contenant son prénom ainsi que quelques mots. Puis un · e destinataire est choisi · e (ou plutôt sa clé de chiffrement dans l'**annuaire**). Le message est chiffré selon RSA/OAEP, renommée en "message_pour_prenom_nom_X.bin" et déposé dans dans le dossier **canal**. Si le nom existe déjà, changez X.
3. Répéter la question précédente plusieurs fois fois pour que chaque personne ait plusieurs messages à déchiffrer ensuite.
4. Déchiffrez tous les messages qui vous sont destinés (et remercier les personnes qui les ont écrits).
5. Choisissez les messages d'une autre personne et essayez de les déchiffrer avec votre clé privée. Pourquoi n'y arrivez-vous pas?
6. Choisissez un des messages qui vous est destiné et modifiez un élément dedans. Quel serait votre rôle si vous faisiez ceci? Constater que RSA/OAEP vous informe qu'il y a une erreur.

2 Pourquoi le chiffrement hybride est-il indispensable ?

l'objectif de cet exercice est de mesurer empiriquement l'écart de durée d'exécution entre un chiffrement symétrique (AES-256) et asymétrique (RSA-2048). On comprendra alors le principe du chiffrement hybride employé dans les protocoles réels (TLS, PGP, SSH).

1. Générez une paire de clés RSA de 2048 bits (`mPubKey.pem` et `mPrivKey.pem`) via `Crypto.PublicKey.RSA`.

2. Générez une clé symétrique AES de 256 bits.
3. Générez un ensemble de 1000 messages court aléatoires différents de 32 octets.
4. Mesurez le temps nécessaire T_{RSA} pour chiffrer et déchiffrer tout cet ensemble de messages avec RSA-OAEP.
5. Faites de même pour calculer T_{AES} avec AES-GCM.
6. Calculez le ratio d'accélération $\frac{T_{\text{RSA}}}{T_{\text{AES}}}$.
7. Essayez de chiffrer une image de petite taille (1 MO) pour constater que ce n'est pas possible en raison de la taille de celle-ci.
8. Donnez une conclusion quant-à l'utilisation du chiffre RSA.