TD5 – Chiffrement avec RSA

4ASTI – Programmation Java – B.Nguyen

Exercice : Chiffrement avec RSA (Rivest, Shamir, Adleman)

Le chiffrement RSA est un chiffrement dit *asymétrique* car les deux participants, Alice et Bob ne partagent pas la même clé. Chaque participant doit construire une « paire » de clés RSA, composée de deux parties :

- Une partie dite clé publique (PK_{Alice})
- Une partie dite clé privée (SK_{Alice})

La clé publique peut être partagée à tout le monde, la clé privée doit rester secrète, connue uniquement de son propriétaire. Lorsqu'on chiffre un message avec la clé publique, seule la personne avec la clé privée peut la déchiffrer. Ainsi, on peut envoyer des informations secrètes à Alice si on chiffre un message avec la clé publique d'Alice.

Question 1 – Construction de la clé

Pour générer sa paire de clés, on utilise : `java.security.KeyPair` et `java.security.KeyPairGenerator` de la manière suivante :

```
KeyPairGenerator generator = KeyPairGenerator.getInstance("RSA");
generator.initialize(2048);
KeyPair pair = generator.generateKeyPair();
```

Chaque partie de la clé peut être récupérée de la manière suivante :

```
java.security.PrivateKey privateKey = pair.getPrivate();
java.security.PublicKey publicKey = pair.getPublic();
```

Que fait generator.initialize(2048) ?

Générez une paire de clés et affichez la clé publique sous forme de String. Que voyez-vous ?

Vous aurez besoin de deux informations pour reconstruire la clé : un exposant (publicExposant) et un modulo (modulus), en utilisant le constructeur : public RSAPublicKeySpec(BigInteger modulus, BigInteger exponent)

Vous pouvez récupérer ces informations de la manière suivante :

```
RSAPublicKey rsaPub  = (RSAPublicKey)publicKey;
BigInteger publicKeyModulus = rsaPub.getModulus();
BigInteger publicKeyExponent  = rsaPub.getPublicExponent();
System.out.println("publicKeyModulus: " + modulus);
System.out.println("publicKeyExponent: " + exponent);
```

Pour reconstruire la clé, vous pouvez procéder de la manière suivante en utilisant une KeyFactory:

```
RSAPublicKeySpec newKey = new RSAPublicKeySpec(modulus, exponent);
KeyFactory keyFactory = KeyFactory.getInstance("RSA");
PublicKey key2 = keyFactory.generatePublic(newKey);
```

Trouvez un moyen de stocker la clé dans un fichier, de la recharger, et de créer un nouvel objet avec les mêmes paramètres, en écrivant simplement en entête du fichier (en format binaire avec un OutputStream) un int représentant le nombre de bytes pour le module, puis un `int` représentant le nombre de bytes de l'exposant, puis les bytes du module, puis les bytes de l'exposant.

Faites de même avec la clé privée. Sauvegardez ces deux clés. Chargez les clés d'un de vos camarades et vérifiez que les valeurs sont correctes.

Question 2— Chiffrement et déchiffrement

Pour chiffrer un message, il faut construire un objet Cipher de la manière suivante :

```
Cipher encryptCipher = Cipher.getInstance("RSA");
encryptCipher.init(Cipher.ENCRYPT_MODE, publicKey);
```

Si vous chiffrez avec la clé publique que vous avez générée, qui peut déchiffrer ? Si vous chiffrez avec la clé publique de votre camarade et que vous avez chargée depuis un fichier, qui pourra déchiffrer ?

Le texte à chiffrer doit être un tableau de bytes, qu'on va écrire dans un fichier binaire. On peut le générer à partir du message secret de la manière suivante :

```
String messageSecret = "Alea Jacta Est !";
byte[] messageSecBytes = messageSecret.getBytes(StandardCharsets.UTF_8);
byte[] messageSecChiffreBytes = encryptCipher.doFinal(messageSecBytes);
```

Affichez le message chiffré à l'écran et stockez le tableau de byte correspondant dans un fichier en format binaire.

Pour déchiffrer le texte, il faut appliquer la transformation suivante :

```
Cipher decryptCipher = Cipher.getInstance("RSA");
decryptCipher.init(Cipher.DECRYPT_MODE, privateKey);
byte[] bytesDechiffres = decryptCipher.doFinal(messageSecChiffreBytes);
String messageDec = new String(bytesDechiffres, StandardCharsets.UTF_8);
System.out.println(messageDec);
```

Question 3 – Application

Chiffrez un message pour votre camarade en utilisant leur clé publique qu'ils vous auront fourni dans un fichier. Stockez ce message dans un fichier et transmettez leur le fichier. Votre camarade doit déchiffrer le fichier en utilisant sa clé privée, puis afficher le message. Vérifiez que votre message est bien le bon !