1 TD

Exercice 1 : Cryptage matriciel à clef secrète
Pour coder des messages pouvant contenir des caractères de diverses langues, on représente chaque caractère par un entier sur 16 bits (codage UTF16 par exemple). On décide ensuite de crypter le message par blocs de 2 caractères, en multipliant chaque vecteur v des 2 entiers représentant un bloc par une matrice carrée A de taille 2 modulo un entier p qui sera supposé premier dans les questions 1 à 6. La matrice A est tenue secrète, c'est la clef secrète du système de cryptage.

1. On suppose que p est un nombre premier qui permet de représenter de manière unique tous les entiers sur 16 bits par leur classe dans $\mathbb{Z}/p\mathbb{Z}$. Quel est le nombre minimal de bits de p ?
2. On admettra que $p = 65537$ est bien premier. On pose :
$$p = 65537, \quad A = \begin{pmatrix} 263 & 4122 \\ 1 & 1799 \end{pmatrix}$$
Crypter le vecteur v correspondant aux deux caractères "intégrale double" et "intégrale triple" représentés respectivement par les entiers 8748 et 8749, i.e. calculer $w = Av$ modulo p
3. Déterminer tous les vecteurs z à coefficients dans $\mathbb{Z}/p\mathbb{Z}$ tels que $Az = Av$ modulo p pour $p = 65537$.
En déduire tous les vecteurs z à coefficients entiers qui vérifient $Az = Av$ modulo p.
Si on impose que les coordonnées de z sont dans l'intervalle du codage sur 16 bits, combien y-a-t-il de solutions ?
4. Que se passerait-il si on avait choisi la même matrice mais $p = 6551$?
5. Expliquer comment on peut décrypter, i.e. étant donné w quelconque, comment on trouve v tel que $Av = w$ modulo p.
6. Quel est le nombre de matrices A possibles pour $p = 65537$? Cela vous parait-il suffisant pour résister à une attaque de la clef secrète A par force brute ? Peut-on imaginer un autre type d'attaque ? Comment pourrait-on renforcer la sécurité ?
7. Dans cette question, on suppose que $p = 2^{16}$ et n'est donc pas premier. Peut-on appliquer l'algorithme du pivot de Gauss pour résoudre $Az = w = Av$ si on travaille modulo p bien que p ne soit alors pas premier ?
En est-il de même pour la matrice
$$p = 65536, \quad A = \begin{pmatrix} 262 & 4122 \\ 1 & 1799 \end{pmatrix}$$
8. Quel(s) avantage(s) et inconvénient(s) y-a-t-il à travailler modulo 2^{16} par rapport à $p = 65537$?

Exercice 2 : RSA jouet On prend $p = 17$ et $q = 13$ donc $n = 221$.
Déterminer $\varphi(n)$. Vérifier qu'on peut utiliser $e = 7$ comme exposant de chiffrement. Calculer l'exposant de déchiffrement d. Chiffrer $M = 3$. Déchiffrer $C = 198$.
Pour p et q quelconque, estimez la complexité des opérations de calcul des clefs, chiffrement, déchiffrement.

Exercice 3 : Diffie-Hellman, secret commun
Exemple jouet :
Alice et Bob choisissent de travailler dans $\mathbb{Z}/19\mathbb{Z}$ et d'utiliser $g = 2$ qui est un générateur de $\mathbb{Z}/19\mathbb{Z}^*$.

1. Alice choisit $a = 7$ et Bob choisit $b = 13$. Donner A et B puis le secret commun.
2. Que se passe-t-il si Alice choisit $a = 25$? À quelle valeur maximale pour a et b Alice et Bob peuvent-ils se restreindre ?
3. Déterminer la table de toutes les puissances de 2 dans $\mathbb{Z}/19\mathbb{Z}$.
4. Alice et Bob choisissent deux autres entiers a et b et s'envoient $A = 4$ et $B = 17$. En utilisant la table, déterminer les valeurs de a et b. Quelle est la valeur du secret commun ?

Sécurité :
On a vu qu'une personne qui connait les entiers A et B (par exemple en espionnant les échanges entre Alice et Bob) pouvait calculer le secret commun si on travaille dans $\mathbb{Z}/19\mathbb{Z}$. Pour espérer sécuriser le secret commun, il faut travailler dans $\mathbb{Z}/p\mathbb{Z}$ avec p un nombre premier plus grand.

1. Commençons par essayer avec $p = 65537$ et $g = 3$.
   (a) Vérifier que 3 est un générateur de $\mathbb{Z}/p\mathbb{Z}^*$.
   (b) Si Alice choisit $a = 12345$, combien doit-elle effectuer de multiplications pour calculer A par l'algorithme de la puissance rapide ?

(c) À quelle valeur maximale pour a et b Alice et Bob peuvent-ils se restreindre ?

(d) Quelle est la taille de la table des puissances de 3 modulo p ? Comparez avec la question (b). La sécurité du secret commun vous semble-t-elle suffisante ?

2. On prend maintenant un nombre premier dont l'écriture en base 2 comporte exactement 1024 bits, et tel que $g = 2$ est un générateur de $\mathbb{Z}/p\mathbb{Z}^*$. La sécurité du secret commun vous semble-t-elle suffisante ?

Exercice 4 : attaque contre RSA (module partagé)
Alice et Bob décident de recevoir des messages cryptés en utlisant le système RSA. Ils publient donc chacun leur clef publique.

On va étudier une attaque qu'un espion peut exploiter si Alice et Bob utilisent tous les deux la même valeur de n. On suppose donc que la clef publique d'Alice est (n, e_A), et celle de Bob (n, e_B). On suppose que Catherine envoie une même information m à Alice et à Bob, donc envoie le message crypté $c_A = m^{e_A} \pmod n$ à Alice et le message crypté $c_B = m^{e_B} \pmod n$ à Bob. Un espion Daniel intercepte les deux messages cryptés.

Dans l'exercice, on prendra $n = 4897$ pour pouvoir faire des calculs à la calculatrice.

1. Dans cette question on suppose qu'on connait la factorisation de n : $n = 59 \times 83$. Expliquer pourquoi on peut prendre $e_A = 71$ et $e_B = 227$. Déterminer la clef privée d'Alice.
2. Déterminer une identité de Bézout entre 71 et 227.
3. En déduire deux entiers u et v tels que $m = c_A^u c_B^v \pmod n$
4. Daniel intercepte $c_A = 1846$ et $c_B = 487$. Déterminer m sans utiliser la factorisation de n
5. Expliquer pourquoi Daniel ne peut pas utiliser la factorisation de n dans une attaque réelle contre RSA.

Exercice 5 : attaque RSA par itération
Exemple jouet. On prend $n = 77$ et une clé publique $c = 7$, le message original est $a = 2$. Retrouver le message original par itération de la fonction de cryptage sur le message crypté. En général, quelles sont les valeurs de c vulnérables à une attaque par itération ?

Exercice 6 : Logarithme discret Calculer le logarithme en base 2 de 3 modulo 11.

Exercice 7 : Logarithme discret et restes chinois.
On considère le nombre premier 101 et on cherche, s'il existe, un entier $x \in \mathbb{N}$ tel que $3^x = 2$ [101].

1. À l'aide de l'algorithme d'exponentiation modulaire, calculer 3^4, 2^4, 3^{25} et 2^{25} modulo 101.
2. Déterminer des entiers a, b tels que $(3^4)^a = 2^4$ et $(3^{25})^b = 2^{25}$ modulo 101 (pour déterminer a, on pourra chercher les puissances successives de 3^4 modulo 101).
3. Déterminer x tel que $x = a$ [25] et $x = b$ [4], vérifier que pour cet x, on a $3^x = 2$ [101].

Exercice 8 : Logarithme discret p-adique Calculer le logarithme en base 2 de 3 modulo 197 en utilisant 2 fois le logarithme en base 7 au lieu du logarithme en base 49.

2 TP

Exercice 0 Vérifiez les résultats de TD.

Exercice 1 : Générer une paire de clefs
Génèrez deux grands nombres premiers p et q au hasard, en utilisant par exemple les fonctions `nextprime` et `randint` de Xcas ou le test de Miller-Rabin si votre langage préféré n'a pas de test de primalité

Exercice 2 : Codage et décodage d'un message (sur PC)
On transforme une chaine de caractère en une liste d'entiers et réciproquement (avec `asc` et `char` en Xcas, ou l'application répétée de `ord` et `chr` en Python). Pour le moment on code caractère par caractère, sans s'inquiéter de la sécurité du codage. Pour coder/décoder une liste `l` d'entiers, on peut utiliser `pow(l,c,n)` en Xcas et Python. En utilisant la paire de clefs de l'exercice 1, codez un message puis décodez ce message pour vérifier. Décodez le message authentifié situé à l'URL
`http://www-fourier.ujf-grenoble.fr/~parisse/mat249/rsa1`.

Exercice 3 : Attaque simple
On a vu que le codage monoalphabétique n'est pas une bonne idée, une attaque possible étant la recherche de fréquences, ici on peut utiliser une attaque encore plus simple : la personne souhaitant décoder un message codé avec une clef publique sans en connaitre la clef secrète calcule la liste des $a^c \pmod n$ pour les 256 valeurs possibles de a et compare au message. Décodez de cette manière le message situé à l'URL
`http://www-fourier.ujf-grenoble.fr/~parisse/mat249/rsa2`

Exercice 4 : Padding aléatoire
Pour parer à cette attaque, on va augmenter le nombre de valeurs possibles de a pour que le calcul de la liste de

toutes les puissances des a possibles soit trop long. Plusieurs stratégies sont possibles, l'une d'elle consiste à ajouter à a un multiple aléatoire de 256. Comment la personne qui recoit un message crypté retrouvera-t-elle le message en clair ? Implémenter cette méthode.

Exercice 5 : Groupement de lettres
On peut aussi grouper par paquets de x caractères et on associe à un groupe de caractères l'entier correspondant en base 256. Par exemple, si on prend des groupes de $x = 3$ caractères, "ABC" devient `65*256^2+66*256+67` car le code ASCII de A, B, C est respectivement 65, 66, 67. Donner une condition reliant n et x pour que le décodage redonne le message original. Choisissez une paire de clefs vérifiant cette condition pour $x = 3$ (calculatrices avec entiers représentés par des flottants) ou $x = 8$ (autres). Ecrire un programme de codage et de décodage avec groupement (on commencera par compléter le message original par des espaces pour qu'il soit un multiple de 8 caractères, en Xcas et Python l'instruction `len` permet de connaitre la taille d'une chaine de caractères, en Xcas, on pourra utiliser la fonction `convert(.,base,256)`d'écriture en base 256).

Exercice 6 : Sécurité du codage 1
Vérifiez sur l'exemple de l'exercice 1 que la connaissance de $\varphi(n)$ et de n permet de calculer p et q par résolution d'une équation de degré 2. Si on connait seulement c et s, peut-on retrouver $\varphi(n)$? La sécurité du codage repose donc sur la difficulté de factoriser n. Tester sur des entiers de taille croissante le temps nécessaire au logiciel pour factoriser p et q. Une valeur de n de taille 128 bits, 512 bits, 1024 bits parait-elle suffisante ?

Exercice 7 : Sécurité du codage 2
Le choix de c et de s est aussi important. Pour le comprendre, prenons $p = 11$ et $q = 13$. Représentez pour différentes valeurs de c les points $(a, a^c \pmod n)$, plus le dessin obtenu est aléatoire, plus il sera difficile à une personne mal intentionnée de déchiffrer un message sans connaitre la clef. En Xcas, on pourra utiliser les instructions `seq` pour générer une suite de terme général exprimé en fonction d'une variable formelle, et `scatterplot(l)` qui représente le nuage de points donné par une liste `l` de couples de coordonnées. En Python, on peut utiliser l'instruction `plot` de `matplotlib`. Observez en particulier les cas où c n'est pas premier avec $\varphi(n)$ (comment voit-on que RSA ne fonctionne pas ?) et également le cas $c = 3$.

Exercice 8 : attaque par les restes chinois
Une personne souhaite envoyer le même message x à trois destinataires différents, ayant chacun leur propre clef publique $c = 3, N_1$, $c = 3, N_2$ et $c = 3, N_3$ avec $c = 3$ pour les 3 destinataires. Il envoie donc $y_1 = x^3 \pmod{N)_1}$, $y_2 = x^3 \pmod{N)_2}$ et $y_3 = x^3 \pmod{N)_3}$. Une personne mal intentionnée arrive à intercepter y_1, y_2 et y_3. En appliquant les restes chinois, elle peut en déduire x. Par exemple, retrouver x sans chercher à factoriser les clefs pour

```
46693373016 % 180711261397,
(-111575037168) % 840724735099,
(-18270191368) % 372130013641
```

Exercice 9 : attaque RSA On suppose qu'on connait un couple de clé secrète/publique
`96664445695884629095302836378328116675824715046626033` pour 65537 pour n valant
`14853687637916039711650329532817378529646911522656401 13`
En déduire la factorisation de n.

Exercice 10 : Attaque RSA fraction continue. Fonctionne si la clef privée d est petite et si p et q sont du même ordre de grandeur :

$$p > q, \quad p < 2q$$

Le principe consiste à calculer les réduites de e/n. Soit $k \in]0, d[$ tel que

$$ed = 1 + k\phi(n) = 1 + k(n + 1 - p - q) = kn + 1 + k(1 - p - q)$$

On divise par dn :

$$\frac{e}{n} = \frac{k}{d} + \frac{1 + k(1 - p - q)}{dn}$$

donc :

$$\begin{aligned}
|\frac{e}{n} - \frac{k}{d}| &= \frac{k(p+q-1)-1}{dn} \\
&\leq \frac{k(p+q)}{nd} \\
&\leq \frac{kq(\frac{p}{q}+1)}{nd} \\
&\leq \frac{3kq}{nd} \\
&\leq \frac{3k}{\sqrt{n}d} \\
&< \frac{3}{\sqrt{n}}
\end{aligned}$$

Si $3/\sqrt{n} < 1/(2d^2)$, alors les résultats connus sur les fractions continues permettent de conclure que k/d est une réduite de e/n. On calcule ces réduites en utilisant les résultats intermédiaires de l'algorithme d'Euclide étendu et on teste si $m^{de} = m \pmod{n}$.
Mettre en oeuvre cette attaque, par exemple pour `n=24121770232611008805519974758722894470290624535341`
`c=70789631335552059501741838043400261591217206072 29`

Exercice 11, log discret Programmer l'algorithme Baby-step giant-step en stockant dans une liste triée les pas de bébé pour comparaison par dichotomie.

Exercice 12, Pollard-rho Programmer l'algorithme de Pollard-rho pour chercher un facteur de taille au plus environ 10 digits d'un entier.

Exercice 13, Pollard-rho pour le logarithme discret Programmer l'algorithme de Pollard-rho pour chercher le logarithme discret modulo un nombre premier p pour des nombres d'ordre n (avec n divisant $p-1$).