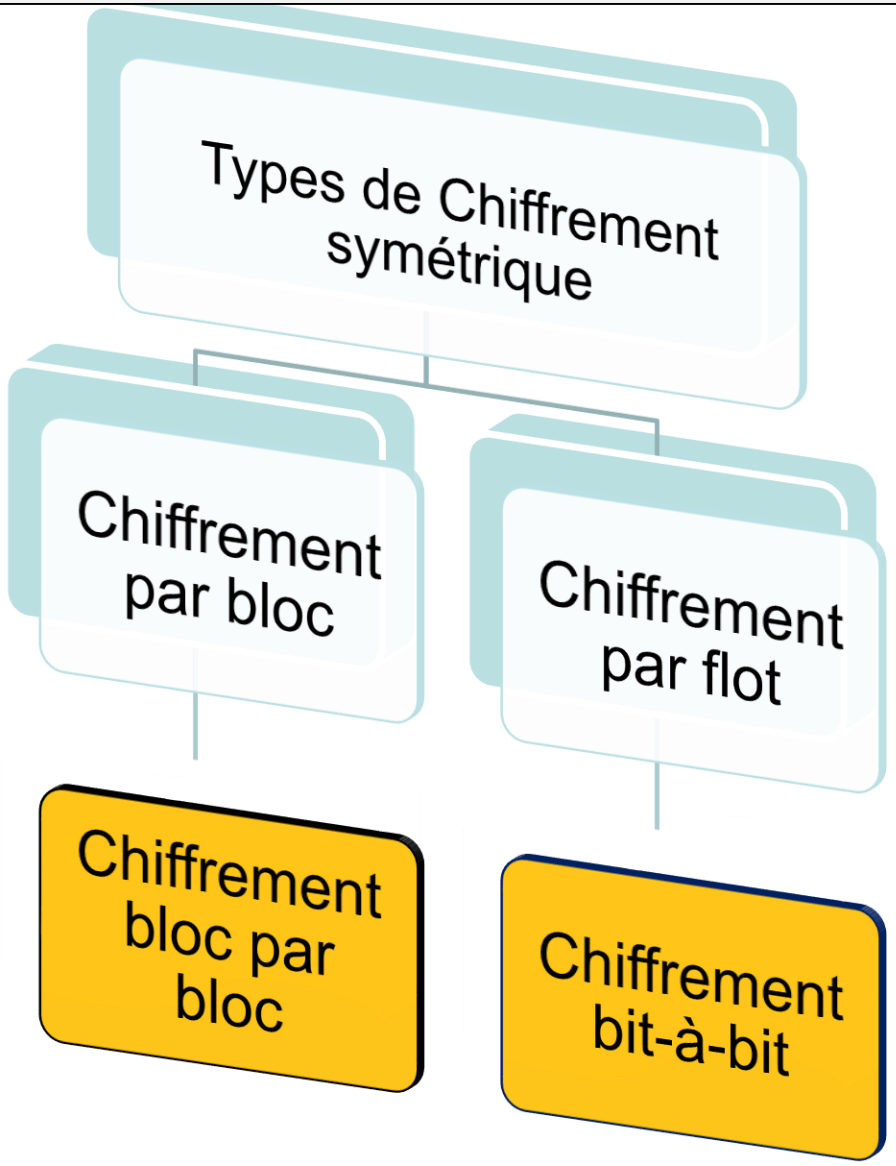
Module: Cryptographie

3ème année licence en Informatique

Par : Prof. Cherif Foudil

Chiffrement par flot



Chiffrement par flot

Plan

- Chiffrement par flot
- Génération des nombres pseudo-aléatoires
- Algorithme RC4
- Sécurité de RC4

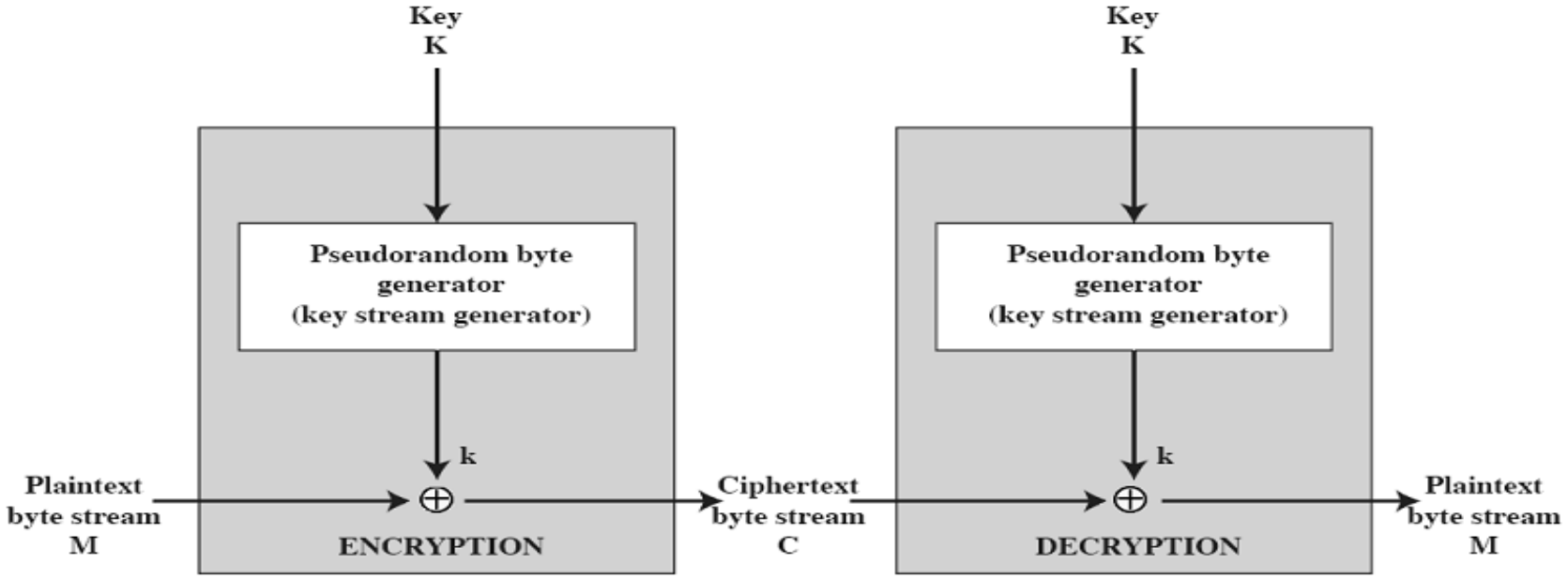
Chiffrement par flot

- Le chiffrement de flux ou chiffrement par flot (en anglais stream cipher)
- Un chiffrement par flot arrive à traiter les données de **longueur quelconque** et n'a pas besoin de les découper.
- La structure d'un chiffrement par flot repose sur un générateur de clés produit une séquence de clés $r_1, r_2, \ldots, r_i$
- La longueur de la clef est égale à celle du message.

Chiffrement par flot

- Dans les algorithmes de chiffrement par flot, une suite d'octets ou de bits r_i est produite à partir de la clé. Cette suite est combinée aux octets ou aux bits du clair m_i pour donner les octets ou les bits du chiffré c_i, suivant la formule $c_i = m_i \oplus r_i$.

Chiffrement par flot: Exemple



- Message : « **SALUT** »
  01010011 01000001 01001100 01010101 01010100
- Clef$_{(\text{générée aléatoirement})}$=
  01110111 01110111 00100100 00011111 00011010
- *((Salut)$_{binaire}$ xor clef)=*
  00100100 00110110 01101000 01001010 01001110
- Le message chiffré est $6jJM

Chiffrement par flot: Les Algorithmes

- 1-RC4:

- RC4, le plus répandu, conçu en **1987** par Ronald Rivest, utilisé notamment par le protocole WEP du WiFi (Wired Equivalent Privacy, de la norme 802.11). est remplacé par le WPA (Wi-Fi Protected Access), mais celui-ci utilise toujours le RC4

- Le plus important des algorithmes de chiffrement symétrique

Chiffrement par flot: Les Algorithmes

- A5/1(**1994**)
- utilisé dans les téléphones mobiles de type GSM pour chiffrer la communication par radio entre le mobile et l'antenne-relais la plus proche, d'autres variantes existent: A5/2 et A5/3(par bloc),
- Ils utilisent une clé de 64 bits
- **Principe**: Une conversation GSM s'effectue par division en blocs temporels, chacun dure 4,6 millisecondes et contient 2×114 bits pour les deux canaux de communication (full-duplex).

Chiffrement par flot: Les Algorithmes

- Une clé de session K est utilisée pour la communication. Pour chaque bloc, K est mélangée à un compteur de blocs et le résultat est utilisé comme état initial d'un générateur de nombres pseudo-aléatoires qui produit 228 bits. Ceux-ci servent au chiffrement après un XOR avec les données des deux canaux

- K1 = clé(k) + compteur de bloc
- G(k1) ➔ 228 bits générateur pseudo-aléatoires
- Message(2 x 214) XOR G(K1) ➔ C(message codé)

Chiffrement par flot: Les Algorithmes

- En 1994, l'architecture du A5/1, à l'origine secrète, est publiée.
- En 1999, les versions 1 et 2 sont complètement spécifiées.
- En 2000, on comptait environ 130 millions d'utilisateurs du GSM basé sur A5/1.

Chiffrement par flot: Les Algorithmes

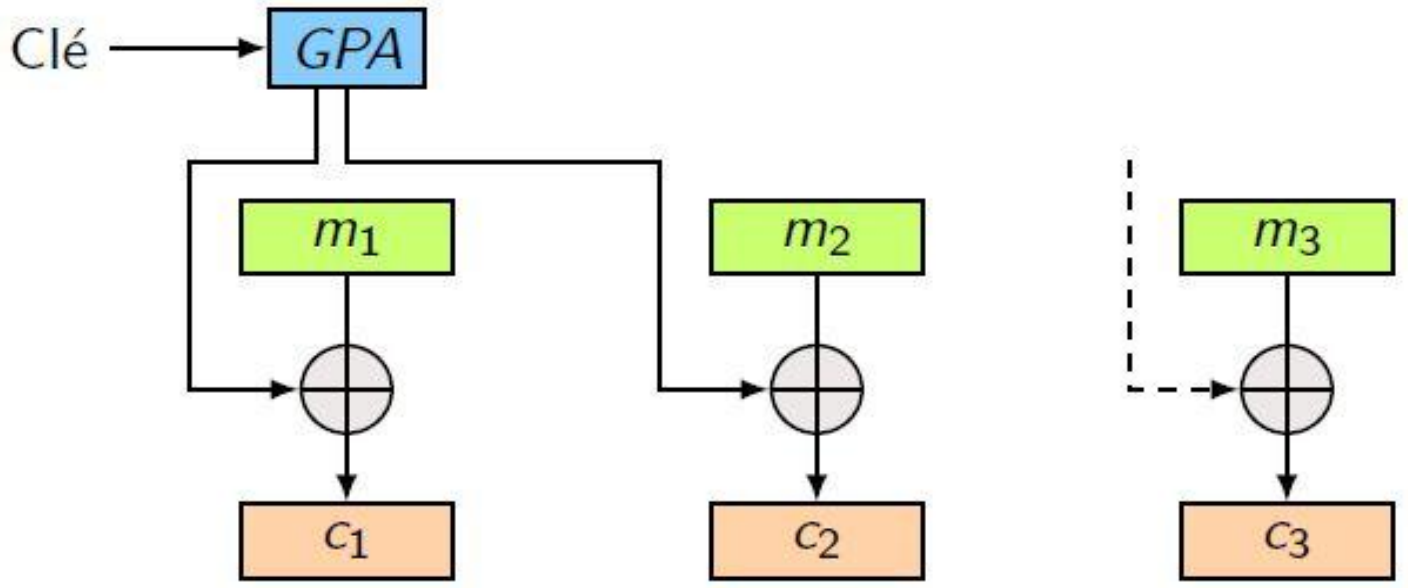
- E0 est un algorithme de chiffrement par flot utilisé par le protocole Bluetooth pour protéger les transmissions. Il crée une suite pseudo-aléatoire avec laquelle on effectue un XOR avec les données. La clé peut avoir une taille variable mais sa longueur est généralement de 128 bits. *Cassé en 2004*
- E0 se divise en trois parties :
  - préparation de la clé;
  - génération du flux ;
  - chiffrement.
- *Autres algorithmes:*

FISH - Helix · ISAAC · LEVIATHAN · MUGI · Panama ·SEAL · SOBER · WAKE

Tous les algorithmes utilisent des registres de décalage

Génération des nombres pseudo-aléatoires

- En pratique, un chiffrement par flux est réalisé en utilisant un générateur pseudo-aléatoire (GPA).

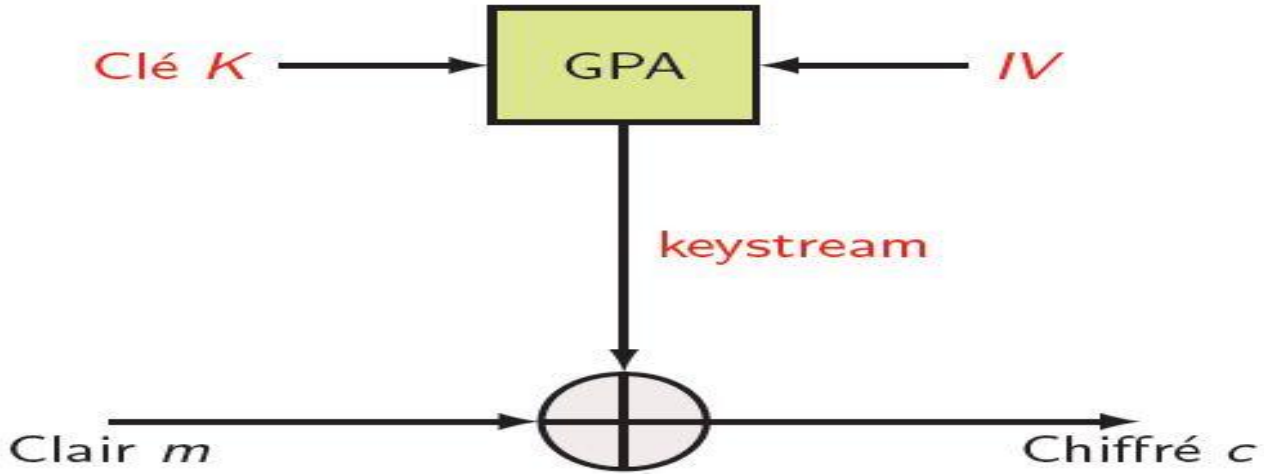


Chiffrement par flot

Génération des nombres pseudo-aléatoires

Problème:
Même clé => même aléa

Solution:
On utilise une entrée auxiliaire IV (vecteur d'initialisation)



L'algorithme RC4(caractéristiques)

Algorithme développé par Ron Rivest en 1987

- RC = Rivest Cipher (ou encore Ron's Code)
- Breveté et tenu secret par la société RSA Inc.
- détails postés anonymement sur Usenet en 1994. . .
- mais jamais approuvés officiellement par RSA
- Les implémentations matérielles ou logicielles: facile à mettre en œuvre.
- excellente performance (vitesse de chiffrement)
- RC4 est un chiffrement par flux (octet par octet)

L'algorithme RC4(applications)

Il est utilisé dans:

- les protocoles ***WEP*** *(Wired Equivalent Privacy*, de la norme 802.11),
- **WPA** (*Wi-Fi Protected Access*)
- ainsi que le protocole **SSL/TLS**.

L'algorithme RC (Paramètres)

Longueur de clé : variable, entre 40 et 1024 bits.

- L'état interne de RC4 est formé d'un tableau de 2^n éléments de n bits. A chaque instant, ce tableau correspond à une permutation des mots de n bits. Généralement, on prendra n=8, qui correspond à un tableau de 256 octets.
- 1 256

L'algorithme RC4(Principe Général)

RC4 fonctionne de la façon suivante :

- la clé RC4 permet d'initialiser un tableau de 256 octets en répétant la clé autant de fois que nécessaire pour remplir le tableau.
- Par la suite, des opérations très simples sont effectuées :
  - les octets sont déplacés dans le tableau,
  - des additions sont effectuées, etc.

  Le but est de mélanger autant que possible le tableau.
- Finalement on obtient une suite de bits pseudo-aléatoires qui peuvent être utilisés pour chiffrer les données via un XOR.

L’algorithme RC4: Description détaillée

Pour générer le flot de bits, l'algorithme dispose d'un état interne, tenu secret, qui comprend deux parties :

- une permutation *S* de tous les 256 octets possibles,
- deux pointeurs *i* et *j* de 8 bits qui servent d'index dans un tableau.

- La permutation est initialisée grâce à la clé de taille variable, typiquement entre 40 et 256 bits, grâce au *key schedule* de RC4.

Chiffrement RC4: *Key Schedule*

- ***Génération de la permutation:***
  - La permutation *S* est initialisée grâce à la clé *K*.
  - La longueur de la clé varie de 1 à 256 octets (8 à 2048 bits).
  - En pratique, elle est souvent choisie de taille égale à 5 octets (pour 40 bits) ou 16 octets (pour 128 bits).
  - La permutation se présente sous la forme d'un tableau de 256 entrées.

Chiffrement RC4: *Key Schedule*

[0] [1] [2] [3] [4] ... [255]

Initialisation (pour une clé K de longueur ℓ)

RC4 key schedule

```
for i = 0 to 255 do
    S[i] := i                                   // permutation identité
j := 0
for i = 0 to 255 do
    j := (j + S[i] + K[i mod ℓ]) mod 256
    swap(S[i], S[j])
```

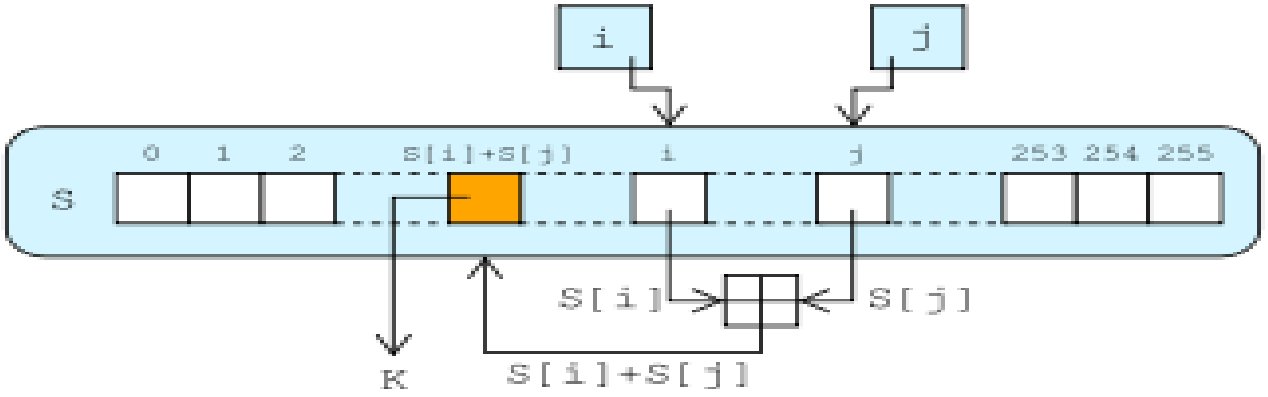
Longueur de clé: de 5 octets (40 bits) à 256 (= 2048 bits !)

À la fin, le tableau pourrait se présenter comme suit :

[184] [106] [163] [64] [70] ... [201]

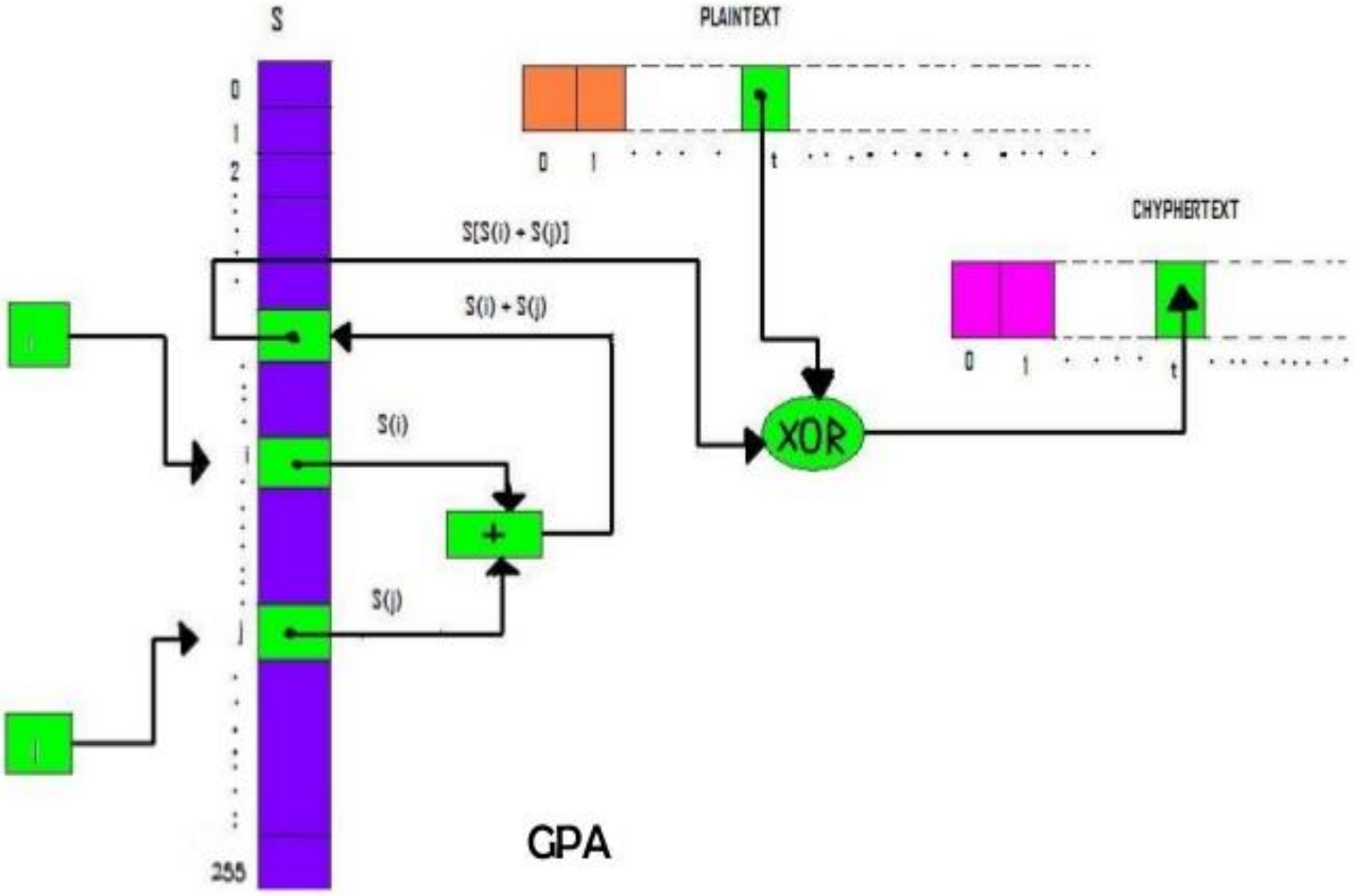
Chiffrement RC4: Génération de flux pseudo- aléatoire

- Pour générer un nouvel octet aléatoire, on applique les opérations suivantes :




```
i := 0
j := 0
tant_que générer une sortie:
i := (i + 1) mod 256
j := (j + S[i]) mod 256
swap(S[i], S[j])
key_chiffrement  = S[(S[i] + S[j]) mod 256]
result_chiffré = key_chiffrement XOR octet_message
fintant_que
```

Chiffrement RC4: Génération de flux pseudo- aléatoire



Chiffrement RC4: Attaque (Sécurité) FMS

- En 2001, S. Fluhrer, I. Mantin et A. Shamir (FMS) ont présenté une nouvelle attaque.
- Parmi toutes les clés possibles en RC4, les statistiques sur les premiers octets du flux générés montrent un certain biais(écart) qui permet de retrouver des informations sur la clé.
- Les premiers octets du flux utilisés pour le chiffrement *ne sont pas aléatoires*.

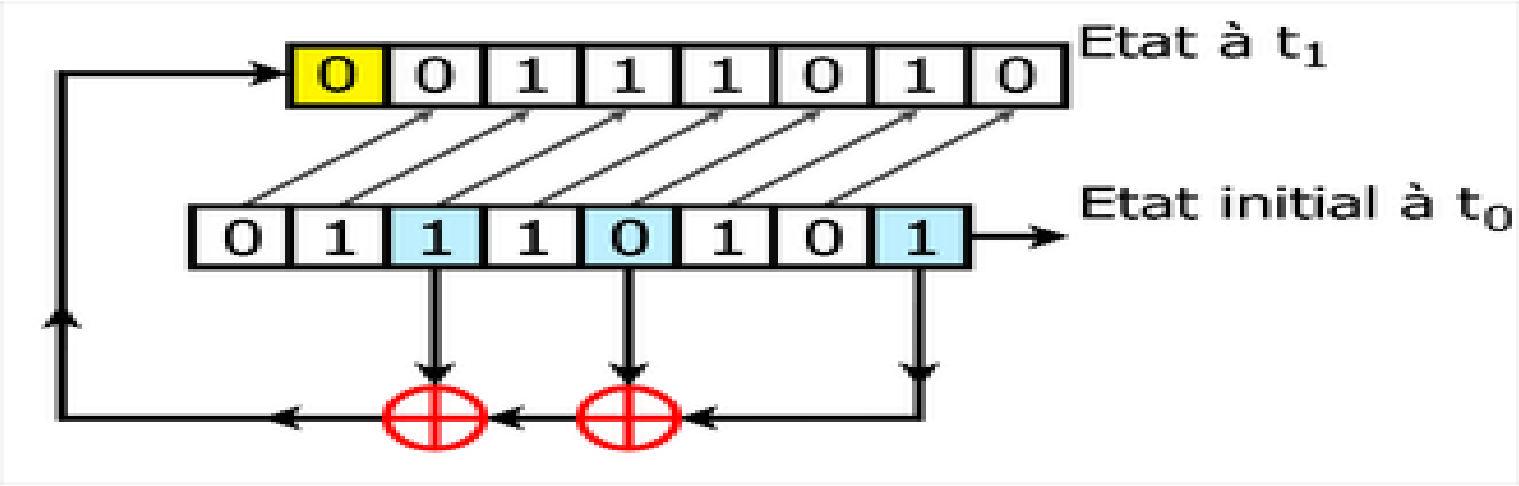
Chiffrement RC4: Sécurité FMS

- C'est ce type d'attaque qui a été utilisée pour casser le WEP des réseaux sans fil, une action qui a abouti à la mise en place d'une version améliorée du chiffrement, à savoir WPA.

- Il semblerait que le RC4 soit toujours sûre pour autant que certaines conditions soient remplies au niveau de la clé d'initialisation

Autres algorithmes par flot

- D'autres algorithmes de chiffrements par flot sont basés sur des *registres à décalages à rétroaction linéaire (LFSR)* dont la structure est efficace dans les implémentations matérielles et un peu moins dans le cas logiciel.



Chiffrement par flot (Conclusion)

Il existe deux types de chiffrement à clé symétrique :

- Le chiffrement par flots (ou par stream ou de flux) : l'opération de chiffrement s'opère sur chaque élément du texte clair (caractère, bits). On chiffre un bit/caractère à la fois. La structure d'un chiffrement par stream repose sur un générateur de clés qui produit une séquence de clés k1, k2, ..., ki
- **La sécurité du chiffrement dépend de la qualité du générateur**

Limites de chiffrement symétrique

- Chaque personne doit posséder **autant de clés secrètes** qu' elle a d' interlocuteurs.
- Un système de chiffrement symétrique pose donc **le problème de la gestion** et de la distribution des clés secrètes. Cet aspect constitue une **faiblesse des systèmes de chiffrement symétrique**.
- Pour pallier la **complexité du problème de la gestion et la distribution des clés** des systèmes de chiffrement symétrique qu'un autre type de système de chiffrement, qualifié de **chiffrement asymétrique** ou **chiffrement à clé publique** a été conçu et est actuellement largement utilisé dans le monde d' Internet.

Problème de l'échange de clés

Même avec un cryptosystème très sécuritaire, un problème qui reste. Il faut distribuer les clés secrètes qui seront utilisées sans qu'elles soient interceptées par des curieux.

Ces clés peuvent être échangées à l'aide d'un courrier diplomatique ou en temps de guerre, elles peuvent être distribuées aux unités avant leur départ.

Questions:

Qu'arrive-t-il si on manque de clefs?

Pas très pratique sur Internet!

Y a-t-il une solution?

Question de réflexion

Comment cryptanaliser le chiffrement de ce message (chiffrement symétrique)?

De Zanzibar à la Zambie et au Zaïre, des zones d'ozone font courir les zèbres en zigzags zinzins.

Question de réflexion

De Zanzibar à la Zambie et au Zaïre, des zones d'ozone font courir les zèbres en zigzags zinzins.

L'analyse fréquentielle n'est pas applicable,

Chapitre 5

Système de chiffrement asymétrique

.

Système de chiffrement asymétrique

Mode opératoire

- Un système de chiffrement asymétrique est basé sur l' utilisation d'un **couple unique de deux clés complémentaires**, calculées l'une par rapport à l'autre. Cette biclé est constituée d'une clé **publique** (connue par tous) et d'une clé **privée** (connue par son propriétaire). Seule la clé publique peut être connue de tous, tandis que la clé privée doit être confidentielle et traitée comme un secret.
- On doit connaitre la clé publique d'une destinataire pour lui envoyer des informations chiffrées. Ce dernier les décryptera à leur réception avec sa clé privée qu'il est le seul à connaitre . Le message est confidentiel pour le destinataire dans le mesure ou lui seul peut le déchiffrer par sa clé privée.

Système de chiffrement asymétrique

Mode opératoire

- De cette façon, pour transmettre des données de manière confidentielle avec un système de chiffrement asymétrique, l' émetteur chiffre un message avec la clé publique du destinataire du message et le destinataire le déchiffre avec sa clé privée.

Système de chiffrement asymétrique

Autre manière de définition:

La cryptographie à clé publique ou asymétrique est basée sur un concept très différent.

- Chaque intervenant possède une clé publique.
  - Cette clé peut être connue de tous. Par exemple, disponible dans un répertoire accessible publiquement.
  - Toute personne connaissant cette clé peut envoyer un message chiffré au propriétaire de cette clé.
  - Chaque intervenant possède une clé privée.
  - Cette clé doit demeurer confidentielle.
  - Cette clé est **liée (mathématiquement)** à la clé publique correspondante.
  - Cette clé permet de déchiffrer tout message chiffré avec la clé publique correspondante.

Système de chiffrement asymétrique(Schéma)

Emetteur

Le destinataire possède deux clés :

Clé publique + clé privée

Clé publique du destinataire

Texte en clair

Algorithme de chiffrement asymétrique

RSA, Diffie-Hellman, ElGamal, …

Texte chiffré

Réseau

Texte chiffré

Clé privée du destinataire

Algorithme de chiffrement asymétrique

Texte en clair

Destinataire

Système de chiffrement asymétrique (RSA)

Un exemple : RSA

- Développé par Rivest, Shamir & Adleman à MIT en 1977, publié en 1978
- Le plus connu et le plus utilisé comme algorithme de cryptage asymétrique.
- Breveté par RSA, et cette patente a expiré en 2000.
- Utilise des entiers très larges 1024+ bits
- Le fonctionnement du cryptosystème RSA est basé sur la difficulté de factoriser de grands entiers.

Le système RSA repose sur 4 concepts arithmétiques:

- Les nombres premiers
- Les nombres premiers entre eux
- La fonction modulo
- La décomposition binaire

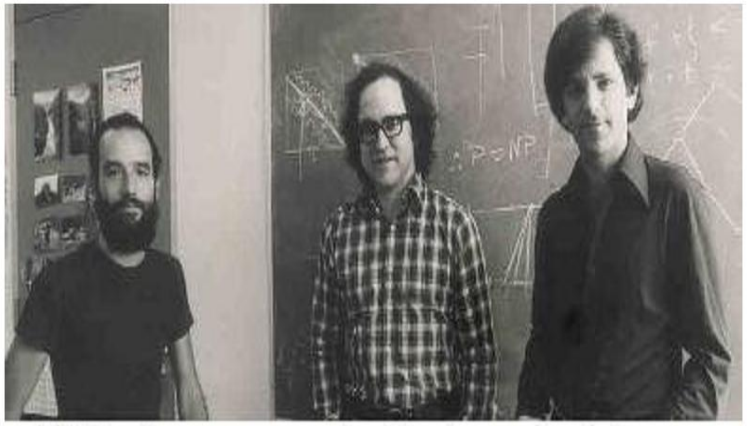
Exemples – Rivest : 1978

Le meilleur algorithme connu pour factoriser un nombre de 129 digits requiert:

- 40 000 trillions d'années soit $40*10^{15}$ années
- En supposant l'utilisation d'un superordinateur capable d'effectuer
  - 1 multiplication de nombres à 129 digits en 1 ns

Système de chiffrement asymétrique

Inventé en1977 par ==➔



Utilisation:

Serveurs Web, Internet (SSL/TLS), PGP, Cartes de crédit, Paiement électronique, Téléphones portables,

Microsoft, Apple Computer, Cisco Systems, Intel, Nokia, Sonny Ericson.

Système de chiffrement asymétrique (Rappels mathématiques)

Rappel

- ***A mod B*** est le reste de la division entière de A par B
- **La multiplication et le modulo**

$$(A \bmod B)(C \bmod B) = A * C \bmod B$$

- **L'exponentielle et le modulo**

$$a^n \bmod m = (a \bmod m)^n \bmod m$$

- Pierre **Fermat** :

Si on utilise un nombre premier comme module, alors quand on élève un nombre à la puissance (nombre premier -1), on obtient 1

  - Pour n'importe quel nombre ***m*** et pour ***p*** premier :

$$m^{(p-1)} \bmod p = 1$$

  - Exemple : $7^{10} \bmod 11 = 1$ …pas besoin de calcul car 11 est premier

$$x \equiv y(\mathrm{mod}\, n) \quad \text{ssi} \quad x = kn + y \quad \text{avec} \quad y < n$$

$$27 = 3*7 + 6 \quad \text{alors} \quad 27 \equiv 6(\mathrm{mod}\, 7)$$

$$x(\mathrm{mod}\, n) + y(\mathrm{mod}\, n) \equiv x + y(\mathrm{mod}\, n)$$

$$9 + 11 = 20 \equiv 6(\mathrm{mod}\, 7) \quad \text{et}$$

$$9(\mathrm{mod}\, 7) + 11(\mathrm{mod}\, 7) \equiv 2 + 4 \equiv 6(\mathrm{mod}\, 7)$$

$$x(\mathrm{mod}\, n) * y(\mathrm{mod}\, n) \equiv x * y(\mathrm{mod}\, n)$$

Exponentiation modulaire

Comment calculez $165789^{23456781} (\mathrm{mod}\, 456712)$

$x^8 = \left(\left(x^2\right)^2\right)^2$ donc pour calculer $x^{(2^k)} (\mathrm{mod}\ n)$

on calcule $x \leftarrow x^2 (\mathrm{mod}\, n)$ k fois

$21 = 10101 = 2^4 + 2^2 + 2^0$ et

$5^2 = 8 (\mathrm{mod}\ 17)$ $5^4 = 13 (\mathrm{mod}\ 17)$

$5^8 = 16 (\mathrm{mod}\ 17)$ $5^{16} = 1 (\mathrm{mod}\ 17)$

$5^{21} (\mathrm{mod} 17) = 5^{16} 5^4 5^1 (\mathrm{mod} 17) = 1 * 13 * 5 = 14 (\mathrm{mod} 17)$

PGCD

$PGCD(a,b) \qquad a > b$

$(a,b) \rightarrow (b, a \bmod b)$

si $b = 1$ alors répondre a

$PGCD(42,30) = 6$

$(42,30)$

$(30,12)$

$(12,6)$

$(6,1)$

$PGCD(105,45) = 5$

$(105,45)$

$(45,15)$

$(15,1)$

Inverse multiplicatif

$a^{-1} \bmod m$ avec $PGCD(a,m) = 1$

$(m, a, 1, 0)$

$(a, b, c, d) \rightarrow (b, a \bmod b, d - c(a \text{ div } b) \bmod m, c)$

si $b = 1$ alors répondre c

$5^{-1} \equiv 8 \bmod 13$

$(13, 5, 1, 0)$

$(5, 3, 0 - 1*(2) \bmod 13, 1) = (5, 3, 11, 1)$

$(3, 2, 1 - 11*(1) \bmod 13, 11) = (3, 2, 3, 11)$

$(2, 1, 11 - 3*(1) \bmod 13, 3) = (2, 1, 8, 3)$

$5*8 = 40 \equiv 1 \bmod 13$

$7^{-1} \equiv 19 \bmod 22$

$(22, 7, 1, 0)$

$(7, 1, 0 - 1*(3) \bmod 22, 1) = (7, 1, 19, 1)$

$7*19 = 133 = 6*22 + 1 \equiv 1 \bmod 22$

Avec un ordinateur, on calcule le PGCD et l'inverse multiplicatif de très grands nombres efficacement.

Système de chiffrement asymétrique (Rappels mathématiques)

- Leonhard **Euler** :
  - Lorsqu'on utilise un module comme étant le produit de deux nombres premiers on a :

    Soit n = p * q, avec p et q premiers, et quelque soit m

    $$m^{(p-1)(q-1)} \bmod n = 1$$

  - Exemple : soit p = 11 et q = 5, n = 55 et (p – 1)(q – 1) = 10 * 4 = 40

    $$38^{40} \bmod 55 = 1$$

  - Si on manipule le résultat d'Euler en multipliant par m l'équation :

    $$m * m^{(p-1)(q-1)} \bmod n = m$$

    $$m^{(p-1)(q-1)+1} \bmod n = m$$

Système de chiffrement asymétrique (RSA)

- **Chiffrement :**

L' émetteur récupère la clé publique (n,e) du destinataire et souhaite lui envoyer la version cryptée d'un texte en clair, représenté par la donnée d'un entier m tel que : $0 \leq m < n$. L' émetteur calcule :

$$c = m^e \bmod n$$

- **Déchiffrement :**

Lorsque le destinataire reçoit c , il calcule c^d et récupère ainsi le message m puisque :

$$m = c^d \bmod n$$

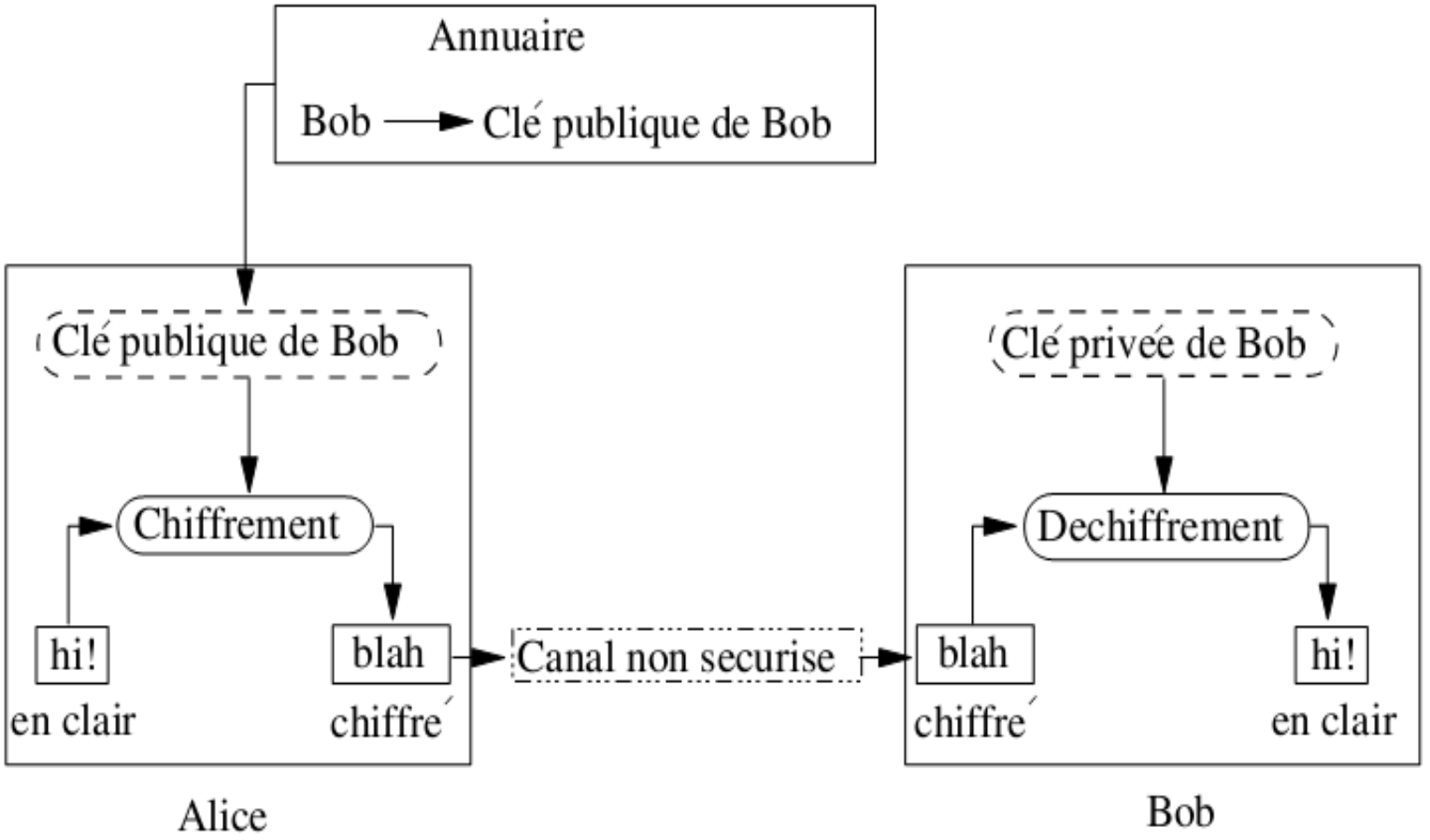
- **Notes :**

1) Ce qu'est en rouge sont des nombres privée du destinataire.

2) Quelle est la relation qui lie la clé privée et la clé secrète ?

3) Comment casser donc RSA?

Système de chiffrement asymétrique (RSA)



Système de chiffrement asymétrique (RSA)

Première étape : la création des clés de Bob pour RSA

Input Bob: rien

Output Bob: Clé publique (N, e); Clé privée d

begin

Choisir deux grands nombres premiers p et q

Calculer $N = pq$ et l'indicateur d'Euler $\phi(N) = (p-1)(q-1)$

Choisir un entier $1 < e < \phi(N)$ premier avec $\phi(N)$, c.à-d. $\text{pgcd}(e, \phi(N)) = 1$

Calculer d l'inverse de $e \pmod{\phi(N)}$

end

Désormais, toute personne ayant accès à la clé publique de Bob peut lui envoyer des messages sécurisés. A aucun moment, Alice n'est intervenue dans la création de la clé publique de Bob.

Système de chiffrement asymétrique (RSA)

Deuxième étape : Protocole RSA

Input general:La clé publique (N, e)
Input Bob: la clé privée d
Input Alice: un message x avec $x < N$
Output Bob: le message x est reçu
begin Cryptage
- Alice calcule $y \equiv x^e \pmod{N}$
- Alice envoie y à Bob

end Cryptage
begin Décryptage
- Bob calcule $x \equiv y^d \pmod{N}$

end Décryptage

Système de chiffrement asymétrique (RSA)

Emetteur

Clé publique du destinataire : (n,e)

Texte en clair : m

Algorithme de chiffrement asymétrique : RSA

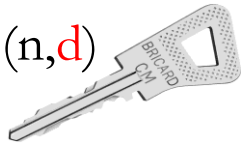
Texte chiffré : $c = m^e \bmod n$

Réseau

Texte chiffré : $c = m^e \bmod n$

Le destinataire possède deux clés :

(n,e) +



(n,d)

Clé privée du destinataire : (n,d)

Algorithme de chiffrement asymétrique : RSA

Texte en clair : $m = c^d \bmod n$

Destinataire

Système de chiffrement asymétrique (RSA)

Exemple numérique de RSA:

- Prenons deux nombres premiers aléatoirement : p = 29 , q = 37.

  On calcule n = pq = 29 * 37 = 1073 .
- On doit choisir e au hasard tel que e et (p -1)(q -1) = (29 -1)(37 -1) = 1008 soient premiers entre eux. On prend e = 71.
- On choisit d tel que 71*d mod 1008 = 1. On trouve d = 1079 .
- Voici alors les deux clés privée et publique :

$$\begin{cases} la\ clé\ publiqe\ est & (e\,,n)=(71,1073) \\ lé\ clé\ privée\ est & (d\,,n)=(1079,1073) \end{cases}$$

Système de chiffrement asymétrique (RSA)

- On va encrypter le message 'HELLO'. On va prendre d'abord le code ASCII de chaque caractère et on les met bout à bout :

m = 72(H) 69(E) 76(L)76(L) 79(O) .

Ensuite, il faut découper le message m en blocs qui comportent moins de chiffres que n . n comporte 4 chiffres, on va donc découper notre message m en blocs de 3 chiffres : 726 976 767 900 (on complète avec des zéros) par suite on encrypte chacun de ces blocs :

$$726^{71} \bmod 1073 = 436$$

$$976^{71} \bmod 1073 = 822$$

$$767^{71} \bmod 1073 = 825$$

$$900^{71} \bmod 1073 = 552$$

Système de chiffrement asymétrique (RSA)

- Le message encrypté est 436 822 825 552.

On peut reconstituer le message chiffré avec

d :

$$436^{1079} \bmod 1073 = 726$$

$$822^{1079} \bmod 1073 = 976$$

$$825^{1079} \bmod 1073 = 767$$

$$552^{1079} \bmod 1073 = 900$$

- C'est à dire la suite de chiffre 726 976 767 900.On retrouve notre message en clair 72 69 76 76 79 c'est à dire 'HELLO'.

Système de chiffrement asymétrique (exemple d'utilisation de RSA)

- Prenons **p**=47 et **q**=71
- **n** = 3337 et ϕ**(n)** = 46·70 = 3220
- Choisissons **e**=79 et **d**=1019
  (on peut vérifier que ed = 80501 = 1+ 25 ·3220)
- Pour chiffrer le message 688232687966668 on commence par le diviser en petit blocs de trois chiffres:

$$m_1\ m_2\ m_3\ m_4\ m_5\ m_6 = 688\ 232\ 687\ 966\ 668$$

- En utilisant la clef (e,n)=(79, 3337) on calcule le premier bloc:

$$688^{79} \bmod 3337 = 1570 = c_1$$

Système de chiffrement asymétrique (exemple d'utilisation de RSA)

- Les autres blocs sont chiffrés de la même manière et on obtient:

$$c_1\, c_2\, c_3\, c_4\, c_5\, c_6 = 1570\ 2756\ 2091\ 2276\ 2423$$

- Pour déchiffrer le message on utilise la clef (d,n)=(1019, 3337) et on obtient pour le premier bloc:

$$1570^{1019} \bmod 3337 = 668 = m_1$$

- Le reste du message en clair est obtenu de la même manière.

Système de chiffrement asymétrique (challenge de RSA)

Le challenge RSA :

- Après la découverte de l'algorithme RSA, la société RSA Security a été créée pour vendre le cryptosystèmes RSA. en 1991, **ils ont mis en place un concours de factorisation avec de l'argent à la clé pour montrer à quel point ils étaient sûrs de la sécurité de leurs algorithmes** (factoriser des nombres de 100 à 617 chiffres décimaux). En 2007, RSA Security a mis fin au concours.
- **RSA-768** est un nombre du concours RSA. Il est long de 768 bits (232 chiffres décimaux). **Il a été factorisé après 2 ans et demi de recherche et calculs**. Cinq grands organismes de recherches étaient associés (EPFL à Lausanne, NTT au Japon, l'Université de Bonn, CWI au Pays-Bas et LORIA à Nancy).

Système de chiffrement asymétrique (challenge de RSA)

Le challenge RSA :

Quelques chiffres pour donner une idée des calculs nécessaires :

- 5To de données traitées ;

- Calcul distribué sur plus de 1700 cœurs ;

- Environ 10^{20} opérations (soit 2000 ans sur un processeur simple cœur 2,2GHz) ;

- Notes : Pour avoir une sécurité optimale il ne faut plus utiliser des clés RSA inférieures à 1024 bits. Selon les chercheurs, RSA-1024 devrait résister pendant encore 5 à 10 ans (sauf découverte mathématique majeure).

Système de chiffrement asymétrique (Briser RSA)

La seule technique connue pour briser RSA consiste à calculer l'exposant de déchiffrement.
$d=e^{-1} mod\ (p-1)(q-1)$ où $pq=n$.
Pour ce faire, il faut factoriser n.

Par contre, comme l'algorithme de chiffrement est connu publiquement, si on devine le message, on peut vérifier facilement que c'est le bon.

Système de chiffrement asymétrique (avantages et inconvénients)

- Avantages
  - Pas besoin d'établir un canal sûr pour la transmission de la clé.
  - Plusieurs fonctions de sécurité: confidentialité, authentification, et non-répudiation
- Inconvénient
  - Généralement dix fois plus lent que le cryptage symétrique.
  - Problème d'implémentation sur les équipements disposants de faible puissance de calcul (ex: cartes bancaire, stations mobiles, etc.)
    - Clés longues
    - Complexité algorithmique de la méthode (ex: réalisation des opérations modulo n)

➔ Solution: Utilisation du cryptage asymétrique pour l'échange des clés secrètes de session d'un algorithme symétrique à clés privées.

Conseils d'utilisation du RSA

Pour garantir une bonne sécurité, il faut respecter certaines règles telles que :

– Ne jamais utiliser de valeur n trop petite,

– N'utiliser que des clés fortes (p-1 et q-1 ont un grand facteur premier),

– Ne pas chiffrer de blocs trop courts,

– Ne pas utiliser de n communs à plusieurs clés,

– Si (d,n) est compromise ne plus utiliser n.

Attaquer RSA

Il existe trois approches pour attaquer le RSA :

1– Recherche par force brute de la clé (impossible étant donné la taille des données)

2– Attaques mathématiques (basées sur la difficulté de calculer (n), la factorisation du module n) :

- factoriser n=p*q et par conséquent trouver (n) et puis d,
- déterminer (n) directement et trouver d,
- trouver d directement.

3– Attaques de synchronisation (sur le fonctionnement du déchiffrement).

RSA (conclusion)

La seule technique connue pour briser RSA consiste à calculer l'exposant de déchiffrement.
$d=e^{-1} mod\ (p-1)(q-1)$ où $pq=n$.
Pour ce faire, il faut factoriser n.

Méthode simple mais difficile à factoriser n

Un autre exemple : l'algorithme ElGamal

Si RSA est un des algorithmes à clé publique les plus populaires, il en existe quelques autres. L'un d'entre eux, **l'algorithme ElGamal**, a été développé par Taher ElGamal en 1984. Il n'est pas breveté et peut être **utilisé librement**.

Un autre exemple : l'algorithme ElGamal

- Ce cryptosystème repose sur la **difficulté** de la résolution **du logarithme discret.**

  (si $y = g^x \bmod p$, on dit que x est le logarithme discret de y de base g modulo p)

- Il présente l'avantage de faire appel à **deux processus aléatoires**, mais présente l'inconvénient de générer des messages chiffrés particulièrement **longs**.

- El Gamal est 2 fois plus lent que le RSA.
- L'inconvénient majeur reste la taille des données chiffrées qui représente 2 fois celle des données en clair

l'algorithme ElGamal

Le fonctionnement de l'algorithme d' ElGamal

- **Génération des clés :**

Alice (destinataire) choisit deux paramètres non-secrets : un nombre premier p et un nombre g inferieur à p. Elle choisit aléatoirement un nombre a dans l'intervalle [1, . . , p-2] et calcule $\alpha = g^a \bmod p$.

Elle publie sa clé (p, g, α) et garde secrète sa clé a.

Le fonctionnement de l'algorithme d' ElGamal

- **Chiffrement:**

Bob, qui désire envoyer un message m à Alice, l'exprime sous la forme d'un nombre entre 0 et p-1. Il choisit aléatoirement un nombre b dans l'intervalle [1, . . , p-2] et calcule $\beta = g^b \bmod p$.

Il chiffre alors son message m en $m' = \alpha^b.m \bmod p$.

Il transmet enfin à Alice le couple (β, m')

l'algorithme ElGamal

- **Déchiffrement :**

Pour déchiffrer le message de Bob, Alice détermine le nombre x = p-1-a.

Elle calcule $\beta^x.m' \bmod p$ et retrouve le message m initial (**à vérifier tout seul**).

en effet on :

$$\beta^x \cdot m' \equiv g^{bx} \cdot m' \equiv g^{b(p-1-a)} \cdot \alpha^b \cdot m \equiv g^{b(p-1-a)} \cdot g^{ab} \cdot m$$

$$\equiv g^{b(p-1)} \cdot g^{-ab} \cdot g^{ab} \cdot m \equiv m \bmod p$$

l'algorithme ElGamal

car on a d' après le petit théorème de Fermant

(Théorème : pour tout entier naturel non nul n et tout entier a premier avec n, on a:

$$g^{b(p-1)} \equiv 1 \bmod p$$

$$a^{\varphi(n)} \equiv 1 \bmod n$$

où φ(n) désigne la fonction φ d'Euler comptant les entiers entre 1 et n qui sont premiers avec n. Si n est un nombre premier, alors φ(n) = n - 1)

- Note importante : Casser cet algorithme c'est trouver a partir de l' équation :

$$\alpha = g^a \bmod p$$

l'algorithme ElGamal

Application numérique de l' algorihtme de ElGamal

- Alice : p = 97 et g = 13. Elle choisit aléatoirement un nombre a, disons 45, dans l'intervalle [1, . . , 95]. Elle calcule $\alpha = 13^{45} \bmod 97 = 20$. Elle publie sa clé (97, 13, 20) et garde secrète sa clé 45.

- Bob veut envoyer le message "vew" à Alice. En utilisant le code ASCII, son message est 118 101 119. Il le découpe en nombres entre 0 et 97 : 11 81 01 11 09. Il choisit aléatoirement un nombre b, disons 35, dans l'intervalle [1, . . , 95]. Il calcule $\beta = 13^{35} \bmod 97 = 71 \bmod 97$.

l'algorithme ElGamal

Il chiffre alors son message nombre par nombre :

11 est chiffré $20^{35} \cdot 11 \mod 97 = 46 \cdot 11 \mod 97 = 21$

81 est chiffré $20^{35} \cdot 81 \mod 97 = 46 \cdot 81 \mod 97 = 40$

01 est chiffré $20^{35} \cdot 1 \mod 97 = 46 \cdot 1 \mod 97 = 46$

09 est chiffré $20^{35} \cdot 9 \mod 97 = 46 \cdot 9 \mod 97 = 26$

Il transmet enfin à Alice le couple (71, 21 40 46 26)

- Pour déchiffrer le message de Bob, Alice détermine le nombre x = 97-1-45 = 51. Pour chaque partie m' du message, elle calcule $71^{51} \cdot m' \mod 97$:

$71^{51} \cdot 21 \mod 97 = 19 \cdot 21 \mod 97 = 11$

$71^{51} \cdot 40 \mod 97 = 19 \cdot 40 \mod 97 = 81$

$71^{51} \cdot 46 \mod 97 = 19 \cdot 46 \mod 97 = 11$

$71^{51} \cdot 26 \mod 97 = 19 \cdot 26 \mod 97 = 9.$

Elle retrouve le message 11810109 après concaténation (deux chiffres par deux chiffres).

Algorithme Elgamal

- **A noter:**
- Cet algorithme est utilisé par le logiciel libre GNU Privacy Guard, de récentes versions de PGP

- le même message est chiffré différemment à chaque essai
- le message chiffré est deux fois plus long que le message secret
- les calculs de chiffrement / déchiffrement sont complexes

Chiffrement asymétrique

- **Remarques:**

- La mise en œuvre du chiffrement asymétrique permet de vérifier l' origine d' un message et d' authentifier un émetteur (utilisation de signature et certificat).

- L'utilisation des algorithmes de chiffrement asymétrique est très utilisé pour réaliser des échange confidentiels et pour effectuer des signatures électroniques, notamment dans le domaine du commerce électronique et des transactions financières.

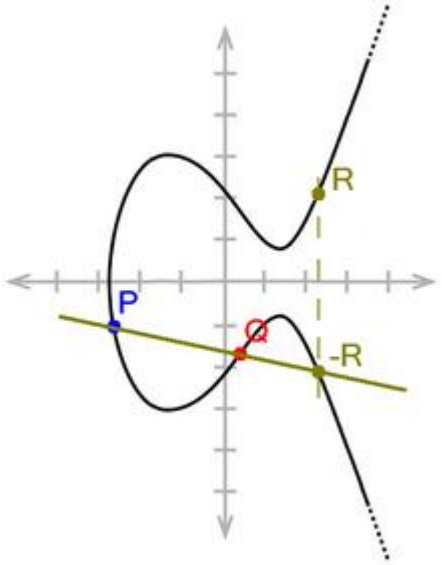
- Le temps d' exécution de ces algorithmes produit des temps de traitement processeur supplémentaire importants, rendant non performant le chiffrement de messages longs.

Chiffrement asymétrique(*Principaux algorithmes)*

- Les principaux algorithmes de chiffrement à clé publique, dont le nom est celui de leur inventeurs, exploitent le plus souvent des clés de longueur variant de 512 à 1024 bits voir 2048. Nous retiendrons les algorithmes importants :
- ***RSA*** (pour **R**on Rivest, Adi **S**hamir, Len **A**delman) qui est basé sur la difficulté de factorisation des nombre premiers.
- ***Diffie-Hellman*** et ***ElGamal*** qui sont basés sur le problème mathématique de calcul de logarithmes discrets.
- Certains algorithmes basés sur les équations de calcul de circonférences des ellipses sont à l'origine de la cryptographie à courbe elliptique (ECC, Elliptic Curve Cryptography), qui pourrait remplacer les algorithmes actuels.

Chiffrement asymétrique(*Principaux algorithmes)*

- En 2005, la NSA a officiellement annoncé la Suite B de ses recommandations cryptographiques, qui utilise exclusivement la cryptographie sur les courbes elliptiques pour l'échange de clé et les signatures numériques.



Additions de points sur une courbe elliptique

Avantages et inconvénients des algorithmes de Chiffrement

Symétrique Avantages:

- Rapidité (jusqu'à 1000 fois plus rapide)
- Facilité d'implantation sur hardware
- Taille de clé : 128 bits ( ) 16 caractères : mémorisable)

Asymétrique Avantages:

- Distributions des clés facilitées : pas d'authentification
- Permet de signer des messages facilement
- Nombre de clés à distribuer est réduit par rapport aux clés symétriques

Symétrique Inconvénients:

- Nombre de clés à gérer - Distribution des clés (authentification, confidentialité)
-Certaines propriétés (p.ex. signatures) sont difficiles à réaliser

Asymétrique inconvénients:

- Taille des clés
- Vitesse de chiffrement

Solution possible :

Combiner les deux systèmes

Chiffrement asymétrique(Cryptanalyse)

- La cryptanalyse des systèmes de chiffrement asymétrique s'appuie sur les mathématiques. Elle est basée sur la résolution ou la réduction de la complexité des fonctions inverses à celles utilisées par les systèmes de chiffrement symétrique.
- Donald Coppersmith (www.research.ibm.com) a élaboré en 1983 une méthode permettant de calculer les algorithmes discrets dans un temps polynomial (qui varie linéairement avec la longueur de la clé). Celle-ci est devenue un moyen incontournable pour les cryptanalyse de l'algorithme Diffie-Hellman.
- Des mathématiciens ont trouvé en 1980, un algorithme qui permettait de factoriser des nombres à 50 chiffres en 10^{12} opérations élémentaires.

Chiffrement asymétrique(Cryptanalyse)

- En 2005, une équipe chinoise a annoncée qu'elle avait réussi à réduire la complexité de 2^{80} à 2^{69} opération élémentaires et a démontré qu'elle a cassé l' algorithme SHA-1.

- Note : Même en l'absence de la diffusion d'une méthode prouvant qu'un algorithme a été casse, cela ne veut pas dire que la méthode n'existe pas ou que l'algorithme n' a pas été cassé. Aucune preuve mathématique ne permet d'affirmer que ce n'est pas possible.

➔ **C'est un problème qui restera ouvert**