TD Cryptographie

Jean-René Reinhard
jean-rene.reinhard@m4x.org

Mars 2016

Problème A : Utilisation de RC4 dans TLS

RC4, malgré ses nombreux défauts, est encore très utilisé de nos jours. Il est notamment mis en œuvre dans le protocole TLS, qui permet de chiffrer les données échangées dans une connexion TCP. Le protocole TLS fonctionne en deux étapes :

- dans un premier temps les interlocuteurs établissent de manière sécurisée des clés communes valides pour la durée de la session ;
- dans un deuxième temps ces clés sont utilisées pour protéger les communications, notamment les chiffrer.

Lorsque RC4 est mis en œuvre, il utilise des clés de 128 bits. Son initialisation ne fait pas appel à un vecteur d'initialisation. Pour chiffrer le flux de paquets TCP échangés, on utilise l'unique suite chiffrante produite à partir de la clé de session. Par exemple si le premier paquet compte 1500 octets, les 1500 premiers octets de la suite chiffrante sont utilisés pour le chiffrer. Si le deuxième paquet compte 100 octets, les 100 octets suivants de la suite chiffrante sont utilisés, ... Nous avons vu en cours que le WEP utilise un IV pour chiffrer chaque trame WiFi échangée.

1. Pourquoi la solution adoptée par TLS n'est pas envisageable pour le WiFi ? Autre formulation de cette même question : quelle est la propriété du protocole TCP qui permet de se passer d'IV dans le chiffrement ?

Réponse. TLS protège des données échangées à travers le protocole TCP. Celui-ci garanti la fiabilité des communications : les données arrivent dans l'ordre et sans perte. Lors de la réception d'un paquet TCP, on peut donc déterminer correctement sa position dans le flux de données échangées. Cette synchronisation étant assuré, on peut utiliser une seule suite chiffrante pour protéger la communication en confidentialité. Le Wifi, nom commercial du protocole 802.11, est un protocole de niveau 2 dans le modèle OSI (couche lien). À ce niveau, la fiabilité des communications n'est pas assurée. Il n'est donc pas possible de situer correctement dans le flux de données une trame reçue. C'est cette absence de synchronisation qui requiert l'utilisation une suite chiffrante différente pour chaque trame.

Des chercheurs ont récemment déterminé de manière expérimentale la distribution de probabilité des premiers octets de suite chiffrante de RC4, c'est à dire pour chaque position i de suite chiffrante plus petite qu'une certaine borne i_{limit}, pour chaque valeur d'octet z, déterminer la probabilité que $z_i = z$.

2. Écrire un algorithme permettant de déterminer de manière expérimentale la distribution du premier octet de suite chiffrante.

Réponse. **Entrée:** n, nombre d'échantillons
Initialiser un tableau N de 256 compteur à 0
Pour $i = 1; i < n; i++$ **Faire**
Tirer aléatoirement une clé de 128 bits
Initialiser RC4 à l'aide de cette clé et générer un octet de sortie z
Incrémenter $N[z]$
Fin Pour
Diviser toutes les valeurs dans le tableau N par n.
Renvoyer N

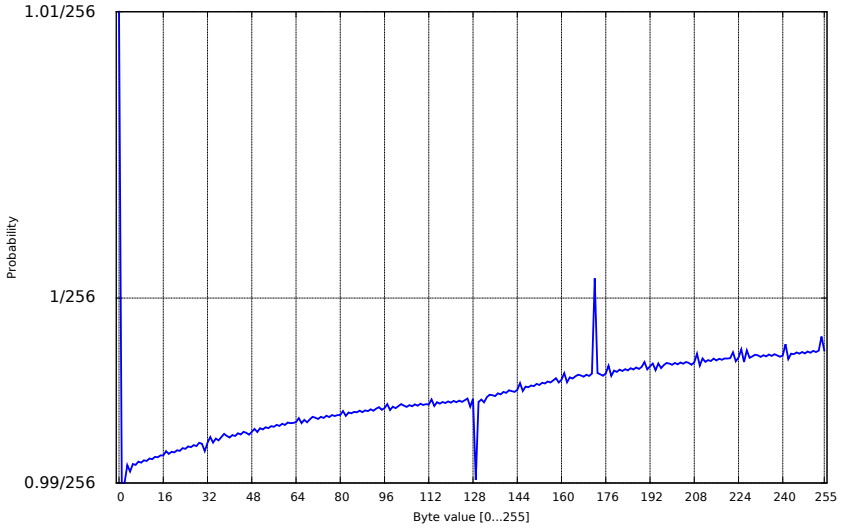


FIGURE 1 – Distribution de probabilité du deuxième octet de sortie de RC4

On donne en figure 1 la distribution de probabilité obtenue pour le deuxième octet de suite chiffrante : pour chaque valeur en abscisse, on lit en ordonnée la probabilité d'appartion de la valeur.

3. Quelle distribution attend-t-on pour un bon GPA ? Quel défaut observe-t-on ?

Réponse. Pour un bon GPA, les sorties doivent être indistingables des sorties d'un générateur d'aléa idéal. Pour un octet de sortie, on attend donc une distribution uniforme : chacune des 256 valeurs possibles doit apparaître avec probabilité égale, de valeur 1/256. La distribution observée est loin d'être uniforme. On constate, entre autre, que la valeur 0 apparait beaucoup plus souvent que les autres.

On peut exploiter les défauts indentifiés à la question précédente pour mettre en place une attaque dans un scénario multi-sessions. On suppose que l'attaquant peut obtenir le chiffré d'un même message clair sous D clés différentes. Ce scénario correspond à une réalité : les premiers messages de certains protocoles envoient de manière systématique un mot de passe.

4. Décrire une attaque permettant d'exploiter les D chiffrés d'un même message clair permettant de retrouver le deuxième octet du message clair.

Réponse. L'idée de l'attaque est de déterminer la distribution expérimentale des chiffrés, puis d'en déduire l'octet de clair le plus probable en fonction de cette distribution et des biais connus de la distribution de suite chiffrante. La première étape peut être

réalisée en réutilisant l'algorithme ci-dessus, en remplaçant le tirage aléatoire de clé et la génération d'un octet par l'utilisation directe d'un des D octets de chiffré. On cherche ensuite la valeur apparaissant le plus fréquemment. Comme l'octet 0 est l'octet le plus probable pour l'octet de suite chiffrante, la valeur identifiée est la valeur la plus probable de l'octet de clair recherché.

En pratique l'attaque décrite à la section précédente permet de retrouver le deuxième octet de clair avec de l'ordre de $D = 2^{20}$ chiffrés. Pourtant l'attaque reste difficilement applicable en pratique.

5. À votre avis, quel facteur limite l'impact pratique de l'attaque décrite ?

Réponse. L'accès à D chiffrés requiert l'établissement de D sessions TLS. Le temps nécessaire pour un établissement est relativement long dans la mesure où cette procédure met en œuvre des mécanismes de cryptagraphie asymétrique. De plus, il est également nécessaire pour obtenir une nouvelle session de faire tomber la session précédente ce qui introduit un délai supplémentaire. Ainsi, bien que 2^{20} soit un chiffre relativement modeste, il reste difficile d'obtenir cette quantité de chiffrés en pratique.

Problème B : Passeport électronique

Les passeports électroniques comportent en bas de la page d'identification deux bandes de caractères. Cette zone peut-être lue par un terminal de manière optique. La deuxième ligne comporte un numéro d'identification, appelé MRZ (Machine Readable Zone), qui fait office de clé pour protéger les communications électroniques entre le passeport et le terminal, dans le protocole BAC. L'objectif est d'assurer la confidentialité des échanges wireless entre un passeport et un terminal de lecture contre un attaquant ne pouvant pas un accès visuel sur le passeport. Outre des sommes de contrôle et du bourrage, cette MRZ est constituée de :

- un numéro d'identification composé de 9 caractères, lettre en majuscules ou chiffre, ou uniquement chiffre ;
- la date de naissance du porteur du passeport, au format YYMMJJ ;
- la date d'expiration du passeport, au format YYMMJJ ;
- le sexe du porteur, M, F ou < (non spécifié)

1. Quel est le nombre de valeurs possibles pour la MRZ d'un passeport ?

Réponse. On suppose que l'âge du porteur est compris entre 0 et 100 ans, que la durée de validité est de 10 ans, et que l'indication de sexe "non spécifiée" n'est pas utilisée. On a donc pour un identifiant alphanumérique

$$N = ((26 + 10)^9) \cdot 365,25 \cdot 100 \cdot 10 \times 365,25 \cdot 2 \approx 2^{73.52},$$

et pour des identifiants n'utilisant que des chiffres

$$N = ((10)^9) \cdot 365,25 \cdot 100 \cdot 10 \times 365,25 \cdot 2 \approx 2^{57.21},$$

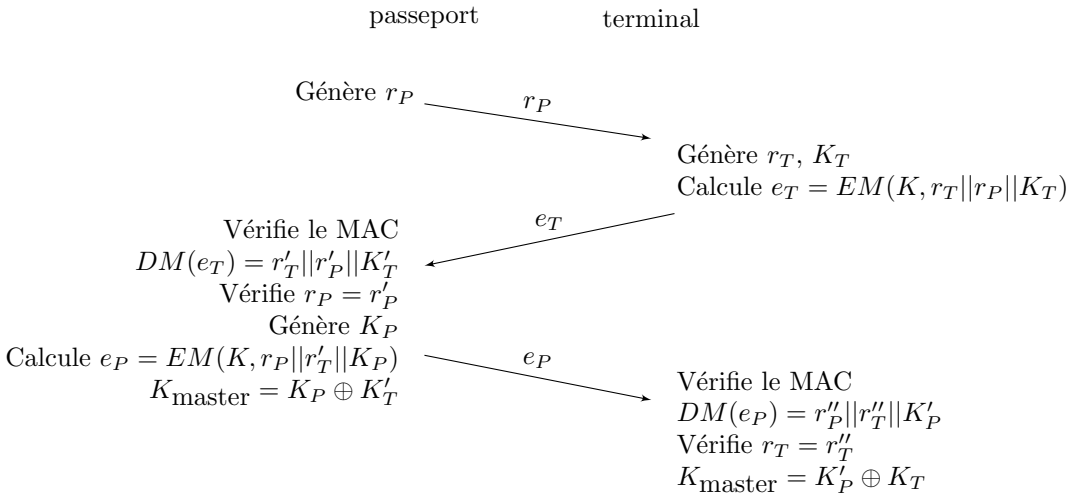
2. On observe une personne présentant un passeport à la douane, sans pouvoir lire la MRZ. Comment le nombre de MRZ possibles est-il réduit ?

Réponse. On peut à partir d'une identification visuelle déterminer de manière plus ou moins précise l'âge et le sexe du porteur. Le nombre de possibilité pour ces champs de la MRZ est réduit en conséquence.

Le protocole BAC, pour Basic Access Control, suit le déroulement suivant :
- Le passeport dispose d'une clé de chiffrement K_E et d'une clé de MAC K_M, on note $K = (K_E, K_M)$.
- Le terminal lit la MRZ et en déduit K.
- Les échanges font appel à une fonction de chiffrement intègre

$$EM(K, S) = E(K_E, S)||MAC(K_M, E(K_E, S)).$$

  La fonction de déchiffrement intègre correspondante est notée DM.
- Déroulement du protocole



3. Montrer qu'un attaquant passif, c'est à dire qui se contente d'écouter une exécution complète du protocole, peut en déduire un test d'arrêt, permettant de réaliser une recherche exhaustive sur K_E.

Réponse. Un attaquant passif qui écoute une exécution du protocole à accès aux données r_P, e_T et e_P. Pour tester si une clé candidate K^* est égale à la clé K utilisé par les participants, il peut réaliser les opérations suivantes :

Entrée: r_P, e_T, e_P
Sortie: Vrai ou Faux, suivant que la clé K^* est compatible avec l'échange
$r'_T||r'P||K'_T = DM(K^*, e_T)$
Si Le déchiffrement échoue ou $r'_P != r_P$ **Alors**
  Renvoyer Faux
Fin Si
$r''_P||r''_T||K''_P = DM(K^*, e_P)$
Si Le déchiffrement échoue ou $r''_P != r_P$ ou $r''_T != r'_T$ **Alors**
  Renvoyer Faux
Fin Si
Renvoyer Vrai

4. En considérant les deux premières questions, que peut-on dire de la protection en confidentialité apportée par le BAC ?

Réponse. Une écoute passive permet d'avoir accès à un test d'arrêt sur la clé. Le nombre de clés possibles étant faible ($< 2^{80}$), une recherche exhaustive est envisageable, et même pratique pour des identifiants qui n'utilise que des chiffres.

Problème C : Propriétés de RSA

RSA est un schéma asymétrique pouvant être utilisé aussi bien en chiffrement qu'en déchiffrement. La clé publique est constitué d'un module RSA N, produit de deux grands facteurs premiers p et q gardés secret, et d'un entier e appelé exposant public. La clé privée est constituée d'un exposant privée d, inverse de e modulo $\varphi(N) = (p-1)\cdot(q-1)$, i.e., $ed = 1 \mod \varphi(N)$.

1. Montrer qu'on peut déduire d de la clé publique et de la factorisation de N.

Réponse. L'algorithme d'Euclide étendu appliqué à e et $\varphi(N) = (p-1)(q-1)$, deux entiers premier entre eux, c'est à dire dont le PGCD vaut 1, permet de calculer d et k tels que $de + k\varphi(N) = 1$. On a $de = 1 \mod \varphi(N)$.

On s'intéresse désormais à la propriété réciproque. On suppose que l'adversaire connait d.

2. Montrer qu'il peut en déduire un multiple λ de $\varphi(N)$.

Réponse. $ed - 1 = k\varphi(N)$ est un multiple de $\varphi(N)$.

3. Justifier que λ est pair.

Réponse. p et q étant impairs, $\varphi(N)$ est divisible par 4, et λ également en temps que multiple de $\varphi(N)$.

Par la suite on note $\lambda = 2^r\gamma$, avec γ impair. On rappelle également que 1 possède 4 racines distinctes dans $\mathbb{Z}/N\mathbb{Z}$, qui correspondent, par le théorème des restes chinois, aux 4 paires suivantes de $\mathbb{Z}/p\mathbb{Z} \times \mathbb{Z}/q\mathbb{Z}$:

$$\begin{array}{rlrlcrl}
(& 1 & \mod p, & 1 & \mod q) & = & 1 \mod N \\
(& -1 & \mod p, & -1 & \mod q) & = & -1 \mod N \\
(& 1 & \mod p, & -1 & \mod q) & = & \alpha \mod N \\
(& -1 & \mod p, & 1 & \mod q) & = & -\alpha \mod N
\end{array}$$

4. Soit g un élément de $(\mathbb{Z}/N\mathbb{Z})^*$. Quelle est la valeur de g^λ ?

Réponse. Le théorème d'Euler énonce que pour N entier, $g \in \mathbb{Z}/N\mathbb{Z}$, on a

$$g^{\varphi(N)} = 1 \mod N.$$

En particulier, si $N = p$ premier, on obtient le théorème de Fermat

$$g^{p-1} = 1 \mod N.$$

$\lambda = k\varphi(N)$, donc

$$g^\lambda = (g^{\varphi(N)})^k = 1^k = 1 \mod N.$$

5. Justifier qu'on peut considérer $g^{\lambda/2}$.

Réponse. λ étant pair, sa moitié est un entier.

6. Quelles sont les valeurs possibles pour $g^{\lambda/2}$?

Réponse. $g^{\lambda/2}$ est une racine carré de 1 mod N. Il y a donc 4 valeurs possibles.

7. Supposons que $g^{\lambda/2} = \alpha$. Calculer $g^{\lambda/2} - 1$ modulo p et modulo q. En déduire la factorisation de N.

Réponse.

$$\begin{array}{rcll} g^{\lambda/2} - 1 & = & 0 & \mod p \\ g^{\lambda/2} - 1 & = & -2 & \mod q \end{array}$$

$g^{\lambda/2} - 1$ et N sont deux multiples de p, et q ne divise pas $g^{\lambda/2} - 1$, d'où $PGCD(g^{\lambda/2} - 1, N) = p$. On parvient donc à factoriser à partir de d, e et N, si on trouve g tel que $g^{\lambda} = \alpha$.

8. Que peut-on faire dans le cas où $g^{\lambda/2} = 1$? dans le cas où $g^{\lambda/2} = -1$?

Réponse. Si $g^{\lambda/2} = -1$, on ne peut pas factoriser, car $g^{\lambda/2} + 1$ est divisible par p et q simultanément. C'est également le cas pour $g^{\lambda/2} - 1$ dans le cas où $g^{\lambda/2} = 1$, mais dans ce cas, si $\lambda/2$ est pair, on peut continuer en considérant $g^{\lambda/4}$, etc.

9. Écrire un algorithme probabiliste permettant de factoriser N à partir de e et d.

Réponse. **Entrée:** $N, \lambda = ed - 1 = 2^r\gamma$, avec γ impair
Sortie: La factorisation de N
Tant que la factorisation n'est pas trouvé **Faire**
Tirer g au hasard dans $[1, N-1]$
Si $PGCD(g, N) > 1$ **Alors**
Renvoyer $PGCD(g, N), N/(PGCD(g, N)$
Sinon
$g \in (\mathbb{Z}/N\mathbb{Z})^*$
Fin Si
Pour $i = 1$ **to** r **Faire**
$h = g^{\lambda/2^i}$
Si $h = -1$ **Alors**
Go to Tirage de g
Sinon Si $PGCD(h-1, N) \notin \{1, N\}$ **Alors**
Renvoyer $PGCD(h-1, N), N/(PGCD(h-1, N)$
Fin Si
Fin Pour
Fin Tant que

On suppose que deux utilisateurs possèdent des bi-clés RSA partageant le même module : (N, e_1) et (N, e_2)

10. Que peut-on dire de la confidentialité des messages adressés au premier utilisateur ?

Réponse. Comme nous l'avons vu dans les questions 2 à 9, un utilisateur connait ou peut retrouver la factorisation de N à partir de sa clé privée. À l'aide de cette factorisation, il peut calculer à partir de l'exposant public de l'autre utilisateur l'autre clé privée. Chaque utilisateur est donc capable de déchiffrer les messages adressés à l'autre utilisateur.

11. Le même message m est chiffré à destination des deux utilisateurs. Montrer que le message m peut être récupéré à partir des chiffrés c_1 et c_2, sans utiliser les exposants privés (Indication : considérer une identité de Bezout).

Réponse. On calcule une relation de Bezout entre e_1 et e_2 à l'aide de l'algorithme d'Euclide étendu :

$$ue_1 + ve_2 = 1.$$

On peut alors calculer, à partir des chiffrés et des coefficients de Bezout

$$c_1{}^u c_2{}^v = m^{ue_1} m^{ve_2} = m^{ue_1+ve_2} = m^1 = m \mod N.$$

En pratique e est choisi petit.

12. Quel avantage y a-t-il à choisir e petit ?

Réponse. L'algorithme d'exponentiation rapide réalise une boucle dont le nombre d'itérations est le nombre de bits de l'exposant. Choisir un exposant petit permet d'accélérer ce calcul en limitant le nombre d'itérations réalisées.

13. Notons N_1, N_2 et N_3 les modules RSA de trois utilisateurs, utilisant pour exposant public $e = 3$. On chiffre un même message m à destination de ces trois destinataires, et on obtient les chiffrés c_1, c_2 et c_3. Montrer qu'un adversaire interceptant ces chiffrés peut retrouver m sans connaîtres les clés privées des destinataires.

Réponse. L'adversaire dispose de

$$\begin{cases} m^3 = c_1 \mod N_1, \\ m^3 = c_2 \mod N_2, \\ m^3 = c_3 \mod N_3. \end{cases}$$

Si N_1, N_2 et N_3, ne sont pas premiers entre eux, on obtient la factorisation d'un module par pgcd. S'ils sont premiers entre eux, on peut appliquer le théorème des restes chinois, et obtenir

$$m^3 = c \mod N_1 N_2 N_3,$$

pour $0 \leq c < N_1 N_2 N_3$ obtenu à partir des c_i. m étant plus petit que le plus petit des N_i, m^3 est plus petit que le produit des N_i et la dernière équation modulaire obtenu est donc vraie dans les entiers :

$$m^3 = c.$$

On peut alors retrouver m en calculant une racine cubique dans les entiers.

Par conséquent on choisit des valeurs de e supérieures, mais restant petites. typiquement, $e = 2^{16} + 1$.

14. e étant choisi petit, pourquoi d ne peut-il pas être également petit ?

Réponse. On a $ed - 1$ est un multiple de $\varphi(N)$. Cette grandeur étant un entier de taille égale à la taille de N, e et d ne peuvent pas être simultanément petit. Une autre raison pour laquelle d ne doit pas être petit est que cela facilite la cryptanalyse. Des attaques permettent de retrouver d si $d < N^{0.292}$.

Afin d'accélérer les calculs en déchiffrement (ou en signature), on considère la stratégie suivante : on réalise les calculs modulo p et modulo q, puis on utilise le théorème des restes chinois pour combiner les résultats.

15. Écrire les opérations réalisées pendant le déchiffrement

Réponse.

$$\begin{array}{rcl} c^d \mod p & = & c^{d \mod p-1} \mod p \\ c^d \mod q & = & c^{d \mod q-1} \mod q \end{array}$$

On note $d_p = d \mod p-1$ et $d_q = d \mod q-1$. La réduction modulo $p-1$ de l'exposant est possible car :
- soit c est nul modulo p. Comme $ed + k(p-1)(q-1) = 1$, $p-1$ ne divise pas d et $c^d = c^{d_p} = 0$.
- soit c est premier avec p, et dans ce cas $c^{p-1} = 1 \mod p$.

Le raisonnement pour le calcul modulo q est identique.

Une fois les calculs modulo p et q réalisés, on les combine à l'aide du thérorème des restes chinois :

$$c^d = \left(c^{d_p} q(q^{-1} \mod p) + c^{d_q} p(p^{-1} \mod q)\right) \mod N.$$

16. Quel est le gain en temps obtenu ?

Réponse. p et q sont de taille moitié par rapport à N et d_p et d_q sont de taille moitié par rapport à d. L'exponentiation modulaire a une complexité cubique, on gagne donc un facteur 8 pour une exponentiation. Comme on en réalise 2, on gagne un facteur 4 globalement. La combinaison finale influe peu le nombre final de multiplications.

17. On se place désormais dans le scénario d'une attaque par faute pendant une signature. On suppose que la faute produit une erreur uniquement pendant le calcul réalisé modulo p. Montrer qu'on peut obtenir la factorisation de N à partir de la signature erronée.

Réponse. On commence par réécrire l'exponentiation CRT en signature :

$$\begin{array}{rcl} m^d \mod p & = & m^{d \mod p-1} \mod p \\ m^d \mod q & = & m^{d \mod q-1} \mod q \\ m^d & = & \left(m^{d_p} q(q^{-1} \mod p) + m^{d_q} p(p^{-1} \mod q)\right) \mod N \end{array}$$

On note σ_p^* la valeur erronnée du calcul modulo p et σ^* la valeur erronnée de la signature. On a

$$\begin{array}{rcl} \sigma^{*e} \mod p & = & \sigma_p^{*e} \mod p \neq m \mod \text{p avec grande probabilité} \\ \sigma^{*e} \mod q & = & m^{ed_q} \mod q = m \mod q \text{ car } ed = 1 \mod q-1 \end{array}$$

On a donc $PGCD(\sigma^{*e} - m, N) = q$, ce qui donne la factorisation de N.

18. Proposer une contremesure contre cette attaque.

Réponse. Pour empêcher cette attaque, il ne faut pas fournir σ^* à l'attaquant. Suite au calcul de la signature, on vérifie que le résultat obtenu vérifie $\sigma^e = m \mod N$. Si ce n'est pas le cas, la génération de la signature détecte une faute et échoue, sinon la signature est renvoyée.

Problème D : Sécurité du $2DES$

L'algorithme de chiffrement DES souffre d'une taille de clé insuffisante (56 bits) et d'une taille de bloc un peu courte (64 bits). Afin de pouvoir utiliser des tailles de clés plus importantes, des constructions du DES en cascade ont été proposées. Elles consistent à composer plusieurs appels à l'algorithme de chiffrement par bloc DES, appels utilisant des clés différentes. La taille de bloc est inchangée.

$$\begin{aligned} 2DES_{K_1,K_2}(P) &= DES_{K_2}(DES_{K_1}(P)), \\ 3DES_{K_1,K_2}(P) &= DES_{K_1}(DES^{-1}_{K_2}(DES_{K_1}(P))), \\ 3DES_{K_1,K_2,K_3}(P) &= DES_{K_3}(DES^{-1}_{K_2}(DES_{K_1}(P))), \end{aligned}$$

1. Pourquoi une taille de bloc de 64 bits est-elle insuffisante en règle générale de nos jours ?

Réponse. Les modes opératoires avec lesquels les algorithmes de chiffrement par bloc sont mis en œuvre sont sûrs tant que moins de $2^{n/2}$ appels à l'alorithme de chiffrement par blocs sont effectuées, où n désigne la taille de bloc. Pour $n = 64$, cela correspond à de l'ordre d'un giga octet de données. Pour des applications haut débit, type chiffreur IP, cette quantité de donnée est rapidement atteinte. Une taille de bloc de 64 bits offre une marge confortable par rapport à cette contrainte.

2. Donner les tailles de clé pour chacune des 3 constructions présentées ci-dessus.

Réponse. $2DES$ et $3DES$ 2 clés ont une taille de clé de $2 \times 56 = 112$ bits. Le $3DES$ 3 clés a une taille de clé de $3 \times 56 = 168$ bits.

On s'intéresse à la sécurité de $2DES$. On suppose disposer de quelques paires (clair, chiffré). On considère en particulier (P, C). On construit la table T_1 obtenue en considérant l'ensemble des chiffrés de P sous toutes les clés K_1^* possibles : $T_1 = \{DES_{K_1^*}(P)\}$.

3. Que peut-on dire du déchiffré de C sous K_2 ?

Réponse. Il appartient forcément à la table T_1, il est égal au chiffré de P sous la clé K_1.

4. En déduire une attaque contre le $2DES$. On prendra soin d'éviter les fausses alarmes. Donner sa complexité en temps et en mémoire. Comparer à la recherche exhaustive naïve.

Réponse. Les attaques que nous étudions dans ce problème sont de type rencontre au milieu (Meet in the Middle). L'idée clé est de découper la clé en deux parties (disjointes) telles que la première partie du calcul ne dépende que d'une des deux sous-clés, la deuxième partie du calcul ne dépende que de l'autre sous-clé. On peut alors réaliser les deux parties du calcul de manière indépendante et ne considérer que les clés complètes formées de deux moitiés de clé correspondant à la même valeur au milieu du calcul.

Pour le $2DES$, cela donne :

Entrée: une paire clair-chiffré P, C (et quelques autres)
Sortie: la clé $K = K_1, K_2$
// Calcul de la table T_1
$T_1 = []$
Pour tous K_1^* **Faire**
Ajouter $DES_{K_1^*}(P), K_1^*$ à T_1
Fin Pour
Trier T_1 selon les chiffrés $DES_{K_1^*}$
Pour tous K_2^* **Faire**

Calculer $C^* = DES^{-1}_{K_2^*}(C)$
Pour tous K_1^* tel que $(C^*, K_1^*) \in T_1$ **Faire**
Tester K_1^*, K_2^* avec les autres clairs, chiffrés.
Si K trouvé **Alors**
Renvoyer K
Fin Si
Fin Pour
Fin Pour

Cet algorithme utilise une table de taille 2^{56} éléments. La première boucle requiert 2^{56} calcul du DES. La deuxième boucle requiert, à chaque itération un calcul de DES^{-1}, une recherche dans une table, et des tests additionnels de paires de clés K_1^*, K_2^*. La table étant triée, la recherche dans la table est rapide, on admettra qu'elle se fait en temps constant. Reste à borner le nombre de paires de clés considérées. Pour une clé K_2^*, le nombre de paires formées entre le déchiffré par K_2^* et le chiffré par K_1^* est 2^{56}. La probabilité d'égalité entre ces deux valeurs est 2^{-64}. On a donc en moyenne 2^{-8} paires par valeurs de K_2^*. Sur l'ensemble des valeurs de K_2^*, on a $2^{56}2^{-8} = 2^{48}$ paires formées. Ceci est conséquent, mais est bien inférieur aux 2^{112} paires à considérer dans une recherche exhaustive classique, et bien inférieur au coût des calculs de DES dans les deux boucles. La complexité en temps est donc de l'ordre de 2^{56} calculs de DES.

Remarques / Questions

N'hésitez pas à contacter l'auteur en cas de typos ou de questions.