RSA

Dario SALVADORE

Table des matières

Chapitre 1 RSA

1.1 Introduction

De nombreuses méthodes de cryptographie modernes fonctionnent grâce à l'utilisation des nombres premiers ; l'une d'elles, le système RSA, s'impose dans le monde des communications informatiques.

Le système RSA a été inventé par **Ron Rivest**, **Adi Shamir** et **Len Adleman** du *MIT*. Cette méthode fut présentée pour la première fois dans la chronique mathématique de Martin Gardner du numéro d'août 1977 du *Scientific American*.

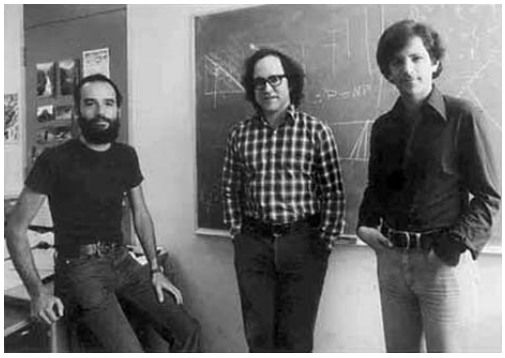

FIGURE 1.1 – Les auteurs : Adi Shamir, Ron Rivest et Len Adleman [1]

Ces trois auteurs avaient décidé de travailler ensemble afin d'établir que les systèmes cryptographiques "à clef publique", où la clef de codage est connue de tous, inventés quelques mois auparavant par W. Diffies et M. Hellman, présentaient des failles. Ils ne réussirent pas dans leur projet, mais, bien au contraire, découvrirent un nouveau système à clef publique qui supplanta celui de W. Diffies et M. Hellman.

Le principe arithmétique du système RSA est basé sur la difficulté de factoriser un nombre qui est le produit de deux grands nombres premiers.

1.2 Le système RSA

Le système RSA est utilisé à la fois pour crypter des messages (de façon à ce que seul le destinataire légitime puisse les lire) et pour les signer (de façon à ce que tous puissent s'assurer de leur provenance). Le qualificatif de système à clef publique, dans le cas du cryptage signifie précisément deux choses :

— on ne cache ni l'algorithme qui transforme la suite numérique du message originel en une autre suite numérique, ni la clef de codage qui sert à ce calcul (nommée de ce fait clef publique) ;
— tout le monde peut ainsi crypter des messages à destination de la personne qui a diffusé cette clef publique (nommons-la **Alice**) ; en revanche, seule **Alice** peut décrypter les messages qu'elle reçoit grâce à sa clef privée, qu'elle cache soigneusement. Les clefs publiques de tous ceux qui souhaitent recevoir des messages peuvent être publiées dans un annuaire, ou obtenues à la demande en contactant au préalable celui à qui on veut faire parvenir des messages codés.

Ces systèmes où la clef de décodage est différente de la clef de codage (nommés dissymétriques) présentent un avantage sur les systèmes classiques à une seule clef (nommés symétriques) : avant un échange, les deux interlocuteurs n'ont plus besoin de se rencontrer pour convenir secrètement d'une clef, ni de faire circuler cette clef secrète sur un réseau informatique ou autre, ce qui, bien sûr, n'est pas sans risques. Seule la clef publique, inutile pour le décryptage, circule préalablement aux communications cryptées entre les deux interlocuteurs.

Outre le cryptage, le système RSA à deux clefs sert aussi, nous l'avons vu, à signer des messages. On procède de façon inverse au cryptage : le signataire (nommons-la **Alice**) utilise sa clef privée pour transformer le message A en le message B, qui constitue sa signature. Le destinataire, ici **Bob**, qui reçoit le message (message A, message B), souhaite en contrôler la signature : il fait alors agir la clef publique d'**Alice** sur le message B. S'il retrouve le message A, c'est que ce message a bien été crypté par quelqu'un détenant la clef privée d'**Alice**, donc par **Alice** elle-même.

L'algorithme RSA est moins rapide que les algorithmes classiques à une seule clef ; aussi, lorsqu'on doit coder des messages volumineux, on l'utilise fréquemment en combinaison avec un algorithme classique. Dans un premier temps, l'expéditeur et le destinataire utilisent le RSA (avec ses deux clefs) pour convenir secrètement d'une troisième clef : l'expéditeur crypte cette troisième clef avec la clef publique d'**Alice**, laquelle, à l'arrivée, décrypte le message avec sa clef privée. Dans un second temps, l'expéditeur utilise la troisième clef avec un algorithme symétrique convenu pour envoyer rapidement à **Alice** le long message.

Le paragraphe suivant résume le principe de la cryptographie à double clef.

1.3 Principe de la cryptographie à double clef

Le principe de la cryptographie à double clef est exactement le même que celui de déposer une lettre dans une boîte aux lettres bien définie :

1. le cryptage d'un message avec la clef publique du destinataire (ici **Alice**) correspond à l'acte de glisser une lettre dans la boîte portant le nom du destinataire. Dans les deux cas, l'opération de l'expéditeur (**Bob**) est irréversible. Chaque clef publique, tout comme chaque boîte, est spécifique à un destinataire.
2. la destinataire (ici **Alice**) utilise ensuite sa clef privée pour décrypter le message codé par l'expéditeur avec sa clef publique. De même, elle utilise la clef de

sa boîte aux lettres, qu'elle est la seule à posséder, pour y prendre les messages que tous ont pu librement lui adresser.

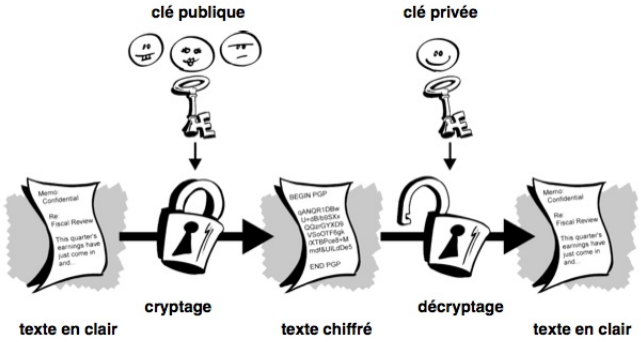

FIGURE 1.2 – Cryptage à clef publique [2]

Chapitre 2 Partie mathématique

2.1 Le petit théorème de Fermat

Le système RSA est fondé sur une série de résultats mathématiques. Le premier est le petit théorème de Fermat.

Théorème 2.1 (*Le petit théorème de Fermat*)

Soit p un nombre premier et a un entier naturel premier avec p. Alors $a^{p-1} - 1$ est divisible par p.

Démonstration : Considérons la suite des multiples de a :

$$a, 2a, 3a, ..., (p-1)a \tag{2.1}$$

Le nombre p n'en divise aucun.

Sinon p étant premier avec a, il devrait diviser l'un des nombres $1, 2, 3, ..., (p-1)$ qui lui sont inférieurs.

Les termes de la suite (2.1), divisés par p, donnent donc des restes non nuls.

Ces restes sont, en outre, distincts.

En effet, si deux multiples ka et $k'a$ donnaient le même reste, leur différence $ka - k'a = (k - k')a$ donnerait un reste nul.

Or, cette différence est un nombre de la suite (2.1) et, on a montré que les restes de la suite ne sont pas nuls. Donc les restes sont distincts.

Finalement, les $(p-1)$ restes des divisions par p, des nombres de la suite (2.1) sont non nuls et distincts.

Ces restes constituent donc, à l'ordre près, les nombres de la suite suivante

$$1, 2, 3, 4, ..., p-1 \tag{2.2}$$

Le produit des nombres de la suite (2.1), divisé par p, donne donc un reste égal au reste de la division par p, du produit des nombres de la suite (2.2).

Le produit des nombres de la suite (2.1) est :

$$a \cdot 2a \cdot 3a \cdot ... \cdot (p-1)a = 1 \cdot 2 \cdot 3...(p-1)a^{p-1} \tag{2.3}$$

Le produit des nombres de la suite (2.2) est :

$$1 \cdot 2 \cdot 3...(p-1) \tag{2.4}$$

On a donc

$$1 \cdot 2 \cdot 3 \cdot ... \cdot (p-1)a^{p-1} - 1 \cdot 2 \cdot 3...(p-1) \text{ est un multiple de } p \tag{2.5}$$

ou

$$1 \cdot 2 \cdot 3 \cdot ... \cdot (p-1)(a^{p-1} - 1) \text{ est un multiple de } p \tag{2.6}$$

Le nombre p est premier avec le produit :

$$1 \cdot 2 \cdot 3 \cdot ... \cdot (p-1) \tag{2.7}$$

Ainsi p divise le nombre $a^{p-1} - 1$. ■

Ce théorème peut s'énoncer aussi sous la forme du corollaire suivant.

Corollaire 2.2

Soit p un nombre premier et a un entier naturel premier avec p.
Alors $a^p \equiv a$ (modulo p).

Démonstration : D'après ce qui précède, si a et p sont premiers entre eux, $a^{p-1} - 1$ est congru à 0 modulo p. Sinon, p étant premier, a est congru à 0 modulo p. Dans les deux cas $a(a^{p-1} - 1) \equiv 0$ (modulo p) et par conséquent $a^p \equiv a$ (modulo p). ■

2.2 Le théorème de Bezout

Théorème 2.3 (*Identité de Bezout*)

Si d est le plus grand diviseur commun (noté PGDC) des nombres a et b, alors il existe deux nombres entiers u et v tels que $a \cdot u + b \cdot v = d$.

Démonstration : Soit I l'ensemble des entiers naturels de la forme $a \cdot n + b \cdot m$ où n et m sont deux entiers relatifs. Nous allons démontrer que le PGDC, noté D, fait partie de cet ensemble.

Il est clair I est un ensemble non vide. En effet 0, a et b en font partie. Comme tout sous-ensemble de $\mathbb{N}$, I admet un plus petit élément non nul. Appelons-le p.

Le nombre p appartenant à I, il existe deux entiers naturels u et v tels que : $p = a \cdot u + b \cdot v$

Comme tout élément de I, p est clairement divisible par les diviseurs communs à a et b. Il l'est donc en particulier pour le plus grand d'entre eux qu'est leur PGDC D.

Ainsi D divise p. Par conséquent $D \leqslant p$.

Il reste à prouver qu'il s'agit d'une égalité. La dernière étape consiste à démontrer que p est un diviseur commun à a et b.

Effectuons la division euclidienne a de p. Il existe un seul couple d'entiers naturels $(q; r)$ tels que :

$a = q \cdot p + r$ avec $0 \leqslant r < p$.

On peut alors écrire que :

$r = a - q \cdot p = a - q \cdot (a \cdot u + b \cdot v) = a \cdot (1 - q \cdot u) + b \cdot v$.

r est donc un élément de l'ensemble I qui est strictement plus petit que p.

Compte tenu de ce qu'est p, r est donc nécessairement égal à 0.

Le reste de la division euclidienne étant nul, p est donc un diviseur de a.

De la même façon, on montre que p est un diviseur de b.

En récapitulant, nous savons que p est un diviseur commun à a et b et aussi que p est supérieur ou égal à leur plus grand diviseur commun D. Donc les entiers naturels p et D sont nécessairement égaux.

Le PGDC de a et b est donc bien de la forme $a \cdot u + b \cdot u$. ■

Théorème 2.4 (*Théorème de Bezout*)

Les nombres a et b sont premiers entre eux si et seulement s'il existe deux nombres u et v tels que $a \cdot u + b \cdot v = 1$.

Démonstration : 1) Supposons que a et b soient premiers entre eux. Cela veut dire que leur PGDC est 1.

En application de l'identité de Bezout, il existe donc deux entiers u et v tels que $a \cdot u + b \cdot v = 1$.

2) a et b sont deux entiers. Supposons qu'il existe deux nombres u et v tels que $a \cdot u + b \cdot v = 1$.

Prenons c un diviseur commun à a et b.

Il est clair que c divise la somme $a \cdot u + b \cdot v$ c'est-à-dire 1.

Or quels sont les diviseurs de 1 ? Il n'y en a que deux : ce sont -1 et 1.

Cela signifie clairement que les entiers relatifs a et b sont premiers entre eux ! ■

2.3 Les propriétés justifiant le codage RSA

Proposition 2.5

Soit p et q deux nombres premiers. Si e, tel que $1 < e < (p-1)(q-1)$, est premier avec le produit $(p-1)(q-1)$ alors il existe un entier d unique tel que $1 < d < (p-1)(q-1)$ et vérifiant la propriété $ed \equiv 1 \,(\mathrm{mod}(p-1)(q-1))$.

Démonstration : Si e et $(p-1)(q-1)$ sont premiers entre eux, il existe d'après le théorème de Bezout deux entiers relatifs u et v tels que $u(p-1)(q-1) + ve = 1$.

Il est clair que si u' et v' vérifient la même égalité alors on a $(u'-u)(p-1)(q-1) = -(v'-v)e$. Il existe donc k entier tel que $u' = u + ke$ et $v' = v - k(p-1)(q-1)$. Soit donc k tel que u soit le plus grand des entiers négatifs, v étant alors le plus petit des entiers positifs.

Dans ces conditions :

$ve = 1 - u(p-1)(q-1)$ et par conséquent le nombre d recherché est égal à v.

Le nombre d est unique ; en effet s'il en existe un autre, appelons-le d', alors $e(d-d') \equiv 0$ (modulo $(p-1)(q-1)$). Comme e est premier avec $(p-1)(q-1)$ alors $d - d' \equiv 0$ (modulo $(p-1)(q-1)$). Mais comme on a $1 < d < (p-1)(q-1)$ et $1 < d' < (p-1)(q-1)$ et bien $d = d'$.

■

Proposition 2.6

Dans les conditions de la proposition précédente, si p et q sont différents et si $b \equiv a^e$ (modulo pq) alors on a $b^d \equiv a$ (modulo pq).

Démonstration : Si $b \equiv a^e$ (modulo pq), alors $b^d \equiv a^{de}$ (modulo pq).

Comme $ed \equiv 1$ (modulo $(p-1)(q-1)$), alors il existe un entier k tel que $ed = 1 + k(p-1)(q-1)$.

Dans ces conditions

$$\begin{aligned} a^{de} &= a^{1+k(p-1)(q-1)} \\ &= a^{1+kpq-kp-kq+k} \\ &= a^{kp(q-1)+1-k(q-1)} \\ &= (a^p)^{k(q-1)} \cdot a^{1-k(q-1)} \end{aligned}$$

D'après le petit théorème de Fermat on a $a^p \equiv a$ (modulo p) et par suite

$a^{de} \equiv a^{k(q-1)} \cdot a^{1-k(q-1)} \equiv a$ (modulo p).

On démontre de la même façon que $a^{de} \equiv a$ (modulo q).

Il existe donc deux entiers k et k' tels que $a^{de} = a + kp$ et $a^{de} = a + k'q$.

Ainsi $kp = k'q$ est un entier qui se trouve être multiple de pq, puisque p et q sont deux nombres premiers distincts.

Dans ces conditions, on a bien $a^{de} \equiv a$ (modulo pq). ■

Chapitre 3 Exemple

3.1 Protocole RSA

Bob veut envoyer le message M à **Alice**.

1. **Alice** choisit deux (grands) nombres premiers p et q et elle calcule $n = p \cdot q$ leur produit.
2. **Alice** choisit un (grand) nombre entier e premier avec $(p-1) \cdot (q-1)$.
3. **Alice** cherche un entier d tel que $d \cdot e - 1$ soit divisible par $(p-1) \cdot (q-1)$, c'est-à-dire que $e \cdot d \equiv 1$ (modulo $(p-1) \cdot (q-1)$).

   Le couple (e, n) représente sa clef publique. Tout le reste est secret et on peut oublier les nombres premiers p et q. Elle rend public le couple (e, n).
4. Le message M que **Bob** désire envoyer à **Alice** est un entier tel que $1 < M < n$.

   Si le nombre M est plus grand ou égal à n, **Bob** coupe le message en blocs de longueur adéquate. Ces blocs seront chiffrés et envoyés séparément.
5. Le message chiffré par **Bob** sera représenté par le nombre $C \equiv M^e$ (modulo n).
6. Pour déchiffrer C, **Alice** doit effectuer l'opération $C^d \equiv M$ (modulo n).

3.2 Exemple concret

Alice choisit deux nombres premiers $p = 31$ et $q = 23$. (Elle n'est vraiment pas très prudente !)

Alice calcule $n = p \cdot q = 713$. (En principe, si n est assez grand, il est difficile de retrouver p et q).

Elle choisit $e = 223$ (223 est premier avec $30 \cdot 22 = 660$).

--→ Alice a constitué sa **clef publique** : **(713 ; 223)**.

Elle détermine sa clef privée (qui ne sera jamais divulguée). Elle cherche l'entier $d > 0$ tel que $223 \cdot d - 1$ soit divisible par 660.

Elle trouve $d = 367$.

--→ Alice a constitué sa **clef privée** : **(713 ; 367)**.

Bob désire envoyer le message suivant à **Alice** :

SALUT

Bob transforme son message à l'aide de la règle :

A = 101, B = 102, ... , Z = 126

Son message découpé en blocs de 3 chiffres devient :

119 101 112 121 120

Il code chacune des parties (calculs effectués en `python` sous Mac) :

$119^{223} \equiv 708$ (modulo 713)

$101^{223} \equiv 016$ (modulo 713)

$112^{223} \equiv 479$ (modulo 713)

$121^{223} \equiv 193$ (modulo 713)

$120^{223} \equiv 401$ (modulo 713)

Remarquons par exemple que

$119^{223} =$

703027624786711158382570224401716199684246849583604672798310013690450567451
699107383362985529462833879084847294477126802378797959971455656559929523268401382349919471466108963242739896755588432449036885677181359974271334999497468024023024197808469675311166573001985469082096486307675399806959795192434853308596529432656773271448989524462500646259058027875531758735258658456774460390616129003733024979081054608608945031896179358010218961224640676653647612333747155 9

Le message codé :

708 016 479 193 401

Bob fait parvenir à **Alice** ce message par voie normale, sans le dissimuler.

Pour décoder, **Alice** utilise sa clef secrète :

$708^{367} \equiv 119$ (modulo 713) et

$16^{367} \equiv 101$ (modulo 713)

$479^{367} \equiv 112$ (modulo 713)

$193^{367} \equiv 121$ (modulo 713)

$401^{367} \equiv 120$ (modulo 713)

Remarquons également par exemple que

$708^{367} =$

916654283328891405458695114220142311748913280148059028795051641549579035652343664198678420442979226502139936480439746655648621247201835208977147011085247580760171958133442589514403468010988312469398954544195289872863412996244359941132260800187206958296921516467685829336002384427929679528467331401608658583023707767731228715348185243201496314576653689553316375330164829318007298602943347045830171367464751768897522001235356783878909747157146705983049021708050332262567038461734451820636122268634023820269719348592898162976520456781689326659182682429299509619258569873893767609261963025293882924493501840780194065872775400774262036579425101504357816246205109381521099867748211506223094315911436820114942104267013443364016895527871842206862334 5

185154726126426102606967035115830644320297049103758816079476511957779181824639747332503736364972265858703599658365800504965236118285125389831454500322282589805020244495219048346394882183583711448403125099610885071280489392096244159459491529573531009939883298070472266097675009999082855609666319779312435 2

Le message est décodé :

119 101 112 121 120

3.3 Programmes Java

3.3.1 Clé publique.

Voici un exemple de programme en Java qui permet de déterminer la clé publique, c'est-à-dire de calculer le nombre e (pour des valeurs de p et q pas trop grandes).

```
import java.io.*;

public class public_RSA {

    // Rechercher le premier entier e tel que
    // e et (p-1)(q-1) soient premiers entre eux.

    private static int public_key(int p, int q){

        int w = (p-1)*(q-1) ;
        boolean test = true ;
        int e = 1 ;

        while(test){

            int k = w % e ;

            if (k==0 ){
                test = true;
                e = e + 1;
                }else{
                test = false;
                }
            }
           return e ;
        }

        public static void main(String[] args){
            System.out.println("e = "+public_key(31,23));
        }
}
```

3.3.2 Clé privée.

Voici un exemple de programme en Java qui permet de déterminer la clé privée, c'est-à-dire de calculer le nombre d (pour des valeurs de p, q et e pas trop grandes).

```
import java.io.*;

public class private_RSA {

    // Rechercher le nombre d tel que ed=1 mod (p-1)(q-1).

    private static int private_key(int p, int q, int e){

        int w = (p-1)*(q-1) ;
        int i = 1 ;
        int k = 1 ;
        int t = 1 ;

        while(t != 0){

            i = (1 + k*w) ;
            k++ ;
            t = i % e ;

            }
            int d = i/e ;
            return d ; // est le nombre d
        }

       public static void main(String[] args){
           System.out.println("d = "+private_key(31,23,223));
        }
}
```

3.3.3 Exponentiation.

Voici un exemple de programme en Java qui permet de calculer le reste de la division de m^e (ou m^d) par n (pour des valeurs de n, e et d pas trop grandes).

```
import java.io.*;

public class expo {

    // Calcul du reste de la division de m^e par n.

    private static int message(int n, int e , int m){

        int j = 1 ;
```

```
        int k = 0 ;

        while(k < e){

            j = j * m ;
            j = j % n ;
            k++ ;

            }

            return j ;
        }

       public static void main(String[] args){
           System.out.println("j = "+message(713,223,119));
        }
}
```

3.4 Sécurité du système RSA

Il est préférable de choisir par exemple des grands nombres premiers (ici à 80 chiffres) :

$p = 70549149250650501590604446082178899520826292283748477553612223961604412362735751$

$q = 17628641385523085957347651727687378494339654589028422862604223955431405426681227$

On choisit $e = 17$ et on trouve

$d = 65842181586712717400487004384776076519104287176069212358824816460355907053223850$

$80024892221029288802399747180134665824959735089596423473951566906784452186038 53$

La théorie actuelle de la factorisation des grands entiers est extrêmement élaborée. Voici un historique grossier des limites des méthodes de factorisation depuis 1976 :

en 1976 : la limite est fixé à 10^{70}

en 2000 : la limite est fixé à 10^{100}

en 2006 : la limite est fixé à 10^{160}

Aujourd'hui il est conseillé de prendre p et $q \geq 10^{100}$ donc $n \geq 10^{200}$.

3.4.1 L'attaque de l'homme du milieu [3]

Cette attaque a pour but de s'insérer entre deux ordinateurs qui communiquent. Soit deux personnes Alice et Bob voulant dialoguer. Maintenant, si un pirate décide de se faire passer pour Alice auprès de Bob et de Bob auprès de Alice, ainsi, toute communication vers Alice ou Bob passera par le pirate, l'homme du milieu.

Cas normal

Alice et Bob veulent échanger des données confidentielles, et Charles veut les intercepter. Ils possèdent chacun une clef privée (resp. As, B_s et C_s) et une clef publique (resp. A_p, B_p et C_p).

1. Alice et Bob échangent leur clef publique. Charles peut les lire, il connaît donc A_p et B_p.
2. Si Alice veut envoyer un message à Bob, elle chiffre ce message avec B_p. Bob le déchiffre avec B_s.
3. Charles, qui ne possède que B_p, ne peut pas lire le message.

Attaque

Admettons maintenant que Charles soit en mesure de modifier les échanges entre Alice et Bob.

1. Bob envoie sa clef publique à Alice. Charles l'intercepte, et renvoie à Alice sa propre clef publique (C_p) en se faisant passer pour Bob.
2. Lorsque Alice veut envoyer un message à Bob, elle utilise donc, sans le savoir, la clef de Charles.
3. Alice chiffre le message avec la clef publique de Charles et l'envoie à celui qu'elle croit être Bob.
4. Charles intercepte le message, le déchiffre avec sa clef privée (C_s) et peut lire le message.
5. Puis il chiffre à nouveau le message avec la clef publique de Bob (B_p), après l'avoir éventuellement modifié.
6. Bob déchiffre son message avec sa clef privée, et ne se doute de rien puisque cela fonctionne.
7. Ainsi, Alice et Bob sont chacun persuadés d'utiliser la clef de l'autre, alors qu'ils utilisent en réalité tous les deux la clef de Charles.

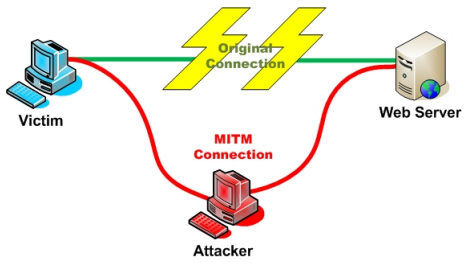

FIGURE 3.1 – L'attaque de l'homme du milieu [4]

Annexe A Références

[1] http://wiki.cs.miami.edu/pages/_media/home/burt/adleman_r_s.jpeg

[2] http://www.pgpi.org/doc/guide/6.5/fr/intro/

[3] http://fr.wikipedia.org/wiki/Attaque_de_l'homme_du_milieu

[4] http://wiki.h4ckers.fr/index.php?title=Man_In_The_Middle

[5] Merveilleux nombres premiers, Jean-Paul Delahaye, Belin

[6] Initiation à la cryptographie, Gilles Dubertret, Vuibert

[7] Réseaux, Andrew Tanenbaum, Pearson