REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
Ministère de l'enseignement supérieur et de la recherche scientifique
Université Mouloud Mammeri de Tizi - Ouzou
Faculté du génie électrique et
De l'informatique
Département d'électronique

Mémoire de fin d'études

En vue d'obtention du diplôme Master II en électronique
Option: réseau et télécommunication

Thème

Sécurité des données par la cryptographie quantique

Dirigé par:
M^r^ ZIANI. R.

présenteé par:
M^elle^ STITI Karima.
M^elle^ BELLOUNI Souad.

Devant le Jury:

Président : M^r^ AMEUR.S
Examinateurs : Mr LAHDIR.M
M^r^ OUALLOUCHE .F

Promotion 2010/2011

REMERCIEMENTS

Nos vifs remerciements vont à notre promoteur Mr R.ZIANI
Docteur –Ingénieur à l'université Mouloud Mammeri de Tizi-Ouzou pour nous avoir confié ce travail et pour le suivi et l'intérêt qu'il nous a manifeste.
Nos remerciements s'adressent aussi à Mr H .Taleb qui nous a aidés.

Nos remerciements s'adressent aussi à Mr M.LAHDIR qui nous a aidés.

Nos remercions également tous les membres de jury qui nous font honneur d'évaluer notre travail.

L'expression de notre gratitude s'adresse aussi à tous nos enseignants des différents cycles d'étude.

Que toutes les personnes, qui de prés ou de loin ont contribué à l'aboutissement de notre travail, trouvent ici l'expression de notre sincère reconnaissance.

Dédicaces

Je dédie ce travail.

A mon cher père Mokrane
Et ma chère mère Fatma

Pour l'éducation et le grand amour dont ils m'ont entouré depuis ma naissance.
Et pour leurs patiences et leurs sacrifices.

A mes chères sœurs;
A mes chères Frères ;
A tous mes proches;
A tous ceux qui m'aiment;
A tous mes amis(es) ;
A tous ceux que j'aime.

KARIMA

Dédicaces

Je dédie ce travail a :

- Mes chers parents
- Mes chers frères Mohamed, Djamel
- Mon oncle, sa femme et mes chers cousins
- Toute ma famille
- Toutes mes amies et a Tous mes amis
- Toute la promotion 2010/2011
- Ma binôme et toute sa famille
- Ma grande mère et mon grand père
- La mémoire des deux autres
- Notre promoteur

Souad

Sommaire

Introduction générale

Le besoin d'assurer la sécurité des communications entre les personnes s'est toujours fait sentir. La confidentialité de l'information et des échanges repose sur des mécanismes visant à dissimuler l'information aux yeux de tous ceux qui ne doivent pas en avoir connaissance. La technique de chiffrement de ces données est le moyen le plus utilisé.

En effet, la cryptographie est l'étude des méthodes permettant de transmettre des données de manière confidentielle, elle sert non seulement à préserver la confidentialité des données, mais aussi à garantir leur intégrité et leur authenticité.

Initialement, les méthodes de chiffrement reposaient sur deux opérations, les substitutions qui consistent à remplacer un caractère par un autre et les transpositions ou confusions qui mélangent l'ordre des caractères. Ensuite, il y eu le chiffrement moderne constitué de deux familles de systèmes, les systèmes symétriques ou à clé secrète et les systèmes asymétriques ou à clé publique. Les premiers utilisent une seule clé pour le cryptage et pour le décryptage. Cette clé commune partagée entre l'émetteur et le récepteur doit être tenue secrète. Les deuxièmes fonctionnent avec le principe d'une paire de clés, l'une est publique, l'autre est privée. La clé publique accessible à tout le monde permet de crypter les messages. La clé privée, qui elle doit être tenue secrète, sert à décrypter le message chiffré. Sur la base de ces deux techniques, une multitude d'algorithmes de cryptage plus ou moins complexes ont été développés.

La dernière nouveauté en matière de cryptologie est la cryptographie quantique. Si jusqu'à présent, les différents algorithmes présentés reposent en grande partie sur les mathématiques, Cette nouvelle classe d'algorithmes est basée sur les principes de la physique quantique.

Nous nous sommes intéressés dans notre travail à cette nouvelle technique de cryptage et plus particulièrement au protocole BB84.

Le mémoire est organisé en quatre chapitres :
Le premier chapitre présente des notions générales sur la cryptographie.
Le deuxième chapitre est consacré à une description des algorithmes de cryptage modernes.
Le troisième chapitre porte sur la cryptographie quantique.
Dans le quatrième chapitre nous présentons une simulation du protocole BB84.

Chapitre I

Notions générales sur la cryptographie

Introduction

L'évolution de la société s'appuie sur des techniques de plus en plus tournées vers la communication et les réseaux sont les vecteurs principaux de cette révolution technologique. Néanmoins dans beaucoup de cas, la sécurité des informations véhiculées doit être garantie.

En effet, ces systèmes de transmission de données se prêtent à des menaces de types divers, susceptibles d'altérer ou de détruire l'information (intégrité), de la révéler à des tiers qui ne doivent pas en avoir connaissance (confidentialité), ou encore de porter atteinte à sa disponibilité.

La sécurité dans les réseaux de communication est basée sur un ensemble de services destinés à garantir l'intégrité et la confidentialité des messages transmis, à fournir l'authentification des utilisateurs et le non-déni de service.

Aussi, avant de concevoir un système de sécurité, il est nécessaire d'identifier les menaces contre lesquelles le réseau doit être protégé.

I. Les menaces

Une menace est une violation potentielle de la sécurité. On peut les classer en deux catégories selon qu'elles ne modifient pas les informations transmises (menaces *passives*) ou qu'elles perturbent le fonctionnement du réseau (menaces *actives*).

- **Les menaces passives :** elles consistent à copier ou à écouter l'information sur le réseau. Ce type de menaces nuit à la confidentialité des données. Dans ce cas, celui qui prélève une copie n'altère pas l'information elle-même. Il en résulte des difficultés à détecter ce type de malveillance, car il ne modifie pas l'état du réseau.

- **Les menaces actives** : elles se traduisent par différents types d'attaques. On distingue le brouillage (émission de bruit sur les canaux de transmission altérant la compréhension des messages), le déguisement (modification des données au cours de leur transmission ou modification de l'identité de l'émetteur ou du destinataire), l'interposition (création malveillante de messages en émission ou en réception). Ce type de menaces nuit à l'intégrité des données.

II. Les objectifs de la sécurité

Les services de sécurité à mettre en œuvre pour assurer la sécurité des communications sont :

- L'**intégrité**, qui garantit que les données sont bien celles que l'on croit être, c'est à dire n'ont pas été altérées durant la communication.
- La **confidentialité**, qui consiste à assurer que seules les personnes autorisées aient accès aux ressources échangées. A cet effet, l'information est rendue inintelligible à d'autres personnes que les seuls acteurs de la transaction.
- La **disponibilité**, qui permet de maintenir le bon fonctionnement du système d'information.

- La **non répudiation**, qui permet de garantir qu'une transaction ne peut être niée.
- L'**authentification**, qui consiste à assurer l'identité d'un utilisateur, c'est à dire garantir à chacun des correspondants que son partenaire est bien celui qu'il croît être. que seules les personnes autorisées aient accès aux ressources. Un contrôle d'accès peut permettre par exemple par utilisation d'un mot de passe crypté l'accès à des ressources uniquement aux personnes autorisées.

III. Les mesures de protection

Afin d'éliminer les vulnérabilités et garantir un niveau élevé de protection du réseau et du système d'information, on peut utiliser des services, des mécanismes, des outils et des procédures que l'on nomme, de façon générale des mesures de protection ou de sécurité.

Parmi les mesures de protection on trouve : la protection physique, la protection de l'exploitation, la protection logique.

III.1 sécurité physique

La sécurité physique se rapporte à la sécurité des infrastructures matérielles. Elle concerne tous les aspects liés à la maitrise des systèmes et de l'environnement dans lesquels ils se situent. Nous retiendrons que la sécurité physique repose essentiellement sur :

- la protection des sources énergétiques (alimentation, etc.) ;
- la protection de l'environnement (incendie, température, humidité, etc.) ;
- la sûreté de fonctionnement et la fiabilité des matériels (composants, câbles, etc.) ;
- le marquage des matériels ;

III.2 sécurité de l'exploitation

La sécurité de l'exploitation désigne tout ce qui touche au bon fonctionnement des systèmes. Cela comprend la mise en place d'outils et de procédures relatifs aux méthodologies d'exploitation, de test, de diagnostic et de mise à jour.

Les points clés de la sécurité de l'exploitation sont :

- le plan de sauvegarde ;
- la gestion des configurations et des mises à jour ;
- la gestion des contrats de maintenance ;

III.3 sécurité logique

Elle correspond à une protection logique des données par des algorithmes de cryptage. La sécurité logique fait référence à la réalisation de mécanismes de sécurité par logiciel .elle repose principalement sur une mise en œuvre adéquate d'un processus de contrôle d'accès logique lequel s'appuie sur un triple service d'identification, d'authentification et d'autorisation.

IV. la cryptographie

IV .1. Définition

La cryptographie vient du grec *kruptos* qui veut dire caché et *graphein* qui signifie écrire. Par définition, la cryptographie est donc un ensemble de techniques permettant de transformer les données, dans le but de cacher leur contenu, empêcher leur modification ou leur utilisation illégale. Un système cryptographique (figure I.1) permet de transformer un message intelligible ou message en clair M en un message chiffré incompréhensible C que l'on appelle également cryptogramme. Ce procédé désigné sous le terme de chiffrement ou cryptage utilise un algorithme et une clé de chiffrement Kc. Le récepteur procédera au déchiffrement du message crypté en appliquant la transformation inverse avec une clé Kd. On a alors :

$C = E_{kc}(M)$ et $M = D_{Kd}(C) = D_{Kd}[E_{kc}(M)]$

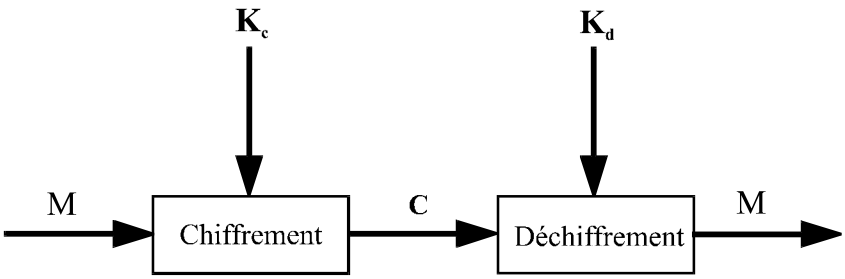


Figure I.1 : Système cryptographique.

La cryptologie est la science qui traite la façon de modifier ou de masquer une information afin de la rendre incompréhensible aux yeux des autres [Bec 90] ; à l'exception de celui qui connait le mode d'emploi et/ou le secret qui lui rendra son aspect initial.

Alors que la cryptographie consiste à sécuriser les données [Sch 95], La cryptanalyse représente les moyens et les méthodes utilisés pour découvrir le sens caché des informations malgré le soin apporté pour en garder la confidentialité par le chiffrement.

IV .2 Les types de chiffrement

Fondamentalement, il existe deux méthodes de chiffrement, les substitutions qui consistent à remplacer un caractère par un autre et les transpositions ou confusions qui mélangent l'ordre des caractères. Sur la base de ces deux techniques, une multitude d'algorithmes de cryptage plus ou moins complexes ont été développés.

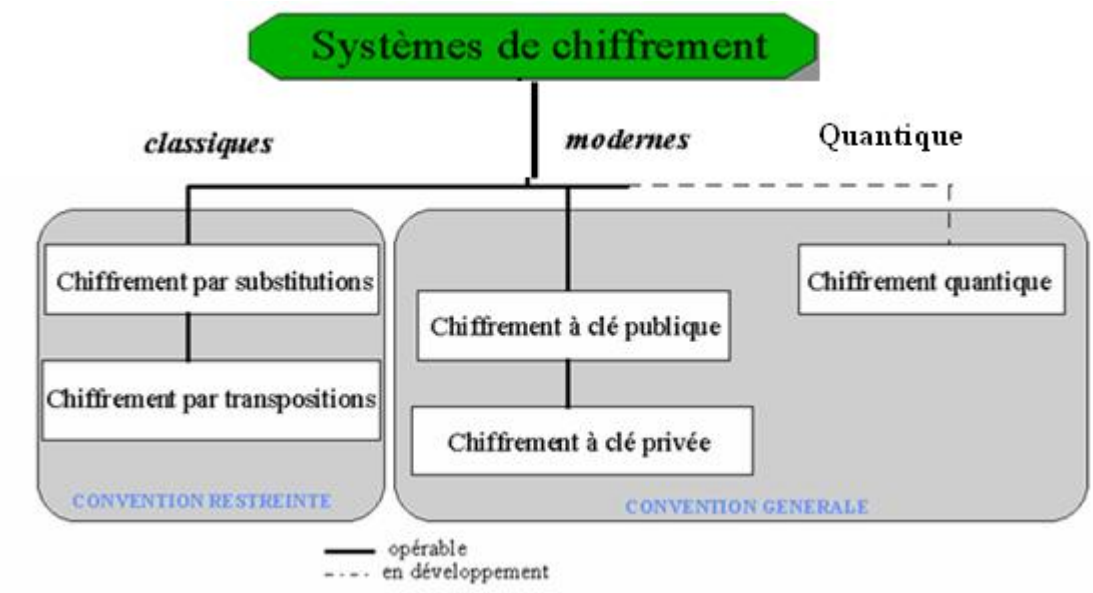


Figure I.2 : principales techniques de chiffrement

IV .2.1 Chiffrement classique

a. Substitution :

On distingue plusieurs types de cryptosystèmes par substitution :

- **substitution monoalphabétique** : chaque caractère du message en clair provenant d'un alphabet A est remplacé par le caractère correspondant appartenant à un alphabet de substitution S.

On prend comme exemple le chiffrement de César où on décale les lettres de 3 positions.

Alphabet A : A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Alphabet S: D E F G H I J K L M N O P Q R S T U V W X Y Z A B C

Le chiffrement du message M : **master deux**

Donne le message chiffré C : ***pdvwhu ghxa***

- **Substitution polyalphabétique :** consiste à utiliser des substitutions multiples périodiques

Par exemple le Code de Vigenère (présenté pour la première fois au courant du 16^{éme} siècle) utilise 26 alphabets décalés. C'est le premier algorithme à introduire la notion de clé.

La clé définit le décalage pour chaque lettre du message en clair. Son avantage est que la même lettre du message peut être codée de façon différente.

Texte clair	C	H	I	F	F	R	E	D	E	V	I	G	E	N	E	R	E
Clé	I	U	P	M	I	A	G	E	I	U	P	M	I	A	G	E	I
Décalage	8	20	15	12	8	0	6	4	8	20	15	12	8	0	6	4	8
Texte crypté	K	B	X	R	N	R	K	H	M	P	X	S	M	N	K	V	M

Ø Substitution par polygrammes :

Les types de substitutions précédents opèrent sur des caractères individuels. Le chiffrement polygrammique opère sur des blocs de caractères, ce qui rend la cryptanalyse plus difficile en éliminant la signification des fréquences des caractères pris individuellement.
Comme exemple, on citera le chiffrement de Playfair (créé en 1854 et utilisé par les Anglais durant la première guerre mondiale).La clé est constituée d'une matrice 5 X 5 et chaque paire de caractères m_1 m_2 du message est chiffré selon les 3 règles suivantes, basées sur la position de m_1 et m_2 dans la matrice.

m1	**C1**	**m2**	**C2**

Règle 1

m1			**C1**
C2			**m2**

Règle 2

	m1	
	C1	
	m2	
	C2	

Règle 3

Exemple :
Soit la matrice suivante construite à l'aide du mot-clé « CESAR » :

C	E	S	A	R
B	D	F	G	H
I/J	K	L	M	N
O	P	Q	T	U
V	W	X	Y	Z

Le chiffrement du message M : **CO NF ID EN TI EL**
Donne le message chiffré C : **BV LH KB RK OM SK**

b. Transposition :

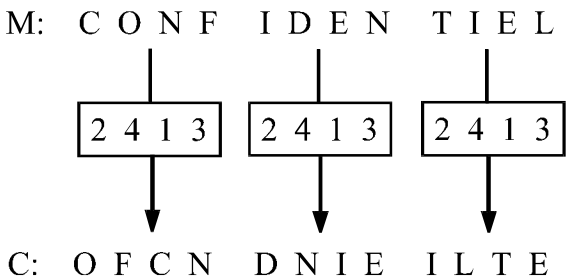
Les algorithmes de transposition consistent à réordonner les caractères sur la base d'une permutation. Ils opèrent en général sur des blocs avec une période fixe d. Soient Zd l'ensemble des entiers de 1 à d et f la permutation sur Zd. La clé de l'algorithme est donnée par la paire K = (d, f). Les blocs successifs de d caractères du message M sont chiffrés en permutant les caractères selon la transformation f.

Exemple :

On cherche à chiffrer le message CONFIDENTIEL, en prenant d = 4 et f données par la permutation suivante

i:	1	2	3	4
f (i):	2	4	1	3

On obtient:



IV.2.2 Chiffrement moderne

Le chiffrement moderne repose sur les mathématiques. Il utilise des algorithmes à clé que l'on subdivise en deux familles, les algorithmes symétriques et les algorithmes asymétriques.

a. Chiffrement symétrique

Le cryptage symétrique ou *chiffrement à clé privée ou clé secrète* a besoin d'une seule clé pour crypter et décrypter le message. Cette clé commune partagée entre l'émetteur et le récepteur doit être tenue secrète. L'avantage est que ces algorithmes sont assez rapides et permettent le cryptage d'un grand nombre de données. Mais l'inconvénient est que la clé doit être transmise par un canal secret.

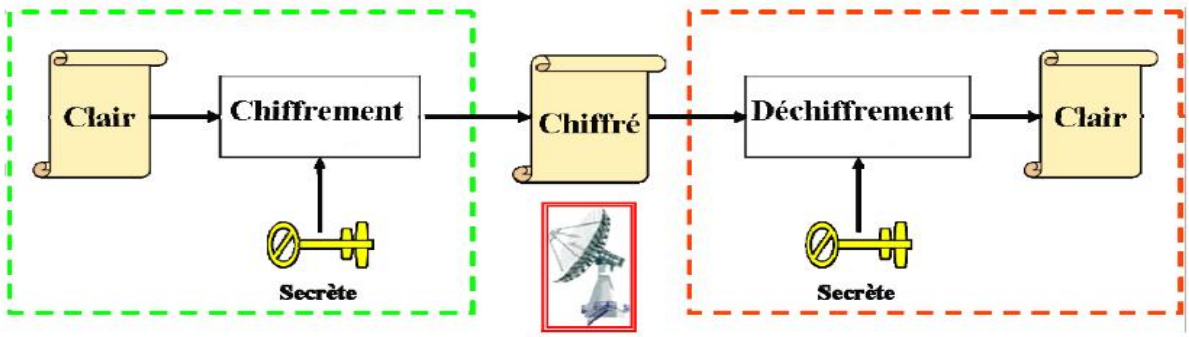


Figure I.3 : chiffrement symétrique

En pratique, on utilise principalement un chiffrement en continu ou chiffrement par flux lorsque la vitesse de traitement est essentielle (téléphonie, liaison entre unité centrale et périphériques). Lorsque la sécurité domine, en emploie plutôt un chiffrement par blocs.

- **Chiffrement par flux**

Ce système chiffre les données bit par bit quelle que soit la longueur du message à coder. Les algorithmes de chiffrement par flux (Stream cipher) peuvent être assimilés à des algorithmes de chiffrement par blocs, où le bloc a une dimension unitaire (1 bit). Leur avantage principal est leur grande rapidité à transmettre les données en complément du fait qu'ils interdisent la propagation d'erreurs éventuelles. Un autre point positif est la possibilité d'utilisation avec des équipements possédant une faible mémoire.

- **chiffrement par blocs**

Dans le chiffrement par bloc (*block cipher*), le message est découpé en blocs de même longueur avant de leur appliquer l'algorithme bloc par bloc. La taille du bloc est comprise entre 32 et 512 bits. Le premier standard dans le milieu des années 1990 était de 64 bits. Depuis l'année 2000, il est passé à 128 bits.

Le chiffrement par blocs opère selon quatre modes :

Ø Mode ECB (Electronic Code Book)

C'est le mode le plus simple. Il est appelé également mode dictionnaire. Le message est subdivisé en plusieurs blocs qui sont chiffrés séparément les uns après les autres. Le bloc chiffré est obtenu comme suit : $C_i = E_k$ (Mi). Le principal inconvénient de cette méthode est que deux blocs clairs identiques engendreront deux blocs chiffrés identiques, ce qui simplifie le travail du cryptanaliste. En revanche, la propagation des erreurs est limitée à un seul bloc.

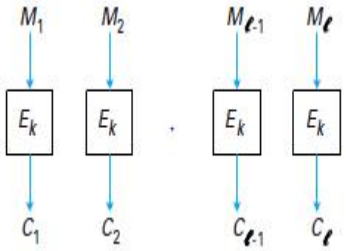


Figure I.4: mode ECB

E : méthode de chiffrement, **K** : la clé, **M** : message en clair et **C** : message crypté.

Ø Mode CBC (Cipher Block Chaining)

Ce mode permet d'introduire une complexité supplémentaire dans le processus de cryptage en créant une dépendance entre les blocs successifs. Chaque bloc clair M_i est d'abord additionné modulo 2 au bloc chiffré précédent C_{i-1} avant le chiffrement. Le bloc chiffré est obtenu comme suit : $C_i = E_k(M_i \oplus C_{i-1})$. Pour le premier bloc C_0, il faudra générer un bloc pour faire ce XOR, qu'on appelle vecteur d'initialisation (IV).
Ce mode constitue une meilleure approche par l'utilisation d'une rétroaction. La rétroaction fait que chaque bloc chiffré dépend d'actions réalisées auparavant, ce qui a pour avantage qu'un bloc de texte clair identique a toutes les chances d'être chiffré différemment à chaque apparition.

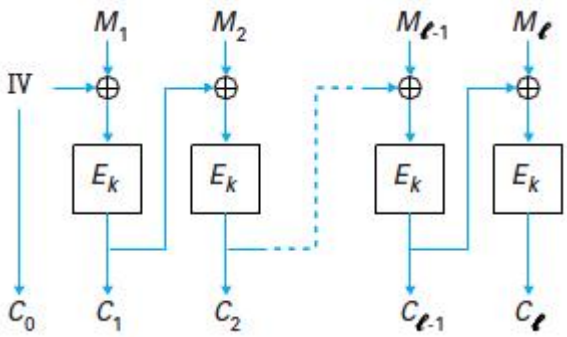


Figure I.5: mode CBC

Ø Mode CFB (Cipher Feed Back)

En mode CFB, les données peuvent être chiffrées en unités plus petites que la taille du bloc.CFB peut être utilisé à 64 bits (taille du bloc normal) ou tout autre CFB à n bits(ou n≤64 bits).comme le CBC, le CFB à en entrée un vecteur d'initialisation(IV).
$M = M_1 + M_2 + ...+ M_L$

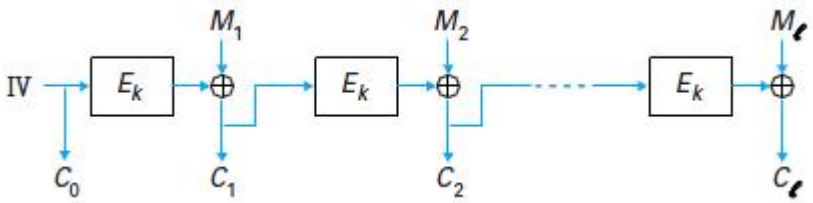


Figure I.6 : le mode CFB

Ø Mode OFB (output feedBack)

Ce mode est une variante de mode CFB. Il présente beaucoup de problèmes de sécurité et il est peu conseillé sauf dans le cas où sa longueur est égale à celle de l'algorithme utilisé.

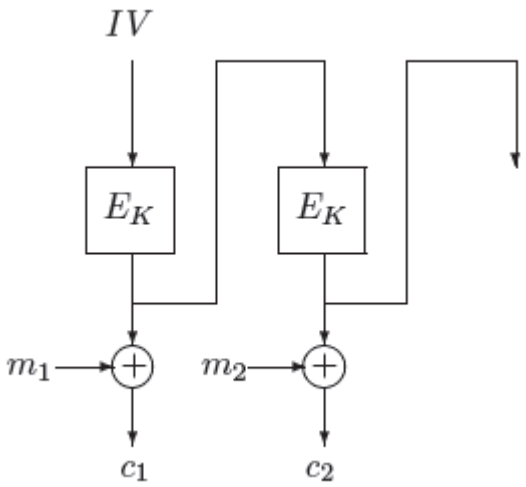


Figure I.7 : le mode OFB

b. Chiffrement asymétrique

Le cryptage asymétrique ou *chiffrement à clé publique* fonctionne sur le principe d'une paire de clés, l'une est publique, l'autre est privée. La clé publique est accessible à tout le monde et permet de crypter les messages. La clé privée, qui elle doit être tenue secrète, sert à décrypter le message chiffré. On s'aperçoit alors que tout le monde peut crypter un message et que seule la personne détenant la clé secrète peut le décrypter.

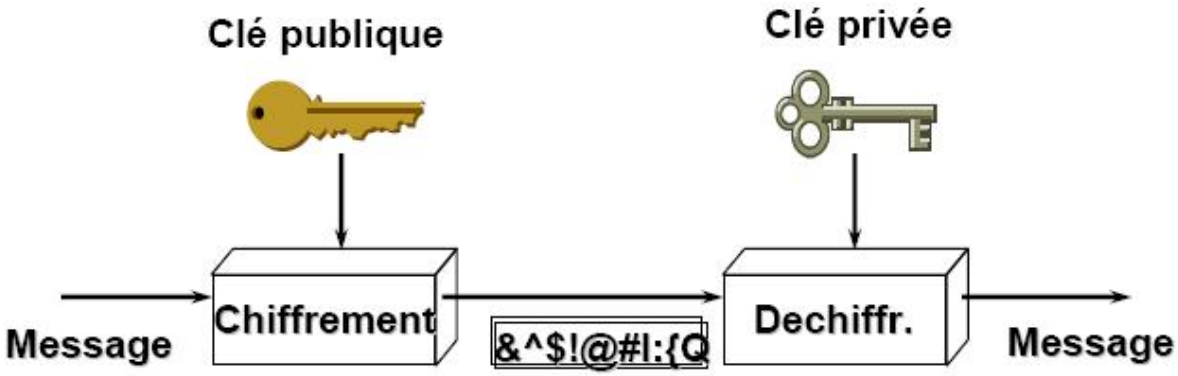


Figure I.8 : Chiffrement asymétrique

Pour faire une image avec le "monde réel", il s'agit pour un utilisateur de créer une clé aléatoire (la clé privée), puis de fabriquer un grand nombre de cadenas (clé publique) qu'il dispose dans un casier accessible à tous (le casier joue le rôle de canal non sécurisé). Pour lui faire parvenir un document, Un correspondant prend un cadenas ouvert, ferme une valisette contenant le document à l'aide de ce cadenas, puis envoie la valisette au propriétaire de la clé publique (le cadenas). Seul le propriétaire sera alors en mesure d'ouvrir la valisette avec sa clé privée.

La cryptographie asymétrique se base sur des propriétés arithmétiques qui exigent des calculs complexes sur de très grands nombres pour que l'algorithme soit suffisamment sûr, ce qui fait que le temps de cryptage est beaucoup plus long que dans le cas des algorithmes symétriques.

L'inconvénient avec les algorithmes symétriques est que la clé doit être transmise de manière sécurisée sur un canal authentifié. Une combinaison des deux systèmes est possible, c'est ce que l'on appelle la cryptographie hybride.

c. Chiffrement hybride

Un système hybride procède de la manière suivante : une clé aléatoire est générée pour un algorithme symétrique. Cette clé est utilisée pour chiffrer le message. La clé est ensuite chiffrée grâce à la clé publique du destinataire du message, c'est ici qu'intervient la cryptographie asymétrique. Chiffrer l'ensemble du message avec un algorithme asymétrique serait bien plus lourd, c'est pourquoi on préfère passer par un algorithme symétrique. Il suffit ensuite d'envoyer le message chiffré avec l'algorithme symétrique accompagné de la clé chiffrée correspondante. Le destinataire déchiffre la clé symétrique avec sa clé privée et via un déchiffrement symétrique.

Ces deux types de chiffrement symétrique et asymétrique utilisent des algorithmes dits réversibles, c'est à dire qu'il est possible de retrouver le message original à partir du message chiffré en utilisant la transformation inverse. Il est également possible d'utiliser des algorithmes irréversibles qui eux, ne permettent pas de retrouver le message clair à partir du cryptogramme. Il est seulement possible de vérifier le message original par une nouvelle application de l'algorithme et comparaison avec le message chiffré. Ce type d'algorithmes sert notamment à effectuer des contrôles d'intégrité et/ou d'origine.

IV.2.3 Systèmes irréversibles

Les systèmes irréversibles sont souvent associés à la génération de signatures permettant d'authentifier le message et son origine. Dans un système irréversible, le message chiffré produit un condensé du message original. Le message chiffré C est généré en utilisant un algorithme basé sur une fonction de hachage H à sens unique (figure I.9).

Cette fonction de hachage doit posséder les propriétés suivantes :

- elle opère sur des messages M de taille quelconque
- elle génère un message chiffré C de taille fixe
- étant donné M, il est facile de déterminer C
- étant donné C, il est pratiquement impossible de retrouver M
- pour tout message M, il est impossible de trouver un autre message M' tel que H(M) = H(M')

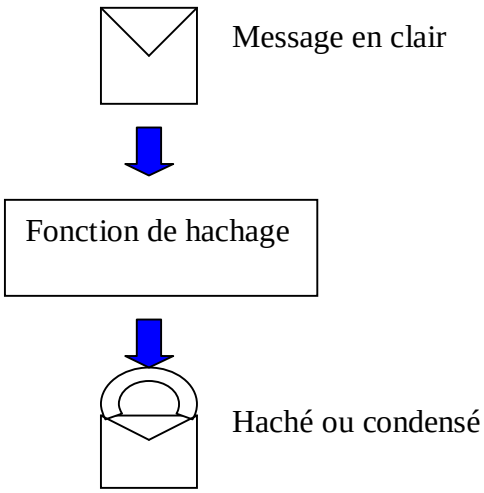


Figure I.9 : Système irréversible.

Le rôle d'une fonction de hachage est de produire une empreinte d'un fichier, d'un message ou d'un bloc de données. Le message chiffré C représente cette empreinte.

Deux familles principales d'algorithmes de calcul d'empreinte sont utilisées : la famille des **MD** (Message Digest) et la famille des **SHA** (Secure Hash Algorithm). Parmi eux, on citera :

L'algorithme MD2, la première version inventée en 1989 qui a ensuite été supplanté par l'algorithme MD4. Finalement, l'algorithme MD4 a été abandonné suite à des faiblesses de conception mises en évidence par des attaques.

MD5 est l'évolution de MD4 faite par Ronald Rivest en 1991 afin de pallier les faiblesses identifiées. Il produit des hachés de 128 bits (16 octets) à partir de messages de 512 bits (64 octets). Il est courant de voir des documents en téléchargement sur Internet accompagnés d'un fichier MD5, il s'agit du condensé du document permettant de vérifier son intégrité.

Le SHA-0 est l'ancêtre des algorithmes SHA. Il a été inventé en 1993 et il a été rapidement abandonné pour des raisons de sécurité insuffisante. Il était soupçonné de contenir des faiblesses qui permettraient d'aboutir rapidement à des collisions.

Le SHA-0 a été modifié pour devenir le SHA-1 en 1995. Le résultat généré est un nombre sur 160 bits (20 octets) à partir de messages de 512 bits (64 octets).

D'autres versions de cet algorithme (SHA-256, SHA-384 et SHA-512) existent aussi. Elles offrent respectivement des empreintes numériques de 256, 384 et 512 bits (soient 32, 48 et 64 octets).

La dernière nouveauté en matière de cryptologie est la cryptographie quantique. Si jusqu'à présent, les différents algorithmes présentés reposent en grande partie sur les mathématiques, Cette nouvelle classe d'algorithmes n'est plus basée sur des principes mathématiques mais plutôt sur les principes de la physique quantique.

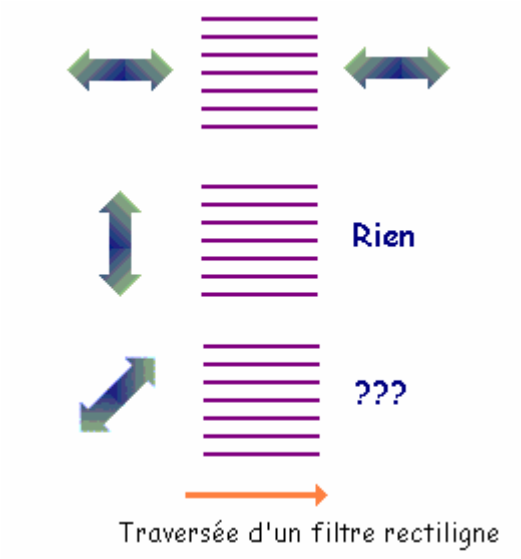
IV.2.4 Chiffrement quantique

La cryptographie quantique, plus correctement nommée distribution quantique de clés, désigne un ensemble de protocoles permettant de distribuer une clé de cryptage secrète entre deux interlocuteurs distants, tout en assurant la sécurité de la transmission grâce aux lois de la physique quantique et de la théorie de l'information. Cette clé secrète peut ensuite être utilisée dans un algorithme de chiffrement symétrique, afin de chiffrer et déchiffrer des données confidentielles. La cryptographie quantique repose sur les propriétés quantiques du photon polarisé.

IV.2.4.1 Le principe de fonctionnement de ce système

Dans le transport de "clé quantique", l'information est donc transportée par les photons. Chaque photon peut être polarisé, c'est-à-dire que l'on impose une direction à son champ électrique. La polarisation est mesurée par un angle qui varie de 0° à 180°. Dans le protocole que nous décrivons, dû aux canadiens CH.Bennett et G.Brassard, la polarisation peut prendre 4 valeurs : 0°, 45°, 90°, 135°. Pour les photons polarisés de 0° à 90°, nous parlons de "polarisation rectiligne", pour ceux polarisés de 45° à 135°, de "polarisation diagonale" :

Il nous faut pouvoir détecter la polarisation des photons. Pour cela, nous utilisons un filtre polarisant suivi d'un détecteur de photons. Si un photon polarisé à 0° rencontre un filtre polarisant orienté à 0°, il traverse ce filtre polarisant et est enregistré par le détecteur placé juste après. Si un photon polarisé à 90° rencontre le même filtre, il est immédiatement stoppé, et le détecteur n'enregistre rien. Maintenant, si le photon est polarisé diagonalement (45° ou 135°), une fois sur deux, il traverse le filtre (superposition de deux états polarisés de manière rectiligne), et une fois sur deux, il est stoppé. Si nous pouvons distinguer entre une polarisation à 0° et à 90°, il est impossible de distinguer en même temps entre une polarisation à 45° et à 135° De la même façon, on peut utiliser un filtre polarisant orienté à 45° : il laisse passer les photons polarisés à 45°, stoppe ceux polarisés à 135°, et se comporte aléatoirement avec ceux à 0° et 90°



IV. 3 Gestion des clés

La vulnérabilité d'un système de sécurité dépend en partie de la complexité des algorithmes utilisés mais également et surtout de la protection des clés de chiffrement. Au delà de l'aspect mathématique des clés de chiffrement, le problème le plus concret et fondamental est celui de la gestion des clés qui traite de leur choix, de leur attribution et de leur transport.

IV. 3. 1 Distribution des clés

La mise en œuvre d'un système cryptographique doit obligatoirement passer par la diffusion des clés aux correspondants concernés. Dans le cas du chiffrement symétrique, ceci se traduit par le choix d'une clé commune secrète et son envoi par un canal sûr (envoi postal, porteur). Dans le cas du chiffrement asymétrique, les correspondants doivent s'échanger leurs clés publiques. L'entité qui s'occupe de la conservation et de la divulgation des clés publiques est appelée le gestionnaire de clés. Ce peut être par exemple un serveur, où ces clés peuvent être consultées.

IV. 3. 2 Certification des clés

Afin d'éviter toute tentative de déguisement, il faut mettre en place un dispositif de certification permettant de vérifier que la clé publique est bien associée au détenteur légitime et d'authentifier le gestionnaire de clés.

Ainsi un certificat permet d'associer une clé publique à une entité (une personne, une machine, ...) afin d'en assurer la validité. Le certificat est en quelque sorte la carte d'identité de la clé publique, délivré par un organisme appelé *autorité de certification.*

Les informations contenues dans le certificat sont :

- Le nom de l'autorité de certification
- Le nom du propriétaire du certificat
- La date de validité du certificat
- L'algorithme de chiffrement utilisé
- La clé publique du propriétaire

L'ensemble de ces informations est signé par l'autorité de certification, cela signifie qu'une fonction de hachage crée une empreinte de ces informations, puis ce condensé est chiffré à l'aide de la clé privée de l'autorité de certification, la clé publique ayant été préalablement largement diffusée afin de permettre aux utilisateurs de vérifier la signature avec la clé publique de l'*autorité de certification*.

Conclusion

La cryptographie constitue la meilleure méthode de sécurisation de la confidentialité des données. Après avoir présenté les différents types de chiffrement, on peut dire qu'il est nécessaire de distinguer les deux cas d'algorithmes symétriques et algorithmes asymétriques. En effet, chacun d'eux a ses particularités.

L'avantage majeur du cryptage symétrique est sa rapidité (quelques dixièmes de secondes pour un message d'un mégaoctet). Son principal inconvénient réside dans l'échange de la clé. Le cryptage asymétrique est beaucoup plus robuste. Cependant sa faiblesse est la lenteur du traitement des données due à des calculs compliqués.

Cette étude nous a permis de cerner les avantages et les faiblesses des divers types de chiffrement. Afin de remédier à leurs faiblesses et particulièrement le problème de distribution des clefs, de nouvelles techniques ont vu le jour à savoir la cryptographie quantique que l'on va étudier plus en détail dans le chapitre III.

Chapitre II

Les algorithmes de cryptage

Introduction

A l'heure actuelle, deux classes d'algorithmes de cryptographie sont utilisées, ceux basés sur une clé symétrique (ou clé secrète) et ceux basés sur une clé asymétrique (ou clé publique). Parmi les algorithmes développés, l'algorithme DES constitue un standard dans le domaine des algorithmes symétriques, tout comme l'algorithme RSA constitue un standard dans le domaine des algorithmes asymétriques

II.1 Algorithmes symétriques [ALW 06]

Ces algorithmes utilisent une même clé pour le chiffrement et le déchiffrement. Cette clé est tenue secrète.

II.1.1 DES (Data Encryption Standard)

L'algorithme DES a été développé en novembre 1976 par IBM. C'est un algorithme de chiffrement par blocs de 64 bits, utilisant également une clé de 64 bits et basé sur un ensemble de transformations composé de transpositions et de substitutions. Son fonctionnement est illustré par la (figure II.1).

Les grandes lignes de l'algorithme sont les suivantes :

- Fractionnement du texte en blocs de 64 bits (8 octets)
- Permutation initiale des blocs
- Découpage des blocs en deux parties : gauche et droite, nommées *G* et *D*
- Etapes de permutation et de substitution (appelées rondes) répétées 16 fois
- Recollement des parties gauche et droite puis permutation finale inverse

Le processus de chiffrement est composé d'une permutation initiale IP, suivie de 16 rondes d'opérations identiques (fonction F) utilisant 16 clés secondaires K_0 à K_{15}, puis d'une permutation finale inverse IP^{-1}.A chaque ronde i, la fonction F est appliquée au bloc de données après génération de la clé secondaire qui lui est spécifique.

La fonction F c'est une fonction mathématique qui représente l'élément important sur lequel repose la sécurité du DES. Consiste des opérations de permutations, substitutions et combinaisons des données par un ou exclusif avec la clé secondaire.

Les permutations

Les permutations sont effectuées à l'aide de tableaux standard à l'algorithme DES. Ces permutations peuvent être bijectives, compressives ou expansives.

Pour les permutations bijectives, il y a autant de bits avant et après la permutation : c'est le cas des permutations initiale et finale (64 bits).

Pour les permutations compressives, le nombre de bits après permutation est inférieur à celui d'avant permutation : c'est le cas de la permutation initiale de la clé (de 64 à 56 bits) et des permutations à chaque ronde, de la clé (de 56 à 48 bits).

Pour les permutations expansives, le nombre de bits final est supérieur au nombre de bits initial : c'est le cas de la première permutation à chaque ronde de la partie droite du message (de 32 à 48 bits).

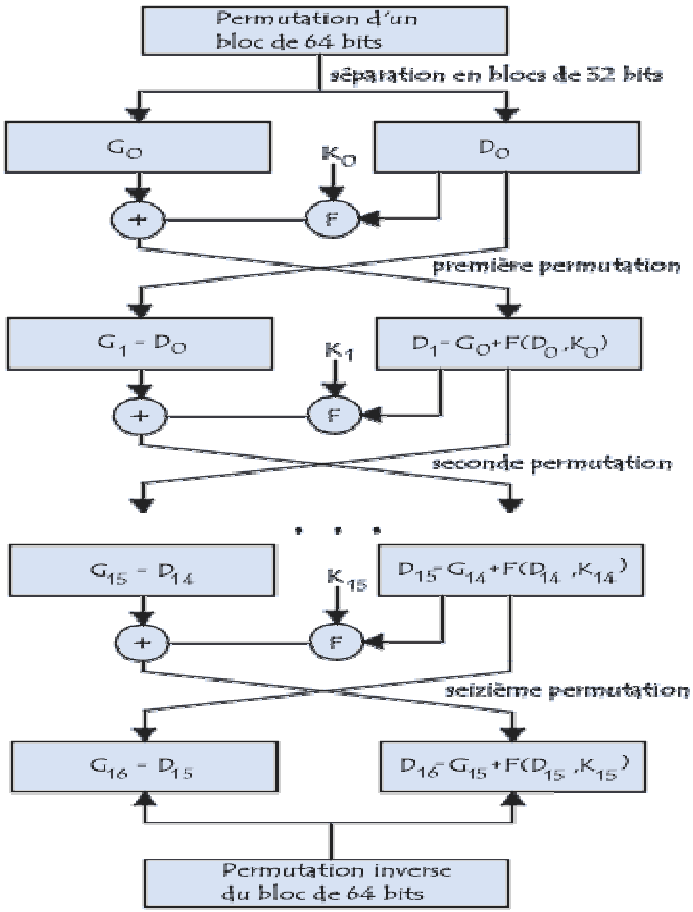


Figure II.1 : fonctionnement du DES

Etape 1 : **permutation initiale**

Bloc en entrée		bloc en sortie
1 2 3 4 5 6 7 8		58 50 42 34 26 18 10 2
9 10 11 12 13 14 15 16		60 52 44 36 28 20 12 4
17 18 19 20 21 22 23 24		62 54 46 38 30 22 14 6
25 26 27 28 29 30 31 32	**PI**	64 56 48 40 32 24 16 8
33 34 35 36 37 38 39 40		57 49 41 33 25 17 9 1
41 42 43 44 45 46 47 48		59 51 43 35 27 19 11 3
49 50 51 52 53 54 55 56		61 53 45 37 29 21 13 5
57 58 59 60 61 62 63 64		63 55 47 39 31 23 15 7

Le premier bit du bloc résultant de la permutation initiale est le $58^{ème}$ bit du bloc en entrée, le deuxième bit est le $50^{ème}$ bit et ainsi de suite.

Etapes 2 à 17

Le bloc de donnée de 64 bits est divisé en deux blocs de 32 bits, "G" et "D", après être passé dans la permutation initiale. Ainsi, "G" contient les bits pairs et "D" contient les bits impairs.

G	**D**
58 52 42 34 26 18 10 2	57 49 41 33 25 17 9 1
60 52 44 36 28 20 12 4	59 51 43 35 27 19 11 3
62 54 46 28 30 22 14 6	61 53 45 37 29 21 13 5
64 56 48 40 32 24 16 8	63 55 47 39 31 23 15 7

Pour calculer F on procède comme suit :

-extension de D_{i-1} de 32 bits en utilisant la table E.

La fonction **E** appelée table d'extension prend un bloc de 32 bits en entrée et donne un bloc de 48 bits en sortie. Les 48 bits, représentés en huit blocs de 6 bits chacun, sont obtenus en sélectionnant les bits du bloc en entrée selon le tableau suivant :

32 1 2 3 4 5

4 5 6 7 8 9

8 9 10 11 12 13

12 13 14 15 16 17

16 17 18 19 20 21

20 21 22 23 24 25

24 25 26 27 28 29

28 29 30 31 32 1

Ainsi les trois premiers bits du bloc sortant sont le 32ème, le 1er et le 2ème bit du bloc en entrée.

-On procède ensuite à des substitutions sur les huit blocs à l'aide de huit tables standard S appelées *boîtes de substitution S-Boxes*. La table S prend un bloc de 6 bits comme entrée et donne un bloc de 4 bits en sortie. Chaque bloc i est manipulé par une table S_i différente. Le tableau ci après représente la première fonction de sélection S1

S1																
Ligne N°	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

Chaque table S a 4 rangs et 16 colonnes. Chaque cellule de la table est un nombre de 4 bits. Les 6 bits du bloc d'entrée spécifient le rang et la colonne à exploiter pour obtenir la sortie. Si les bits du bloc d'entrée sont étiquetés b1, b2, b3, b4, b5, b6, les bits b1 et b6 sont combinés pour donner un nombre entre 0 et 3 correspondant à un rang dans la table.

Les 4 bits du milieu b2 à b5 sont combinés à leur tour pour former un nombre de 4 bits, soit un nombre compris entre 0 et 15, correspondant à une colonne dans la table.

Exemple : Soit un bloc B correspondant à 011011

b1b6 = 01 = 1 b2b3b4b5 = 1101 = 13

Le bloc de 4 bits de sortie sera "5" en base 10, donc "0101" en base 2.

-On dispose maintenant de 8 blocs de 4 bits chacun, soit une fonction de 32 bits qui sera l'entrée d'une table de permutation p.

La permutation "**P**" sur le bloc de 32 bits est représentée par la table suivante :

16	7	20	21
29	12	28	17
1	15	23	26
5	18	31	10
2	8	24	14
32	27	3	9
19	13	30	6
22	11	4	25

Le résultat est un bloc de 32 bits. A la suite des 16 rondes, le bloc de pré-sortie est $D_{16}G_{16}$

Permutation finale

Cette permutation correspond à l'inverse de la permutation initiale **IP^{-1}**.

40 8 48 16 56 24 64 32

39 7 47 15 55 23 63 31

38 6 46 14 54 22 62 30

37 5 45 13 53 21 61 29

36 4 44 12 52 20 60 28

35 3 43 11 51 19 59 27

34 2 42 10 50 18 58 26

33 1 41 9 49 17 57 25

Ainsi, le premier bit du bloc final est le $40^{ème}$ bit du bloc précédent, le deuxième bit est le $8^{ème}$ bit, etc.

Génération des sous clés

Le processus de génération des clés secondaires est composé d'une permutation CP-1 réduisant la clé à 56 bits (enlever les bits de parités), suivie de 16 itérations combinant une opération de décalage circulaire gauche et une permutation CP-2. Chaque itération permet de générer une clé secondaire Ki de 48 bits utilisée dans la ronde i correspondante.

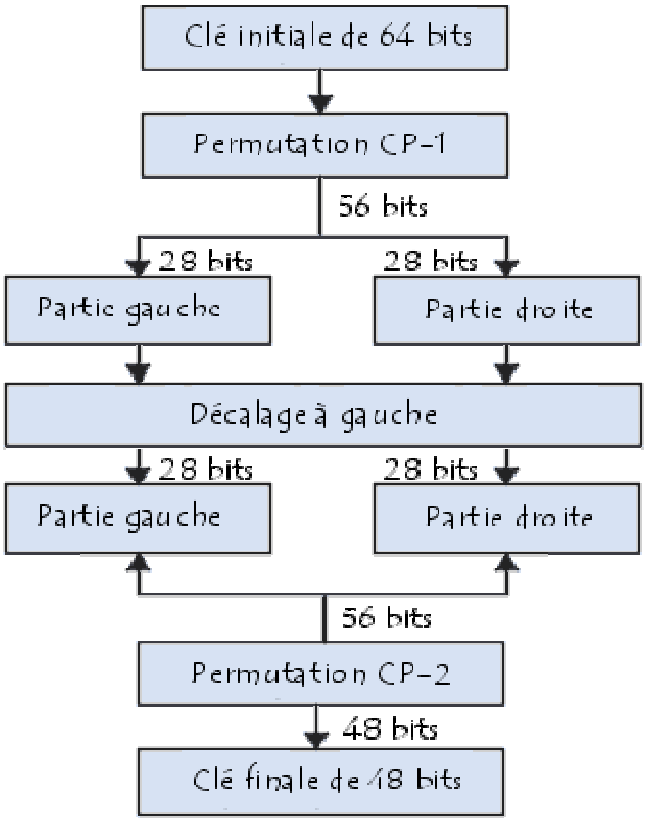


Figure II.2: Génération des clés secondaires

DES est utilisé massivement par les banques pour garantir la sécurité et la confidentialité des données circulant sur les réseaux bancaires. Le système d'exploitation Unix, lui aussi, utilise ce procédé pour crypter ses mots de passe

Avec les progrès réalisés dans le domaine de la cryptanalyse, en particulier la progression de la puissance de calcul des processeurs d'aujourd'hui, le DES avec sa longueur de clé fixe de 56 bits n'est plus considéré comme pratiquement sûr. En effet, de nombreuses attaques cryptographiques ont permis d'y découvrir des petites faiblesses. C'est pourquoi un nouveau standard a pris sa place depuis l'année 2000, il s'agit de l'algorithme AES.

II.1.2 AUTRES ALGORITHMES

-Algorithme AES *(Advanced Encryption Standard)*

L'AES, connu aussi sous le nom de Rijndael, de ses deux concepteurs Joan Daemen et Vincent Rijmen a été choisi en octobre 2000 par le NIST américain (National Institute of

Standards and Technology) pour être le nouveau standard de chiffrement pour les organisations du gouvernement des États-Unis, en remplacement du DES vieillissant.

L'algorithme prend en entrée un bloc de 128 bits (16 octets) et fonctionne avec une clé de 128, 192 ou 256 bits. Les 16 octets en entrée sont permutés selon une table définie au préalable. Ces octets sont ensuite placés dans une matrice de 4x4 éléments et ses lignes subissent une rotation vers la droite. L'incrément pour la rotation varie selon le numéro de la ligne. Une transformation linéaire est ensuite appliquée sur la matrice. Cette transformation consiste en la multiplication binaire de chaque élément de la matrice avec des polynômes issus d'une matrice auxiliaire. La transformation linéaire garantit une meilleure diffusion (propagation des bits dans la structure) sur plusieurs tours. Finalement, un XOR entre la matrice et une autre matrice permet d'obtenir une matrice intermédiaire. Ces différentes opérations sont répétées plusieurs fois et définissent un tour. Pour une clé de 128, 192 ou 256 bits, l'algorithme nécessite respectivement 10, 12 ou 14 tours.

-Algorithme IDEA (International Data Encryption Algorithm)

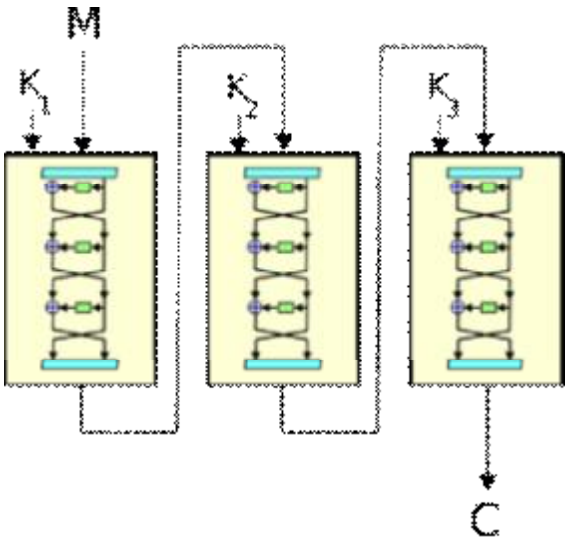
Développé à Zurich en Suisse par Xuejia Lai et James Massey en 1992, le chiffrement par blocs IDEA utilise des blocs de 64 bits et est contrôlé par une clé de 128 bits. Le déchiffrement est effectué avec la même fonction que celle utilisée dans le chiffrement. L'algorithme a innové dans son domaine en utilisant des opérations de trois groupes algébriques différents.

Le processus de chiffrement est le même que pour le déchiffrement, à moins d'une utilisation de différentes sous-clés, ce qui est rare. En gros, le processus se résume à huit étapes de chiffrement identiques, les rounds, suivis d'une transformation au bloc de sortie. Contrairement au DES et à Blowfish, IDEA n'utilise aucune S-Box. L'algorithme peut être utilisé avec les modes d'opération ECB, CBC, CFB et OFB. Sa vitesse est environ la même que le DES.

Les opérations utilisées sont le OU exclusif, l'addition modulo 2^{16} et la multiplication modulo $2^{16} + 1$ (65 537, qui est un nombre premier). Le bloc de 64 bits en entrée est tout d'abord divisé en quatre blocs de 16 bits chacun. À noter que toutes les opérations algébriques utilisées dans le chiffrement fonctionnent avec des blocs de 16 bits. Un processus produit, pour chacun des 8 rounds, 6 sous-clés de 16 bits chacune selon la clé de 128 bits. Avec quatre autres clés générées pour la transformation finale à la suite des huit rounds, un total de 52 sous-clés doit être généré.

-Algorithme Triple DES

Le *Triple DES* enchaîne 3 applications successives de l'algorithme DES sur le même bloc de données de 64 bits, avec des clés en principe différentes (parfois la même est utilisée selon les implémentations). La longueur de la clé est de 168 bits (3 x 56 bits).



Le *Triple DES* est généralement utilisé avec seulement deux clés différentes. Le mode d'usage standard est de l'utiliser en mode EDE (Encryption, Decryption, Encryption, c'est-à-dire Chiffrement, Déchiffrement, Chiffrement), ce qui le rend compatible avec DES quand on utilise trois fois la même clé. Dans le cas d'une implémentation matérielle, cela permet d'utiliser le même composant pour respecter le standard DES et le standard Triple DES. Cet algorithme est en voie de disparition également, parce que lent et son niveau de sécurité est peu performant. On le rencontre encore toutefois sur de nombreux équipements matériels.

-Algorithme Blowfish

Conçu en 1993 et entièrement libre, cet algorithme utilise des clés de 32 à 448 bits. Il est environ 5 fois plus rapide que le Triple DES. Il reste encore très fiable (selon la taille de la clé).

-Algorithme TwoFish

Cet algorithme reprend certaines caractéristiques de Blowfish mais avec des clés de 128 à 256 bits. Il a été l'un des concurrents possibles à l'AES, mais n'a pas été retenu, malgré sa rapidité. C'est un algorithme résistant.

-Algorithme Serpent

Comme TwoFish, l'algorithme Serpent été présenté à l'évaluation pour remplacer l'AES mais n'a pas été retenu car trop proche du DES. Pourtant, il n'existe pas d'attaque connue contre cet algorithme. Les clés générées peuvent être de 128, 192 ou 256 bits.

-Les algorithmes de la famille RC (*Ron's Code) :* RC2, RC5 *et RC6*

RC2

Le RC2 a été conçu en 1989. Il avait été programmé pour être efficace avec les processeurs de 16 bits en remplacement du DES. Il opère sur des blocs de 64 bits. La longueur de la clé est variable, de 1 octet (8 bits) à 128 octets (1024 bits). Habituellement, l'algorithme fonctionne avec une clé de 64 bits.

Le procédé nouveau avec cet algorithme a été d'offrir aux utilisateurs la possibilité de choisir la longueur de la clé.

RC5

A été conçu en 1995. Il a l'avantage d'avoir une longueur de bloc de données variable, un nombre de rounds variable et une clé de longueur variable. Ainsi, l'utilisateur a le contrôle sur le rapport entre la vitesse d'exécution et la sécurité de son chiffrement. En général, une longue clé et un nombre élevé de rounds assurent une plus grande sécurité. La taille des blocs de données pour sa part accommode différentes architectures de systèmes.

La simplicité de l'algorithme du RC5 rend son implémentation facile et, le plus important, rend son analyse plus aisée. De plus, la forte utilisation des décalages de bits (appelés rotations).
En plus du mode EBC , le RC5 est aussi utilisé avec le mode CBC.

RC6

Le RC6 a été créé en 1998. Il propose des améliorations au RC5 et, comme celui-ci, est fortement dépendant de la transformation de décalage de bits (rotation). Comme le RC5, il a l'avantage d'avoir une longueur de bloc de données variable, un nombre de rounds variable et une clé de longueur variable.

Il est fondé sur un bloc de 128 bits et supporte des clés de 128, 192 et 256 bits. Sa modularité est cependant plus grande que ces contraintes liées à AES puisqu'il peut travailler avec des clés de taille variable (maximum 2048 bits), des nombres de tours différents (par défaut 20, au minimum 8 et un multiple de 4) et des blocs varie selon un mot de 8 bits.

-Algorithme CAST-128 (ou **CAST5**)

Est un algorithme de chiffrement par bloc utilisé par plusieurs logiciels. Il a été approuvé au Canada par le Communications Security Establishment pour une utilisation gouvernementale.

L'algorithme a été conçu en 1996 par Carlisle Adams et Stafford Tavares. Une version avec une clé plus grande, CAST-256 (ancien candidat pour AES), a été dérivée à partir de CAST-128. Le terme de « CAST » serait basé sur les initiales des inventeurs.

CAST-128 il est de 12 ou 16 tours avec un bloc de 64 bits. La taille de la clé varie entre 40 et 128 bits (par incrément de 8 bits). La version complète avec ses 16 tours est utilisée quand la clé est supérieure à 80 bits. L'architecture interne du chiffrement comprend des S-Boxes de 8x32 éléments dont le contenu provient de fonctions dites *courbe*, des rotations qui varient selon la clé, des additions et des soustractions. Il y a trois types de tours mais ils ne varient que sur le choix exact de l'opérateur (addition, soustraction ou XOR).

II.2 Algorithmes asymétriques

Ces algorithmes utilisent une paire de clés différente pour le chiffrement et le déchiffrement. L'une des clés est publique, l'autre doit rester secrète pour identifier son détenteur.

II.2.1 RSA (*Rivest, Shamir, Adleman*)

L'algorithme RSA a été conçu en 1978 par Ron Rivest, Adi Shamir et Len Adleman. Il utilise la décomposition d'un nombre en facteurs premiers et une Clé de longueur variable (>512 bits).
Cet algorithme sert aussi bien à effectuer le chiffrement des données de taille réduite, mais permet également d'assurer le service d'authentification.

-Principe

L'algorithme fonctionne de la manière suivante :

1. Chaque correspondant génère une paire de clés (Kc, Kd)
2. L'une des 2 clés est révélée dans un répertoire public, l'autre est gardée secrète
3. Une personne A souhaitant envoyer un message à un correspondant B, consulte la clé publique dans le répertoire et envoie le message chiffré avec cette clé
4. Lorsque B reçoit le message, il utilise sa clé privée pour le déchiffrer

-Génération des clés

RSA repose sur la difficulté de décomposer de grands nombres en facteurs premiers. Les clés sont générées selon le processus suivant :

1. Choisir deux grands nombres premiers p et q

2. Calculer n le produit de ces 2 nombres (il est difficile de retrouver p et q à partir de n)
3. Calculer le nombre d'Euler de n $\phi(n) = (p-1)(q-1)$
4. Choisir un nombre aléatoire $e < \phi(n)$ et premier avec $\phi(n)$
5. Déterminer le nombre d tel que $ed \equiv 1 \bmod \phi(n)$
6. Choisir un des 2 nombres (e, d) comme clé secrète et l'autre comme clé publique

   Généralement e représente la clé publique Kc et d représente la clé secrète Kd

   Ainsi les quantités publiques sont : n et e les quantités secrètes sont : d et $\phi(n)$

-Chiffrement et déchiffrement

Soit ***M*** le message ***en clair.*** Le message chiffré **C** est obtenu en calculant

$$\mathbf{C} = \mathbf{M}^{Kc} \bmod (n)$$

Le déchiffrement pour retrouver le message en clair se fait comme suit :

$$\mathbf{C}^{Kd} \bmod (n) = \mathbf{M}^{Kc\,Kd} \bmod (n) = \mathbf{M}$$

Exemple : chiffrement de MESSAGE

A veut envoyer le message « MESSAGE » à **B**.

Soient $p = 79$ et $q = 127$, deux nombre premiers

On a : $n = pq = 10033$ et $\phi(n) = (p-1)(q-1) = 9828$

Soit $e = 97$ choisi aléatoirement et qui n'a pas de facteur commun avec 9828.

On cherche d tel que $ed \equiv 1 \bmod 9828$, soit $d = e^{-1} \bmod 9828 = 2533$

En suite **B** envoie les nombres n = 10033 et e = 97 à **A** pour qu'il puisse lui envoyer le message de manière confidentielle. **A** doit d'abord convertir son texte en une suite de nombres, le mieux est d'utiliser le code ASCII (Amercian Standard Code for Information Interchange). Ainsi :

M=77 E=69 S=83 A=65 G=71

Et le texte MESSAGE devient M : 77 69 83 83 65 71 69. Il décompose ensuite le message en blocs de 4 chiffres : 0077 6983 8365 7169.

A cadence de façon mathématique le message.

7169^{97} = 2605(mod 10033).

8365^{97} = 9691(mod 10033).

6983^{97} = 4000(mod 10033).

0077^{97} = 2642(mod 10033).

Le message codé est donc C : **2642 4000 9691 2605**

-Efficacité et robustesse de RSA

Il est facile de générer de grands nombres premiers, tout au moins en acceptant un taux d'erreur. Dans le cas de RSA, l'erreur n'est pas trop grave. En effet, si l'on commet une erreur en croyant que p et q sont premiers alors qu'ils ne le sont pas, le destinataire se rendra rapidement compte que les nombres ne sont pas premiers : soit la clé n'est pas inversible, soit certains blocs du message décrypté sont incompréhensibles. Dans ce cas, on peut procéder à un autre choix de p et q.

Le calcul du couple (e, d) est extrêmement facile, il suffit d'appliquer l'algorithme d'Euclide étendu. Le chiffrement et le déchiffrement sont réalisés par exponentiation modulaire.

La sécurité fournie par RSA repose essentiellement sur la difficulté à factoriser de grands nombres entiers. En effet, si un attaquant peut factoriser le nombre $n = p\,q$ de la clé publique, il peut alors déduire directement $\phi(n) = (p-1)(q-1)$ et donc calculer la clé privée a partir de la clé publique par l'algorithme d'Euclide étendu . Donc si l'on dispose d'un algorithme rapide pour factoriser de grands entiers, casser RSA devient facile aussi.

RSA est aujourd'hui utilisé dans une large variété de produits (téléphones, réseaux Ethernet, etc...), de logiciels de différentes marques (Microsoft, Apple, Novell, Sun), dans des industries et enfin dans les télécommunications.

II.2.2 AUTRES ALGORITHMES

-Algorithme Diffie_Hellman

C'est pratiquement le premier algorithme à clé publique à avoir été conçu et adopté comme référence en matière d'échange de clés. Il a été développé en 1974 par Whitfield Diffie et

Martin Hellman. Il est conçu pour établir un canal sécurisé entre deux correspondants et leur permettre d'échanger une clé secrète. Il est implémenté dans beaucoup de produits commerciaux. Son efficacité est basée sur la difficulté de calcul du logarithme discret.

La notion de logarithme discret peut être définie de la manière suivante :

Considérons d'abord la racine primitive d'un nombre p qui désigne un nombre dont les puissances génèrent tous les nombres compris entre 1 et p-1, ce qui veut dire, si a est une racine primitive du nombre p, les nombres a mod p, a^2 mod p,…a^{p-1} mod p sont distincts et constitué d'entiers de 1 a (p-1).

Pour tout entier b et une racine primitive a d'un nombre premier p, on peut trouver un exposant unique i tel que $b = a^i \bmod p$ avec $0 \leq i \leq p-1$. L'exposant i ainsi défini désigne le logarithme discret de b pour la base a modulo p.

-Le principe de l'algorithme est le suivant

Soit p un nombre premier et $\alpha < p$ sa racine primitive. Soient X, Y deux parties communicantes qui vont échanger leurs clés.

X choisit un nombre secret a et calcule sa clé publique $A = \alpha^a \bmod p$. Y choisit pour sa part un nombre secret b avec lequel il calcule sa clé publique $B = \alpha^b \bmod p$

X et Y s'échangent leurs clés publiques puis chacun calculera sa clé privée comme suit :

Pour X : $K=B^a \bmod p$ pour Y : $K=A^b \bmod p$

-Algorithme El Gamal

Cet algorithme a été développé par Taher El Gamal en 1984. Il trouve son utilité dans les signatures électroniques. Tout comme l'algorithme Diffie_Hellman, Il est basé sur la difficulté de calculer des logarithmes discrets.

Le chiffrement ElGamal est l'alternative la plus courante au chiffrement RSA. Il est facilement adaptable à de nombreux groupes cryptographiques et possède des propriétés homomorphiques, tout en restant sur (au sens des attaques à clairs choisis).
Malheureusement, l'usage en pratique du chiffrement ElGamal est complexe : les messages
Doivent être encodés en éléments du groupe avant le chiffrement proprement dit, ce qui requiert des conversions incommodes, limite la taille des messages. En utilisant des fonctions de hachage, il est certes possible de passer outre l'encodage du message en élément du groupe, mais alors, la sécurité dans le modèle standard n'est plus assurée.

-Conclusion

L'un des principaux problèmes concernant la cryptographie est le fait que les algorithmes utilisés dans les différents protocoles se servent principalement de la puissance des ordinateurs et des progrès des mathématiques. En effet, les algorithmes de chiffrement et de déchiffrement doivent être suffisamment rapides, donc peu complexes, pour ne pas trop ralentir leur exploitation. Inversement, les algorithmes de découverte des clés doivent être suffisamment complexes, donc lents, pour décourager ou ralentir les cryptanalyses Ceci se confirme par le critère de Shannon qui dit que la longueur de la clé doit être aussi longue que le message à transmettre; c'est pour cela que le message est découpé en blocs. Ainsi, plus la clé est longue et sécurisée lors de sa distribution, plus il sera difficile à un tiers non autorisé de déchiffrer les données interceptées. Pour résoudre le problème de distribution des clés, on utilise le chiffrement quantique.

Chapitre III

La cryptographie quantique

Introduction

La cryptographie quantique est née à la fin des années soixante avec les travaux de Stephen Wiesner, qui cependant n'ont été publiés qu'en 1983 [Wie.83]. Mais la première formalisation de cette nouvelle technologie remonte à 1984 et est due aux travaux de Charles. Bennett et Gilles Brassard [Ben.84]. La grande nouveauté est la possibilité de pouvoir manipuler et observer des objets quantiques individuels : photons, atomes, ions, etc…, et pas seulement d'agir sur le comportement quantique collectif d'un grand nombre de tels objets. Cette possibilité de manipuler et d'observer des objets quantiques élémentaires est à l'origine de l'information quantique, où ces objets quantiques élémentaires permettront de construire physiquement des quantum bits (qubits).

Un qubit est l'état quantique qui représente la plus petite unité de stockage d'information quantique. Si en informatique classique, on code l'information grâce à des bits pouvant être soit des 0, soit des 1, en informatique quantique, on va utiliser des bits quantiques ou qu bits, pouvant prendre un ensemble de valeurs beaucoup plus large. En effet, la physique quantique, avec son principe de superposition, permet à un état d'être un mélange d'autres états. Ainsi, un qubit peut prendre les valeurs 0 ou 1, mais aussi une combinaison de 0 et de 1 en même temps, par exemple un état constitue de 10 % de 0 et 90 % de 1 ou toute autre combinaison. De façon générale, l'état de ce dernier s'écrit $|\Psi\rangle = \alpha|0\rangle + \beta|1\rangle$ où les paramètres complexes α et β vérifient la condition de normalisation $|\alpha|^2 + |\beta|^2 = 1$

La visée de cette technique de nouvelle génération n'est plus tellement de garantir la confidentialité absolue des données, puisque les techniques plus traditionnelles répondent à ce besoin, mais de parvenir à résoudre un problème majeur de cryptographie : la transmission d'une clé de chiffrement sur un réseau public.

Un système de cryptographie quantique est utilisé pour transmettre une clé, et non le message en lui-même, et ceci pour deux raisons essentielles :

- Les bits d'informations communiqués par les mécanismes de la cryptographie quantique ne peuvent être qu'aléatoires. Ceci ne convient pas pour un message, mais convient parfaitement bien à une clé secrète, qui doit être aléatoire.
- Même si le mécanisme de la cryptographie quantique garantit que l'espionnage de la communication sera toujours détecté, il est possible que des bits d'information entrent en possession de l'espion avant que celui-ci ne soit détecté. Ceci est inacceptable pour un message, mais sans importance pour une clé aléatoire qui peut être simplement jetée en cas d'interception.

III. 1 Principe du cryptage quantique

La cryptographie quantique se base sur les lois de la physique quantique pour le traitement et la transmission de l'information. Elle permet de sécuriser la transmission de données en utilisant des clés générées et échangées à l'aide de particules quantiques, les photons. Comme la mécanique quantique stipule que toute observation de l'état quantique d'une particule

modifie cet état (principe d'incertitude d'Heisenberg), toute tentative d'interception de la clé par un espion peut en principe être repérée par les utilisateurs. S'il n'y a pas eu espionnage, une clé parfaitement secrète peut être extraite de la transmission, et celle-ci peut être utilisée dans tout algorithme de chiffrement symétrique afin de transmettre un message. La sécurité des protocoles de cryptographie quantique est donc absolue et garantie par les lois mêmes de la physique.

Dans les systèmes de télécommunications quantiques (figure III.1), les transmissions se font généralement par deux canaux : un canal quantique (destiné à l'échange des photons polarisés), et un canal classique non protégé (destiné à la discussion).

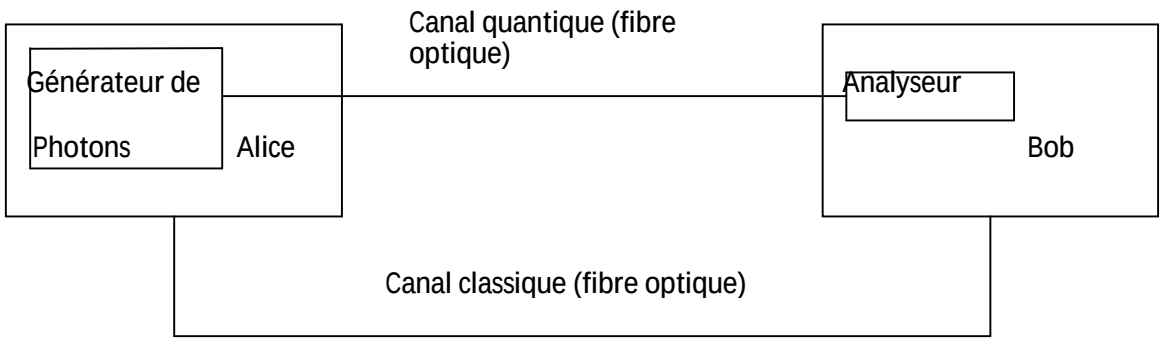


Figure III.1 : les systèmes quantiques.

– **Le canal quantique**

Il s'agit d'un câble de fibre optique permettant la transmission des photons. Bien que les systèmes à fibre optique soient très utilisés, de tels systèmes en cryptographie quantique, ne peuvent cependant pas fonctionner au-delà de la distance de 150 km [Kim.04]. Ceci est dû à la combinaison de la perte induite par la fibre optique et des bruits du détecteur, en plus de difficultés géographiques. C'est la raison qui fait que de plus en plus d'efforts sont fournis pour développer des systèmes qui se basent sur une liaison en air libre, où les photons sont envoyés entre deux télescopes distants.

– **Le canal classique**

Il s'agit généralement du réseau Internet.

Un des éléments les plus importants dans un système de cryptographie quantique est le choix de la source de photons et du compteur de photons. Essentiellement, la cryptographie quantique est basée sur les états de Fock à photon simple. Malheureusement, il est difficile en pratique de

réaliser ces états. Aussi, les expériences pratiques sont fondées sur les impulsions lasers faibles ou les paires de photons intriqués. Quant au compteur de photon, il est réalisé en utilisant une variété de techniques, comme le photomultiplicateur, la photodiode à avalanche, ou le détecteur multicanal.

III. 1. 1. Etat quantique

On sait que la lumière est un rayonnement électromagnétique ou un transfert d'énergie sous la forme d'ondes électromagnétiques, ces ondes se déplaçant dans l'espace selon une direction donnée. On peut également dire que la lumière est constituée d'un champ électrique E et d'un champ magnétique B orthogonal à E. Comme le champ magnétique peut se déduire du champ électrique en utilisant les équations de Maxwell, et que la majorité des instruments sont sensibles au champ électrique, nous utiliserons dans la suite ce dernier.

III. 1. 1. 1. Photon

Le photon est la particule élémentaire (quantum) qui constitue le rayonnement. L'état quantique dans ce contexte désigne l'ensemble des caractéristiques qui aident à décrire le photon : sa position, son énergie, sa polarisation, etc… Ici, on ne s'intéressera qu'à sa polarisation. Un photon peut ainsi être considéré comme étant un minuscule champ électrique oscillatoire. La direction de l'oscillation définit alors la polarisation du photon

III. 1. 1. 2. Polarisation d'un photon

La polarisation est le comportement du vecteur champ électrique E dans le plan orthogonal à la direction de propagation. On observe trois types de lumière polarisée : elliptique, circulaire et linéaire (figure III.2) La lumière provenant d'une source incohérente (comme la lumière naturelle) est dite non-polarisée : il s'agit d'un mélange statistique d'états de polarisation.

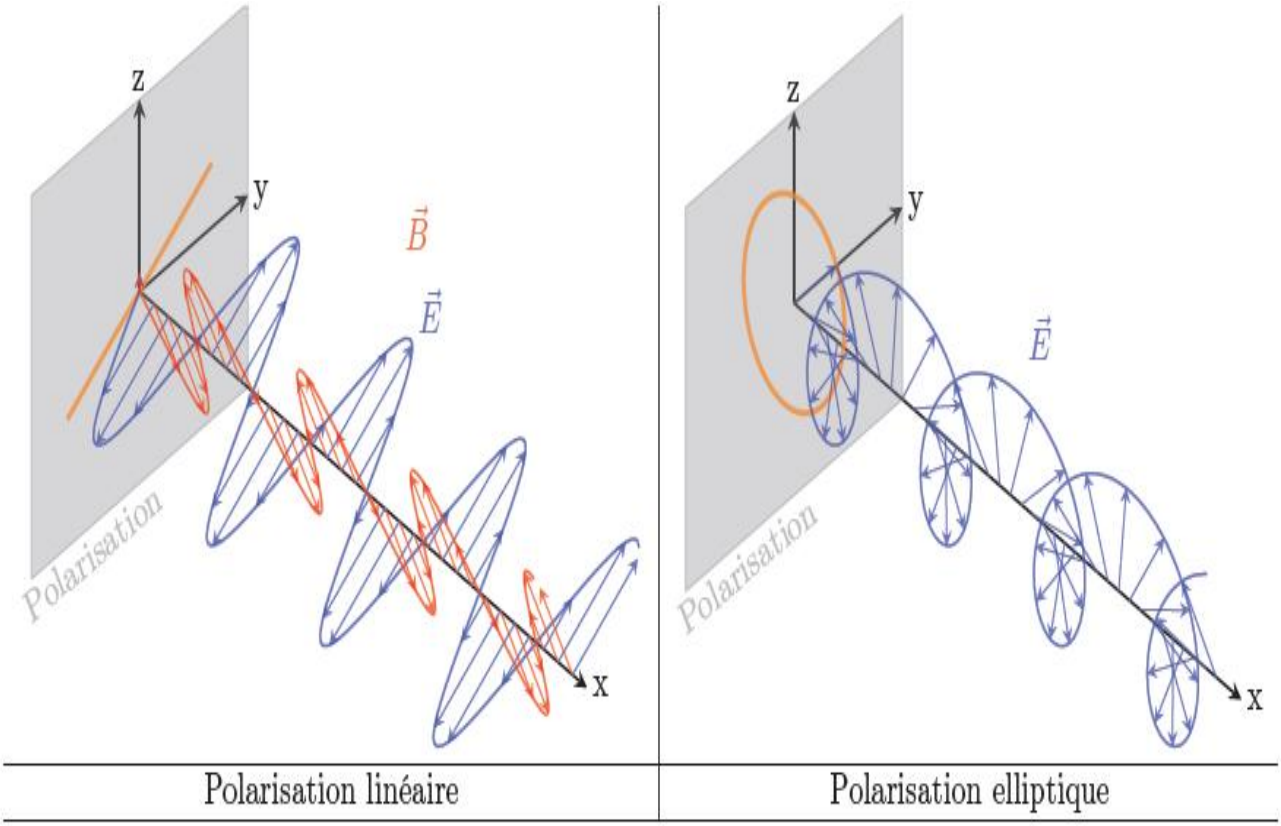


Figure : III.2 polarisation linéaire et elliptique de la lumière

Dans la suite nous travaillerons avec des photons uniques polarisés de façon linéaire et selon quatre angles différents avec l'axe (Oy) .La polarisation est mesurée par un angle qui varie de 0° à 180°. Habituellement, la polarisation peut prendre 4 valeurs (figure III.3) : 0°, 45°, 90°, 135°. Pour les photons polarisés de 0° à 90°, on parle de polarisation rectiligne, pour ceux polarisés de 45° à 135°, de polarisation diagonale. Chaque état polarisé représente une valeur binaire et par convention, on applique la valeur 0 à la polarisation horizontale et diagonale alors que la valeur 1 sera appliquée à la polarisation verticale et anti-diagonale.

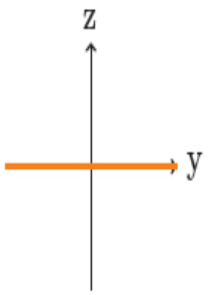
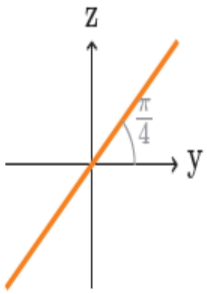
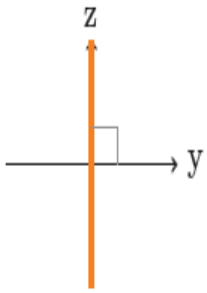
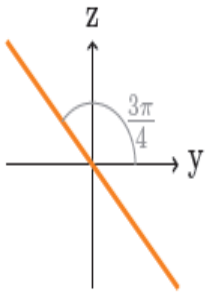
Nom	Horizontal	Diagonal	Vertical	Antidiagonal
Angle (rad)	0	$\frac{\pi}{4}$	$\frac{\pi}{2}$	$\frac{3\pi}{4}$
Polarisation				
Vecteur unitaire	$\|\rightarrow\rangle$	$\|\nearrow\rangle$	$\|\uparrow\rangle$	$\|\nwarrow\rangle$

Figure III.3: polarisation des photons

III. 1. 1. 3. Propriétés des photons polarisés

Lorsque l'on fait passer la lumière à travers un filtre polarisant, les photons seront absorbés ou transmis selon leur polarisation [Ngh 05] :

- Si le photon est polarisé parallèlement à l'angle d'orientation du filtre, alors ce photon sera transmis sans changement de polarisation.
- Si le photon est polarisé perpendiculairement à l'angle d'orientation du filtre, alors ce photon sera absorbé.
- Si le photon est polarisé selon une direction intermédiaire, alors ce photon sera transmis avec une probabilité $\cos^2(\alpha)$, où α est l'angle de polarisation du photon mesuré par rapport à l'angle d'orientation du filtre. C'est à dire que, si le photon est polarisé selon un angle γ et que le filtre est orienté selon un angle β, alors $\alpha = \gamma - \beta$. Si le photon est transmis, alors sa nouvelle polarisation correspondra à l'angle d'orientation du filtre.

La polarisation de la lumière transmise par le filtre polarisant est donc égale à l'angle d'orientation du filtre. Ainsi, les photons initialement polarisés selon un angle γ ont maintenant tous une polarisation correspondante à l'angle du filtre, soit un angle β, ce qui implique que leur polarisation initiale est complètement perdue.

III. 1. 1. 4. Transmission de la clé

Dans la transmission de la clé quantique, l'information est transportée par des photons, chaque photon étant polarisé de façon secrète, selon quatre angles différents sur la fibre optique. A la réception, il faut pouvoir détecter la polarisation de ces photons. Pour cela, on utilise un filtre polarisant suivi d'un détecteur de photons.

Chaque photon correspond à un bit d'information et prendra à l'arrivée la valeur 0 ou 1, qui correspond au choix d'alignement des photons, convenu entre les deux correspondants. Cette valeur est déterminée par la polarisation d'origine fixée par l'émetteur. Le destinataire dispose d'un filtre polarisant, avec seulement deux axes de réglage possible : oblique ou horizontal. Il en choisit un au hasard pour chaque photon reçu et seuls les photons correctement alignés sont reçu, les autres se perdent, Si le photon traverse le filtre, Bob note 0, sinon il note 1.voire la (figureIII.4).

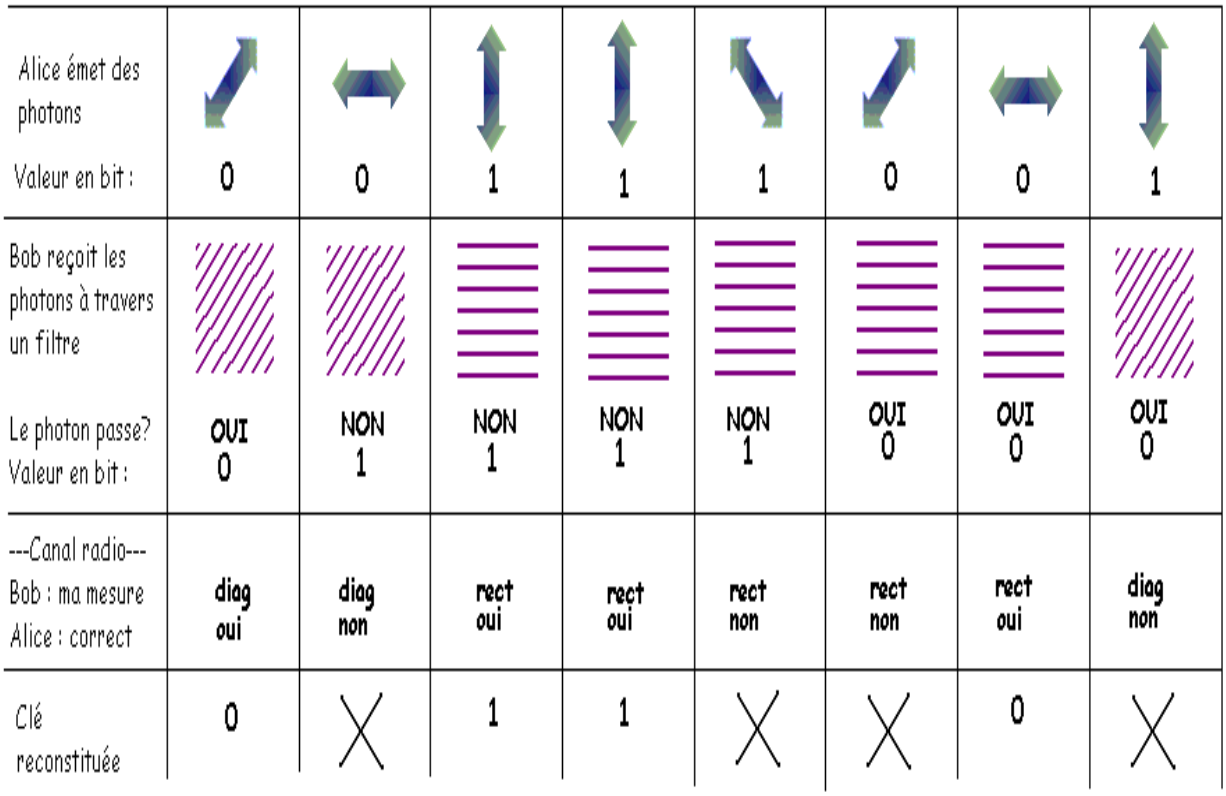


Figure : III.4 étapes de transmission de la clé.

Si un photon polarisé à 0° rencontre un filtre polarisant orienté à 0°, il le traverse et il est enregistré par le détecteur placé juste après. Si un photon polarisé à 90° rencontre le même filtre, il est stoppé et le détecteur n'enregistre rien. Maintenant, si le photon est polarisé diagonalement (à 45° ou 135°), une fois sur deux, il traverse le filtre, et une fois sur deux, il est stoppé. De la même façon, on peut utiliser un filtre polarisant orienté à 45° qui laissera passer les photons polarisés à 45°, stoppera ceux polarisés à 135°, et se comportera aléatoirement avec ceux polarisés à 0° ou 90°.

Cette incertitude quantique garantit l'inviolabilité de la communication. A l'issue de la transmission, les deux partenaires font le point sur les photons émis (polarisation) et les photons reçus (réglage du filtre). Il devient alors possible de déterminer statistiquement le

taux d'erreurs du destinataire dans la réception des photons, erreurs dues au mauvais choix d'alignement mais également à l'imprécision quantique naturelle.

Un intrus extérieur sur la ligne est facilement détecté. Si aucun intrus n'a été détecté, les photons reçus, associés à leur bit d'information (0/1), constituent la clé permettant de chiffrer un document à l'aide de n'importe quel algorithme. On notera que cette clé est jetable (elle ne peut être utilisée qu'une seule fois) et totalement aléatoire. Elle convient donc parfaitement à maintenir la sécurité de la transmission.

III. 1. 2. Principe général d'un protocole de cryptographie quantique

Dans l'utilisation d'un protocole de cryptographie quantique, deux interlocuteurs distants que nous appellerons par commodité Alice et Bob disposent :

- d'objets quantiques, c'est-à-dire d'objets physiques qui se comportent selon les lois de la physique quantique. En pratique, ces objets sont des impulsions lumineuses (photons), qui peuvent prendre plusieurs formes : photons uniques, états cohérents, paires de photons intriqués, etc.
- d'un canal quantique.
- d'un canal classique de communication

Alice code une information aléatoire sur chaque impulsion lumineuse et l'envoie à son correspondent Bob par le canal quantique. Celui-ci mesure alors l'information que porte l'impulsion qu'il a reçue. A la fin de la transmission, Bob possède donc un ensemble de mesures qui sont corrélées aux données envoyées par Alice.

Entre Alice et Bob, il peut se trouver un espion que nous appellerons Eve (abréviation du mot anglais Eavesdropper qui signifie espion), qui a accès à tout ce qui transite entre Alice et Bob, classique ou quantique, et qui n'est limité que par les lois de la physique. En revanche, Eve ne peut pas accéder aux systèmes d'Alice et Bob, qui sont supposés physiquement sécurisés.

III. 1. 2. 1. Détection de l'espion et preuves de sécurité

L'une des propriétés fondamentales de la cryptographie quantique est la capacité des deux interlocuteurs à détecter la présence de l'espion, mais aussi à évaluer précisément la quantité d'information que celui-ci a interceptée.

Ceci résulte de deux aspects fondamentaux de la mécanique quantique : ce sont le principe d'incertitude de Heisenberg et le théorème de non-clonage [Coh .99].

- le principe d'incertitude de Heisenberg stipule que réaliser une mesure sur un objet quantique perturbe généralement l'objet en question.
- le théorème de non-clonage découle du principe d'incertitude. Il stipule qu'il est impossible de dupliquer un objet quantique inconnu. C'est à dire que l'on ne peut pas obtenir une copie identique d'un état aléatoire de quantum.

Si Eve cherche à obtenir de l'information sur l'état de l'objet qui transite par le canal quantique, elle introduit donc des anomalies (bruit ou erreurs), qui peuvent être détectées par Alice et Bob.

Il est également possible d'établir formellement un lien entre la quantité d'anomalies et la quantité d'information interceptée par Eve, grâce à des démonstrations mathématiques appelées preuves de sécurité, qui combinent les lois de la physique quantique et de la théorie de l'information.

III. 1. 2. 2. Transmission de la clé

La clé à transmettre est une série de bits aléatoires, prenant comme valeur 0 ou 1. L'émetteur de la clé code chaque bit de la clé selon l'un des deux modes de polarisation suivants :

- mode 1 : le bit "0" est codé par un photon d'axe de polarisation 0° et le bit "1" par un photon d'axe de polarisation 90°.
- mode 2 : le bit "0" est codé par un photon d'axe de polarisation 45° et le bit "1" par un photon d'axe de polarisation 135°.

L'émetteur émet la clé bit par bit, photon par photon à intervalle régulier, en choisissant aléatoirement le mode de polarisation pour chaque photon émis. L'émetteur note pour chaque bit le mode de polarisation choisi.

Le récepteur utilise le filtre polarisant, pouvant être orienté à 0° (mode 1) ou à 45° (mode 2) et avant l'arrivée prévue d'un photon, il le positionne aléatoirement à 0° ou à 45°. Au moment de l'arrivée du photon, il note le résultat (le photon a passé le filtre ou n'a pas passé le filtre), ainsi que l'orientation choisie du filtre.

Pour chaque bit reçu, deux cas d'interprétation sont possibles :

- l'émetteur et le récepteur ont choisi, par hasard, le même mode de polarisation. Cela se produit en moyenne une fois sur deux. Dans ce cas, le photon reçu est représentatif du bit émis et peut être traduit directement en bit.

- l'émetteur et le récepteur ont choisi une orientation différente et dans ce cas, le photon reçu est parfaitement aléatoire et ne contient aucune information.

III. 1. 2. 3. Information secrète résiduelle

Alice et Bob évaluent tout d'abord le niveau d'erreurs et de bruit séparant leurs deux ensembles de données. Les différences entre leurs données peuvent provenir :

- De l'intervention d'Eve, qui rajoute des erreurs et du bruit
- des erreurs et du bruit de fond, qui ne peuvent jamais être évités complètement.

Néanmoins, puisque les erreurs de communication et les effets de l'observation d'Eve ne peuvent pas être distingués, Alice et Bob doivent supposer que toutes les incohérences sont dues à l'action d'un espion.

Ensuite, grâce aux preuves de sécurité, Alice et Bob peuvent évaluer la quantité d'information notée I_E qui a été interceptée par Eve. En parallèle, la théorie de l'information leur permet d'évaluer la quantité d'information notée I_{AB} qu'ils partagent après la transmission.

Finalement, si la quantité d'information $\Delta I = I_{AB} - I_E$ reste supérieure à zéro, c'est-à-dire que le niveau d'espionnage reste en dessous d'un certain seuil, alors une clé secrète de taille maximale ΔI peut être extraite de la transmission. Dans le cas contraire, aucune extraction n'est possible, et l'échange doit donc être interrompu.

III. 2 Protocoles quantiques de génération de clés

Avant tout, il faut préciser qu'il s'agit bien de protocoles de génération et non pas de distribution de clés. En effet, la clé est créée par le protocole car elle n'existe pas avant. Chaque protocole contient généralement trois phases [Amo.07]:

- Transmission des qubits et réconciliation des bases.
- Réconciliation des clés.
- Distillation de secret.

Sur cette base, plusieurs protocoles ont été développés, ce sont :

III. 2. 1. Le protocole BB84

C'est le premier protocole et le plus connu pour la cryptographie quantique. A sa publication par Bennett et Brassard en 1984 [Ben.84] ,le protocole BB84 était décrit en utilisant la polarisation de photons individuels. La toute première démonstration en 1992 fut d'ailleurs réalisée avec la polarisation et sur une distance de 30 cm à l'air libre [Bes.92] .Une deuxième réalisation de ce protocole en 1996 [Jac.96] porte la distance à 30 kms toujours à l'air libre.
C'est un protocole à quatre états non-déterministe, ce qui signifie qu'il distribue une suite aléatoire de bits. Il ne peut donc pas être utilisé pour la transmission d'un message déterminé. Il est à la base de tous les protocoles qui ont été développés par la suite.

III. 2. 2. Le protocole à deux états (protocole B92)

Le protocole BB84 utilise deux bases conjuguées soit quatre états pour coder les bits. Bennett [Ben.92] montre en 1992, que seulement deux états non-orthogonaux sont suffisants, c'est le principe du protocole B92 qui utilise une seule base non orthogonale. Dans la vérité, la sécurité des bases de cryptographie quantique repose sur l'incapacité d'un espion de

distinguer sûrement et sans perturbation les états différents qu'Alice envoie à Bob, par conséquent deux états sont suffisants s'ils sont incompatibles (c.-à-d., non mutuellement orthogonaux). Toutefois en pratique, ce protocole n'est pas vraiment efficace. En effet, bien que deux états non-orthogonaux ne puissent pas être distingués clairement sans perturbation, on peut les distinguer clairement au coût d'une certaine perte.

III. 2. 3. Le protocole à trois états

Ce protocole constitue une amélioration du protocole BB84. Le protocole BB84 est symétrique dans son utilisation de polarisation. Après la génération de la clé, il est nécessaire d'échanger d'autres informations pour assurer le secret de la clé. Pour casser la symétrie, le protocole à trois états propose d'employer trois états, au lieu de quatre dans BB84, et trois détecteurs, au lieu de deux pour BB84. Ceci réduit la probabilité d'espionnage pour obtenir de bons états, ainsi qu'il minimise la quantité de l'information utile envoyée par Alice.

III. 2. 4. Le protocole à six états

Tandis que deux états sont suffisants et quatre états sont standard, un protocole à six états (figure III.5) respecte plus la symétrie de l'espace d'état du qubit. Les six états constituent trois bases, par conséquent la probabilité qu'Alice et Bob choisissent la même base est seulement 1/3, mais la symétrie de ce protocole simplifie considérablement l'analyse de sécurité et réduit le gain optimal de l'information de l'espion pour un taux donné d'erreur. Si l'espion mesure tous les photons, il induira une erreur de 33%, en comparaison à 25% dans le cas du protocole BB84

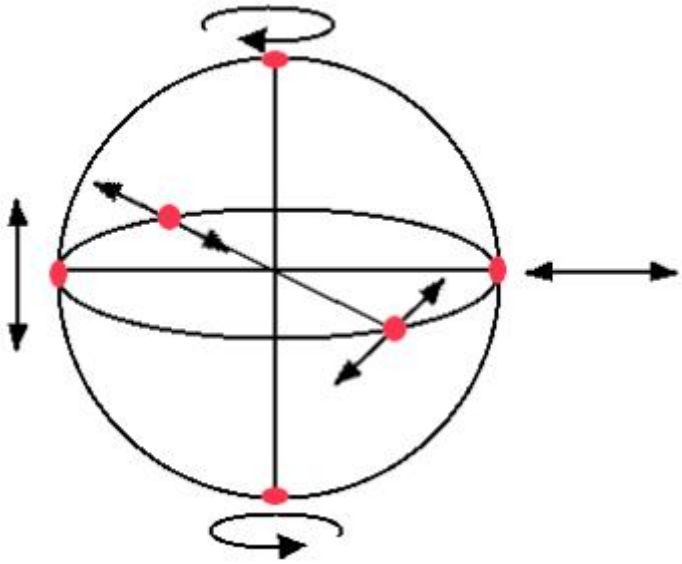


Figure III.5: Trois paires de bases utilisées dans le protocole à six-états

III. 3 Description du Protocole BB84

L'information transmise dans le canal quantique est souvent sous la forme de photons polarisés. Les bits classiques sont codés par des états de quantum en utilisant la direction de la polarisation. Chaque état de quantum peut représenter les deux bits classiques 0 ou 1, et inversement, chaque 0 ou 1 correspond à un mélange de deux états égaux de quantum probablement non-orthogonaux, ce qui nous donne quatre états $|0\rangle$, $|1\rangle$, $|0'\rangle$, $|1'\rangle$.

III. 3. 1. Principe du Protocole BB84

Le protocole BB84 utilise quatre valeurs de polarisation : 0°, 45°, 90°, 135°. Le bit classique 0 est représenté par un photon polarisé à 0º et 45º de l'axe horizontal, et les deux directions orthogonales correspondantes 90º et 135º sont utilisées pour coder le bit 1. Une illustration de ce codage est donnée par la (figure III.6).

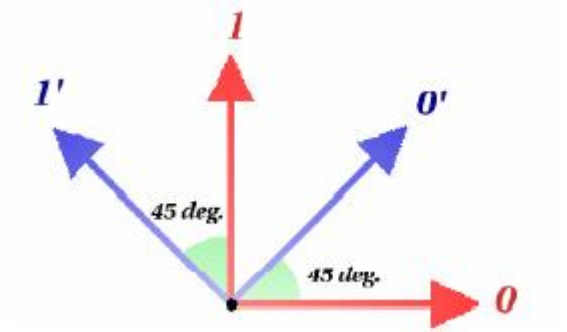


Figure III.6 : Quatre états non-orthogonaux utilisés dans le protocole BB84.

Selon la mécanique quantique, il n'y a aucune manière de différencier sûrement deux états non-orthogonaux. Ainsi une mesure de quantum doit être effectuée pour déterminer l'état reçu et obtenir grâce à cela le rendement classique. Le protocole BB84 utilise deux mesures :

- la mesure permettant d'identifier clairement les deux états $|0\rangle$ et $|1\rangle$. Cette mesure s'appelle la mesure dans la base rectiligne ⊕.
- la mesure permettant d'identifier clairement les deux états $|0'\rangle$ et $|1'\rangle$. Cette mesure s'appelle la mesure dans la base diagonale ⊗.

De façon générale, l'échange de clé quantique avec le protocole BB84 se compose des six étapes [Ngu 05] suivantes:

- Transmission de quantum
- Annonces de bases
- Estimation d'erreurs
- Réconciliation
- Confirmation
- purification

III. 3. 1. 1. Transmission de Quantum

Cette phase correspond à la première étape dans une distribution de clé quantique. Dans cette phase, une chaîne aléatoire de *n* bits classiques codés par une base non déterministe est créée par Alice et envoyée à Bob. Bob reçoit le qubit et il prend par hasard une base rectiligne (ou diagonale) pour le mesurer. Quand la transmission est finie, Bob obtiendra une chaîne de bits classiques, appelée clé crue, différente de celle d'Alice en beaucoup de positions. La prochaine étape du protocole aidera à remplacer les bits non-corrélatifs entre la chaîne d'Alice et celle de Bob, qui sont probablement des erreurs provoquées par l'espion ou la transmission bruyante de quantum.

III. 3. 1. 2. Annonce De Bases

Dans cette phase, toutes les positions, où les mêmes bits sont partagés, sont conservées et le reste sera jeté. Premièrement, à l'aide du canal classique, Bob envoie à Alice toutes les bases qu'il avait utilisées pour mesurer la chaîne des qubits reçus. Ensuite, Alice compare cet ordre des bases avec le sien et révèle toutes les positions non-corrélatives à Bob, toujours sur le canal classique. Enfin, Alice et Bob enlèvent tous les bits aux positions informées par Alice, et on obtient une clé appelée clé plaine ou tamisée identique à Alice et à Bob.

III. 3. 1. 3. Estimation d'Erreurs

Pour réduire la différence entre la clef plaine d'Alice et de Bob due à l'imperfection d'appareil, il est nécessaire de corriger les erreurs. L'émetteur et le récepteur doivent calculer le taux d'erreurs observées et gardent cette transmission si le taux d'erreur est inférieur à un seuil désiré. Dans le cas où le taux d'erreur est supérieur au seuil, la clé sera avortée.

III. 3. 1. 4. Réconciliation

Alice et Bob vont devoir utiliser un algorithme de correction d'erreurs, car il est primordial pour eux de posséder la même clé.

Un exemple d'algorithme simple est le suivant : Alice choisit aléatoirement des paires de bits et divulgue publiquement le numéro de ces bits ainsi que leur somme XOR. Si Bob obtient le même résultat, ils gardent le premier bit de la paire et jettent le second. Dans le cas contraire, ils jettent les deux bits. Cet algorithme permet de réduire le taux de différence entre les clés d'Alice et de Bob.

Un autre type d'algorithme qui peut être utilisé est l'algorithme Cascade [Bra.94] développé par Gilles Brassard et Louis Salvail. Cet algorithme peut également être utilisé dans l'étape de distillation de secret. Il fonctionne avec un certain nombre de rondes.

III. 3. 1. 5. Confirmation

Afin de s'assurer qu'aucune erreur ne sera trouvée, Alice et Bob échangeront et compareront la parité des sous-ensembles aléatoires de positions. En général, si une comparaison à *z* bits de parité est faite et il n'y a aucune différence, alors la clé partagée courante est identique au taux de 2^{-z}. Si cette phase est réussie, on peut estimer que la clef partagée est maintenant la même. Dans le cas où l'erreur persiste, elle va apparaître à un taux acceptable si *z* est assez grand.

III. 3. 1. 6. Purification ou distillation de secret

L'information sur la clé corrigée que possède Eve est nuisible pour Alice et Bob car elle augmente ses chances de briser le chiffrement à clé privée utilisé avec la clé obtenue. Pour régler ce problème, les auteurs de BB84, en collaboration avec d'autres chercheurs, développèrent une technique permettant de diminuer à un niveau arbitrairement faible l'information de l'espion [Ben.88, Ben.95]. Cette technique se nomme distillation de secret (privacy amplification en anglais).

Considérons à nouveau le cas le plus simple : Alice choisit des paires de bits dont elle prend la somme XOR, mais cette fois ci, elle divulgue seulement le numéro des bits. Alice et Bob remplacent simplement la valeur de chacun des bits par leur somme XOR. Ainsi, ils n'engendrent pas de nouvelle différence entre leur clé et réduisent l'information d'Eve. En effet, si Eve ne connaît que la valeur du premier bit mais pas celle du deuxième, elle n'a aucune information sur leur somme XOR.

III. 3. 2. Quelques types d'attaques contre BB84

Deux types d'attaques considérées comme étant les plus fréquentes et les plus significatives ont été étudiées [Bus.03] .Ce sont l'attaque Interception-Renvoi (I-R) et l'attaque séparation du nombre de photons (SNP).

III. 3. 2. 1. Attaque Interception-Renvoi (I-R)

Dans l'attaque Interception-Renvoi, Eve intercepte une fraction des qubits à la sortie du laboratoire d'Alice et tente de déterminer leur état en choisissant judicieusement sa mesure. Elle communique ensuite son résultat à son complice Fred, lequel renvoie l'état correspondant directement dans le laboratoire de Bob (figure III.7).

Figure. III.7 : Attaque Interception-Renvoi

Dans cette situation, Eve réussit à identifier correctement l'état une fois sur deux et n'obtient aucune information le reste du temps. Si Fred renvoie toujours l'état identifié par Eve, elle se trompe 1 fois sur 2, auquel cas l'état cause une erreur chez Bob avec une probabilité 1/2. La probabilité que cette stratégie cause une erreur dans la clé tamisée est donc de 25% par tentative. Bien entendu, Fred ne devrait pas toujours renvoyer l'état dans une seule base, car Bob pourrait facilement s'en rendre compte.

III. 3. 2. 2 Attaque séparation du nombre de photons (SNP)

Ce type d'attaque n'est possible que sur des systèmes utilisant les photons. Ces systèmes utilisent en général des sources laser, dont l'utilisation peut causer une brèche dans la sécurité qui, selon les capacités technologiques de l'espion, peut être sévère ou non.
la distribution du nombre de photons dans une impulsion, x, est poissonnienne :

$$F(x) = \frac{\mu^x e^{-\mu}}{x!}$$

Ou μ est le nombre moyen de photons par impulsion. Dans l'attaque SNP, on suppose qu'Eve n'est limitée que par les lois de la physique. En particulier, on lui donne la possibilité de réaliser une mesure dite non destructive du nombre de photons dans l'impulsion, sans toutefois modifier l'observable dans laquelle le qubit de BB84 est codé. Eve doit également être capable de réaliser une interaction dite < séparation du nombre de photons >, (SNP), qui consiste à séparer de façon déterministe un des photons de l'impulsion [LAU 2000]. Elle doit aussi posséder un canal de transmission Sans pertes. Un tel canal pourrait être réalisé, par exemple, par la téléportation Quantique [BEN 93]. La dernière condition, et non la moindre, est qu'Eve possède une mémoire quantique de durée arbitrairement longue. Dotée de ces outils, l'attaque suivante devient possible :

1. A la sortie du laboratoire d'Alice, Eve détermine le nombre de photons dans l'impulsion a l'aide d'une mesure non-destructive.
2. Si le nombre de photons est supérieur a 1, elle en conserve un a l'aide de l'interaction SNP et téléporte les autres a l'entrée du laboratoire de Bob. Sinon, elle a deux possibilités. Soit qu'elle téléporte simplement le photon a l'entrée de Bob, soit qu'elle applique l'attaque I-R et passe ensuite a l'impulsion suivante.

3. Eve conserve le photon séparé dans sa mémoire quantique jusqu'a l'annonce publique de la base de préparation.
4. Connaissant la base de préparation du photon, Eve mesure son état et découvre la valeur du bit d'Alice à coup sûr .

Donc, lorsque l'impulsion contient deux photons ou plus, le gain d'information est de 1 bit pour une probabilité d'erreur induite égale à 0. Sinon, le gain est celui de l'attaque I-R, avec une probabilité d'erreur induite de 1/4.

III. 3. 3 Preuve de sécurité du protocole

Les deux types attaques que nous avons décrits ne constituent probablement pas une démonstration générale de la sécurité de BB84. Elles font partie d'une catégorie d'attaques dites individuelles, où l'on permet à l'espion de manipuler les qubits individuellement uniquement et d'une catégorie dites cohérentes correspondant à la situation où l'espion manipule simultanément un grand nombre de qubits de façon cohérente. Cependant, depuis 1998, plusieurs preuves ont été apportées à la sécurité de ce protocole [Cha.99, Sho. 2000, Bih.2000, Ina. 01, Koa.03].

En particulier, la preuve de H. Inamori, N. Lutkenhaus et D. Mayers [Ina.01] en 2001, affirme qu'il est possible d'implanter le protocole BB84 de façon sécuritaire même si le montage comporte certaines imperfections, en l'occurrence une transmission inférieure à 1, des détecteurs inefficaces et bruyants et une source émettant des impulsions multi-photons. Ils supposent cependant qu'Alice est en parfait contrôle de la source mais que les détecteurs de Bob ont un bruit et un rendement inconnus et possiblement sous le contrôle de l'espion.

En 2003, M. Koashi et J. Preskill [Koa.03] ont démontré que dans la situation où la source n'est pas caractérisée mais que les détecteurs sont parfaits, alors le protocole BB84 est toujours réalisable de façon sécuritaire.

Conclusion

Le protocole BB84 utilise la mécanique quantique ainsi que la théorie de l'information pour vérifier la présence d'un espion. Il utilise aussi le code de Vernam pour s'assurer que le message échangé est indéchiffrable sans la clé. On a vu qu'il existe diverses techniques d'espionnage. Néanmoins, ces techniques ne permettent pas à un espion de rester indétectable. Dans ce qui suit on va essayer de réalisé un simulateur du protocole BB84.

Chapitre IV

Simulation du protocole BB84

Simulation du protocole BB84

Introduction

Le but de ce chapitre est de présenter une simulation du protocole de distribution de clés quantique (DCQ) en utilisant le protocole BB84. Le simulateur simule le procédé d'échange de clé entre deux acteurs principaux Alice, Bob et un intrus Eve. L'échange est réalisé sur deux canaux, un canal quantique et un canal classique, qui sont considérés en tant que deux autres acteurs du système. Le logiciel est implémenté en Java

IV.1 Description des Fonctions

Alice

Alice est l'expéditeur du message chiffré. Elle doit communiquer avec Bob pour produire une clé aléatoire. Ceci est fait en suivant séquentiellement toutes les étapes du protocole BB84 décrites dans le chapitre 3. Elle prépare la clé sous la forme d'une chaîne de qubits aléatoires et informe Bob du début et de la fin du message de chaque impulsion. Ensuite, elle écoute le canal pour déterminer quand Bob finit sa détection d'une impulsion.

Une fois tous les qubits envoyés et reçus, Alice communiquera avec Bob les bases pour coder les bits envoyés, les positions des bits qu'elle utilise dans la phase de correction d'erreurs et de purification. Sa communication avec Bob est sur le canal quantique et le canal public. La tâche finale d'Alice est d'afficher tous les résultats de ses actions consécutives pour le but de la démonstration. La démonstration fournit aux utilisateurs une interface pour modifier des paramètres affectant le protocole, tel que le nombre de photons dans une impulsion ou la taille désirée pour l'évaluation des erreurs.

Bob

Bob est le partenaire de Alice dans ce protocole. Il est le destinataire de son message. Certaines de ses fonctions, comme écouter le canal, lire et écrire l'information sur le canal sont les mêmes qu'avec celles d'Alice. La phase d'Annonce de Bases est cependant différente. Elle est constituée de toutes les bases aléatoirement choisies par son détecteur de photons pour mesurer les impulsions reçues.

Eve

Eve se tient entre Alice et Bob. Elle joue leurs deux rôles en même temps. Ainsi, elle partage avec eux beaucoup de fonctions communes afin d'exploiter BB84. Sa motivation principale est de rassembler des informations sur la clé partagée par Alice et Bob sans être découverte.

Le canal quantique

L'objectif de la simulation est de faire une représentation du canal quantique le plus identique possible au canal de quantum réel. Cela signifie que celui-ci doit avoir toutes les propriétés physiques mentionnées dans le chapitre 3. Le canal cause parfois des erreurs à la communication, telles que la perte d'impulsions dans la transmission ou la modification de la valeur d'un qubit.

Le canal public

Le simulateur doit permettre aux utilisateurs de voir le déroulement des différentes phases d'annonce de bases, de correction d'erreur et d'amplification de confidentialité.

IV .2 Implémentation du protocole BB84

Maintenant, nous entrons dans la partie principale de cette section, l'implémentation du protocole DCQ BB84. Le protocole est appliqué par trois acteurs : Alice (expéditeur), Bob (destinataire) et Eve (espion). Toutes leurs communications sont effectuées dans les deux canaux séparés. L'utilisation de Java pour développer le programme fait référence à plusieurs classes :

Classe ConfConstants

Les attributs de cette classe sont la taille de la clé initiale, le taux d'erreur et le pourcentage de la clé tamisée à comparer.

Classe Alice

Cette classe représente l'émetteur qui génère les bits aléatoires constituant la clé crue ainsi que les bases à appliquer. L'envoi de l'information sur le canal quantique se fait sous forme d'instances de la classe de qubit (figure IV.1). Un objet qubit contient la base avec laquelle le bit a été codé (0 pour la base rectiligne et 1 pour la base diagonale) et sa polarisation (horizontale, verticale, diagonale, antidiagonale)

Qubit
Int Value
Int Base
Int Polarisation

Figure IV.1 : Classe qubit

Classe Bob

Cette classe représente le récepteur. Elle contient des méthodes qui permettent de générer des bases aléatoires de la même façon que l'émetteur et de récupérer par la suite la séquence de qubits reçus pour obtenir la clé tamisée.

Canal quantique

C'est le canal utilisé pour la transmission des états quantiques. Sa simulation fait appel à deux classes, InitServer et RespServer

- **Classe InitServer**

Server Initiator est responsable sur le canal quantique du côté initiateur. Il transmet les qubits à travers le canal quantique et les renvoie au client de l'initiateur.

- **Classe RespServer**

Responder Server est responsable de la partie répondeur sur le canal quantique. Il reçoit les qubits envoyés par le serveur initiateur et les renvoie au client de répondeur.

Classe InitPlayer

Initiator player a pour rôle de lancer l'exécution du protocole quantique et de choisir le protocole de collaboration avec les paramètres à utiliser. Cette classe établit également une connexion avec le serveur initiateur

Classe Respplayer

Responder Player est celui qui répond à la demande et donc connecté au client de l'initiateur et ou server du répondeur. Le client du répondeur est généralement sur la même machine avec le serveur du répondeur.

Classe canal public

Cette classe contient différentes méthodes qui modélisent l'échange de données nécessaire afin de distiller une clé finale secrète entre Alice et Bob.

Classe canal privé

Ce canal n'est pas nécessaire pour la distribution des clés quantiques, il est utilisé pour la connexion entre le client et le serveur.

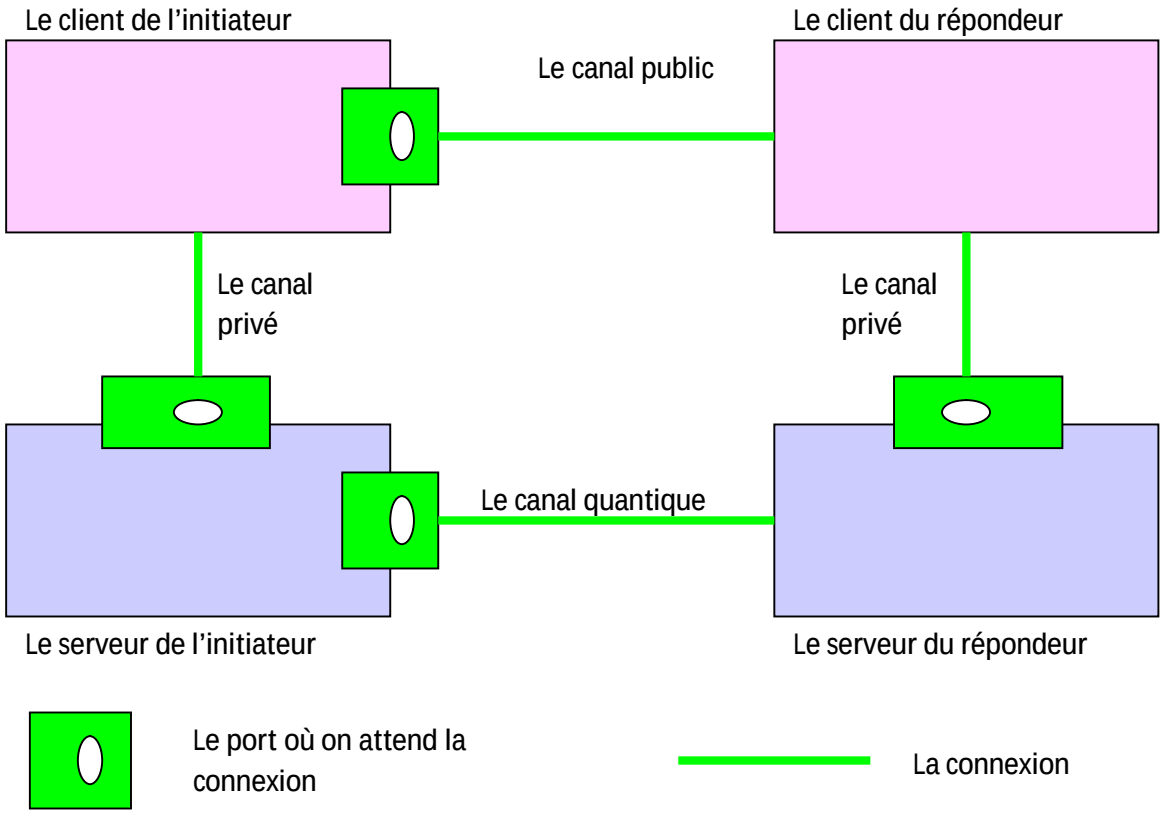
Avec toutes ces classes et leurs caractéristiques, le logiciel présente une architecture à deux niveaux, le niveau serveur et le niveau client.

Le tableau ci-après présente les rôles de chaque élément.

Les éléments	Les rôles
Le serveur de l'initiateur	• Créer des bits quantiques • créer et échanger la clé crue
Le serveur du répondeur	• recevoir et envoyer la clé crue à son client
Le client de l'initiateur	• Initier la session de travail. Réaliser les tâches restantes : • L'estimation de l'erreur • La réconciliation • L'amplification
Le client du répondeur	

Les entités et les relations entre elles sont présentées ci-dessous :

- Ø Le serveur de l'initiateur a deux ports :
  - Un port pour attendre la connexion à partir du serveur du répondeur
  - Un port pour attendre la connexion à partir du client de l'initiateur
- Ø Le serveur du répondeur a un port pour attendre la connexion à partir du client du répondeur
- Ø Le client de l'initiateur a un port pour attendre la connexion à partir du client du répondeur



Dans cette partie, nous avons procédé à la spécification des besoins d'implémentation du simulateur BB84, ainsi qu'a la conception détaillée de ses différentes fonctionnalités. Ensuite nous avons défini les différentes classes pour pouvoir passer à la description du simulateur dans la partie qui suit.

IV.3 description du simulateur

nous allons présenter le résultat de la simulation en décrivant dans un premier temps les différentes interfaces graphiques ainsi que les champs de données a saisir par l'utilisateur pour lancer la simulation.

Lors du démarrage du simulateur, une interface graphique apparaît (figure IV.2)

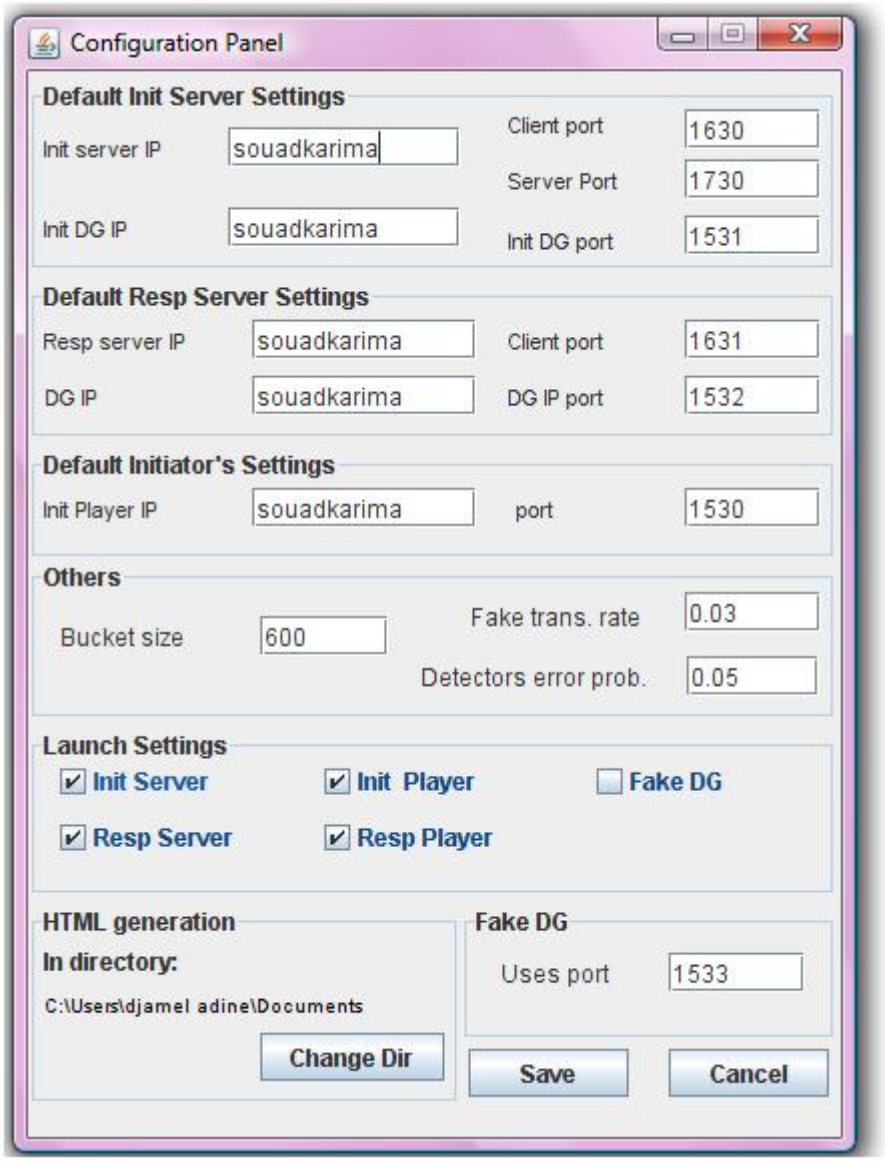
Figure IV.2 : réglage des paramètres du protocole.

Les paramètres du protocole sont :

Settings

L'utilisateur sélectionne les quatre cases : L'**initiator player** (client de l' initiateur) et l'**initiator server** (serveur de l'initiateur) sont les deux entités qui représentent Alice alors que le **responder player** (client du répondeur) et le **responder server** (serveur du répondeur) sont les entités qui représentant Bob :

Le bouton **Edit configurations** permet de générer un fichier de configuration qui va indiquer les paramètres utilisés par défaut à chaque lancement de qucrypt. La fenêtre de configuration se présente comme suit :



Cette fenêtre permet de définir par défaut des valeurs nécessaires pour le lancement de chaque entité. Plus précisément, les valeurs suivantes peuvent être re-définies:

- Default Initiator Server Settings, Default Responder Server Settings, Default Initiator Player Settings permettent de définir les valeurs par défaut pour le serveur de l'initiateur, le serveur du répondeur, le client de l'initiateur

- **Others** permet de régler le nombre de qubits transmis sur le canal quantique. La valeur 0,03 est le taux d'erreur.

- **Detectors error prob** indique la probabilité d'erreur des détecteurs simulés.

- **HTML generation** permet de sélectionner le répertoire qui contient l'initiateur et le répondeur pour la génération de code HTML.

- **Launch Settings** permet de définir les paramètres par défaut de la fenêtre du Lanceur QuCrypt.

Une fois que le fichier de configuration a été créé, QuCrypt doit être lancé à nouveau. Puis on tape sur le bouton **Start** pour lancer la simulation. On aura alors les quatre fenêtres représentées par les (figures IV. 3 et IV.4).

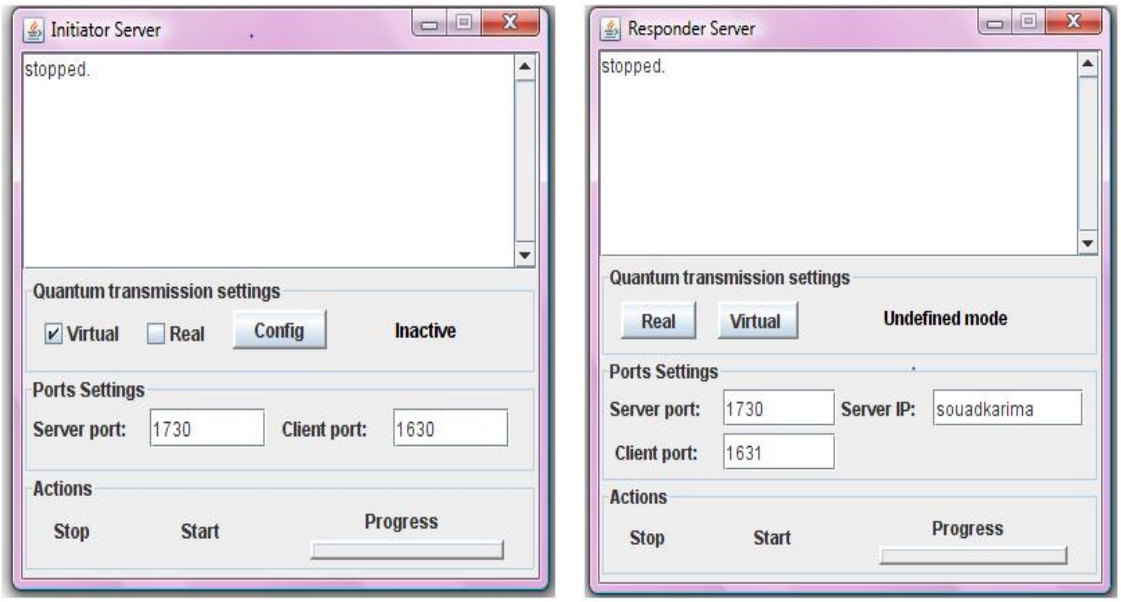


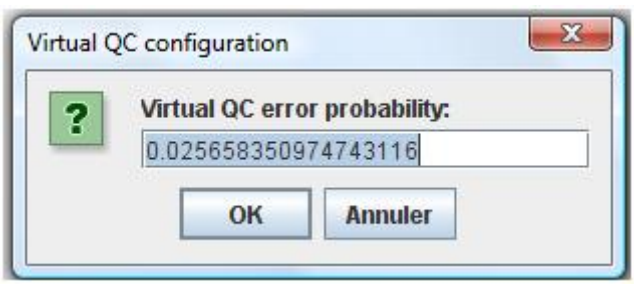
Figure IV.3:initiator server,responder server

Ø **Fenêtre initiator server :** elle affiche des informations à l'utilisateur que le serveur est en marche. Cette fenêtre contient trois groupes d'éléments :

Quantum transmission settings *(*Paramètres *de transmission quantique)*

- **Virtual**, **Real** : indique si la prochaine transmission est à travers un canal quantique simulée, ou par l'expérience réelle
- **Config** : permet la configuration des modes réel ou virtuel de transmission quantique.

Virtual configuration permet de définir le taux d'erreur pour la prochaine transmission quantique.



Ports settings (paramètres des ports)

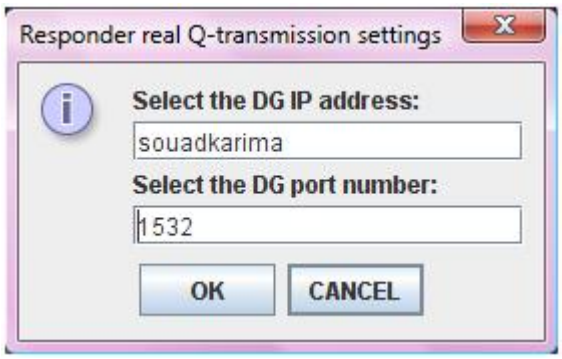
- **Server Port** est le numéro de port où le serveur de l'initiateur attend une connexion avec le serveur du répondeur.
- **Client Port** est le numéro de port où le serveur de l'initiateur attend une connexion avec le client de l'initiateur.

Actions

- **Start et Stop** sont utilisées pour démarrer ou arrêter le serveur.
- **Progress bar** indique l'état d'avancement lors de la transmission quantique.

Ø **Fenêtre Responder Server:** elle contient les éléments suivants:

- **Virtual button** qui permet de définir le taux d'erreur pour la transmission quantique.
- **Real button** qui permet de sélectionner l'adresse IP et le numéro de port pour la connexion au canal quantique.



Ports Settings

- **Server IP** est l'adresse IP du serveur de l'initiateur. Lorsque le serveur du répondeur est lancé, il essaie de se connecter au serveur de l'initiateur .
- **Server Port** indique le numéro de port sur lequel le serveur répondeur se connecte à l'initiateur de serveur .
- **Client Port** est le numéro de port où le serveur répondeur attend une connexion avec le client du répondeur.

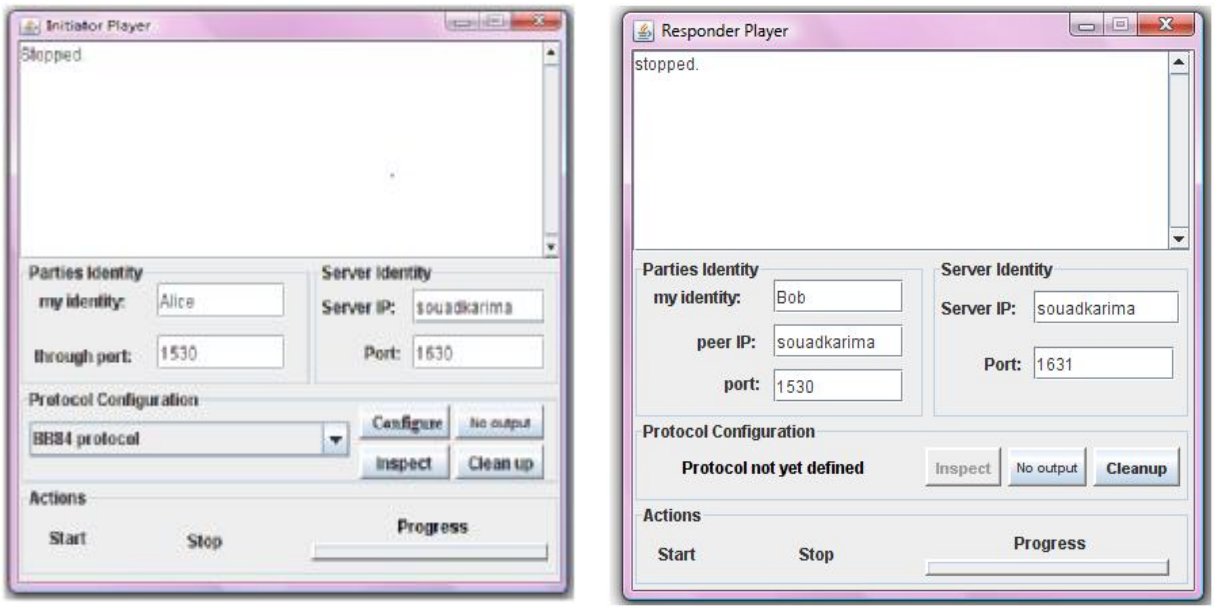


Figure IV.4 : initiater player, responder player

Ø **Fenêtre Initiator player:** elle contient 4 groupes d'éléments

Parties Identity

- **My identity** est une chaîne pour l'identification du client de l'initiateur, généralement appelée Alice.

- **Through port** est le numéro de port sur lequel le client du répondeur doit se connecter avec le client de l'initiateur.

Server Identity (Identité *du serveur)*

- **Server IP** est l'adresse IP pour le serveur de l'initiateur .
- **Port** est le numéro de port sur lequel le client de l'initiateur va se connecter au serveur de l'initiateur .

Protocol Configuration

- **Protocol's selection list** (Liste de sélection Protocole**)** représente la liste des protocoles disponibles pour la prochaine exécution. Il permet au client de l'initiateur de sélectionner un protocole parmi une liste des protocoles. Dans notre cas, on va choisir le protocole BB84.

- **Protocol's configuration button**. (Bouton de configuration Protocole**)** permet de configurer le protocole BB84 sélectionné pour la prochaine exécution. Les valeurs seront envoyées au client du répondeur. la (figure IV.5) ci-dessous nous permet de saisir les paramètres de configuration du protocole BB84.

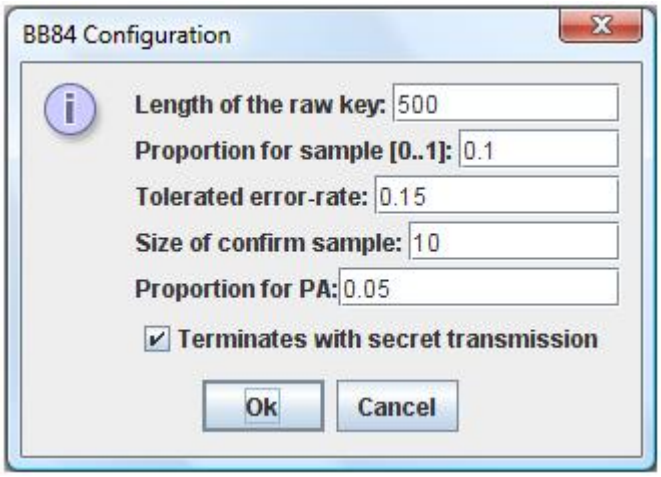


Figure IV.5 configuration du protocole

Alice est chargée d'installer la configuration de l'exécution du protocole.

-Length of the raw key (longueur *de la clé initiale)* est la longueur de la clé avant que les bases ne soient annoncées.

-proportion for sample *(nombre moyen de photons par impulsion)* est un paramètre contenu dans l'intervalle [0…1].

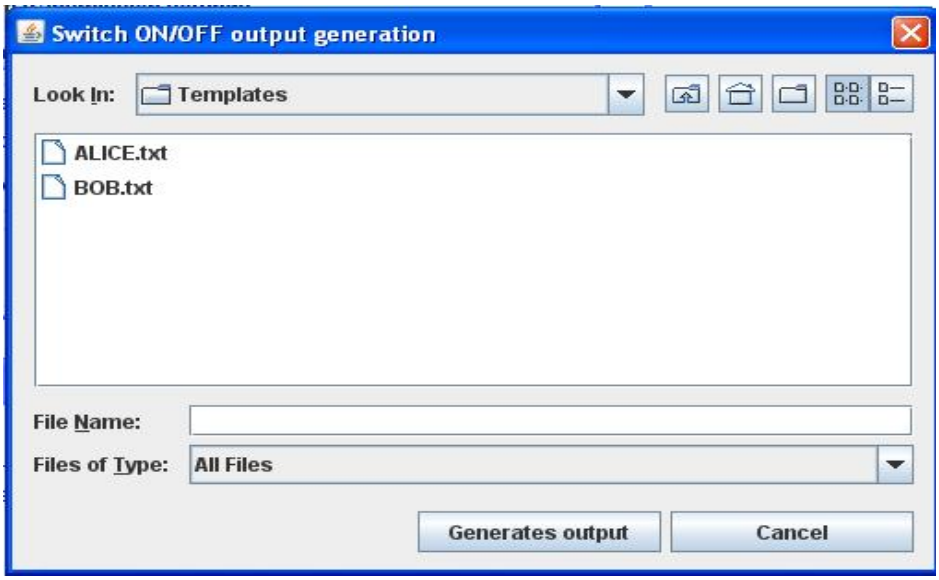
-Tolerated error-rate indique que le taux d'erreur est 0.15

-Size of confirm sample *(*taille *de l'échantillon de confirmation),* 10 sous-ensembles aléatoires seront ramassés et leur parité échangés.

-Proportion for PA indique que la valeur de l'amplification de confidentialité est 0.05

- **Inspect button** permet de voir les valeurs fixées pour tous les paramètres du protocole sélectionné. Cela ne permet pas de les modifier.

- **Output selection button** permet d'activer ou désactiver la génération de la sortie HTML. La sortie de l'exécution sera disponible à partir du fichier d'index du répertoire Template.



- **Clean up button** permet de supprimer certaines exécutions du fichier d'index du répertoire.

Actions

- **start/stop button**: *Start* doit être enfoncée après que le protocole a été sélectionné et paramétré. Il réalise les opérations suivantes:
  - Se connecter au serveur de l'initiateur
  - Attendre que le client du répondeur se connecte
  - Lors de la connexion, le protocole choisi est envoyé au client du répondeur
  - Si le protocole est accepté par le client du répondeur alors le client de l'initiateur demande au serveur de l'initiateur de lancer une transmission quantique.
  - Le protocole est exécuté.
  - Après l'exécution le lecteur s'arrête automatiquement.

- **progress bar** (barre de progression) indique l'état d'avancement et la phase du protocole en cours d'exécution.

Ø **Fenêtre Respondeur Player** est le client qui répond à la connexion du client de l'initiateur et le serveur du répondeur. Le client du répondeur est généralement sur la même machine que le serveur du répondeur.

Une fois que les éléments sont connectés, Alice demande à Bob s'il accepte le protocole BB84 pour la génération de clé quantique.

Si Bob accepte (yes), le protocole est en cours d'exécution. La sortie de l'exécution sera disponible à partir du fichier d'index du répertoire Template. Ce répertoire permet de visualiser les étapes qui se déroulent sur le canal quantique et publique.

a/ Information circulant sur le canal quantique

**********BB84 protocol Output**********

Dimanche 26/6/2011 .15:59:09

****

Transmission of length: 500

My identity: **Alice**

(0,X) (1,+) (0,X) (1,+) (0,X) (1,X) (0,X) (1,+) (0,X) (0,+)

(0,X) (1,X) (1,+) (0,X) (0,X) (0,X) (1,+) (0,X) (0,+) (0,+)

(1,+) (0,X) (0,X) (0,+) (1,X) (1,X) (0,X) (0,X) (0,+) (0,+)

(0,X) (0,+) (0,+) (0,+) (1,X) (1,+) (0,X) (1,X) (0,X) (1,+)

(1,X) (0,+) (0,+) (0,+) (0,+) (1,X) (0,X) (1,X) (1,X) (1,X)

My identity: **Bob**

(1,+) (1,X) (0,X) (1,+) (0,X) (0,+) (0,X) (1,X) (0,X) (1,X)

(1,+) (1,X) (1,+) (1,+) (0,+) (0,+) (1,+) (0,X) (0,X) (0,+)

(0,X) (0,X) (1,+) (0,+) (1,+) (1,+) (0,+) (0,X) (0,X) (0,+)

(0,X) (0,+) (0,+) (0,+) (1,+) (0,X) (0,X) (0,+) (0,+) (1,+)

(0,+) (0,X) (0,+) (0,+) (0,+) (0,+) (0,X) (1,X) (0,+) (1,+)

Alice : les bits et les bases d'Alice correspondent à un choix aléatoire.

Bob : Bob reçoit une suite de qubits avec des états de polarisations envoyés par **Alice**.

Interceptés par **Eve** et modifiés suite aux erreurs dues au canal quantique, **Bob** applique une suite de bases aléatoires et détermine la séquence binaire correspondante.

Alice et **Bob** possèdent désormais chacun une suite binaire de même taille. Ils doivent ensuite utiliser le canal public pour pouvoir déterminer les bases communes, le taux d'erreur, corriger la clé et traiter dans une phase finale cette clé pour extraire une clé finale que **Eve** ignore.

Pour réduire la différence entre la clé plaine d'Alice et Bob due à l'imperfection d'appareils, Alice va extraire une petite série des bits de la clef plaine pour l'envoyer à Bob. L'émetteur et le récepteur doivent calculer le taux d'erreurs observées.

Après cette phase, une clé réconciliée sera obtenue. Le but de cette phase est de corriger les erreurs, Mais il est important de prendre note que peu de bits en tant que possible sont envoyés car l'espion peut exploiter cette information, pour cela le protocole cascade est appliqué. La cascade effectue de correction d'erreurs en envoyant très peu de l'information à travers le canal public.

Afin de s'assurer qu'aucune erreur ne sera trouvée, Alice et Bob échangeront et compareront la parité des sous-ensembles aléatoires de positions.

Alice et Bob disposent finalement d'une clé commune secrète et sans erreur a propos de laquelle Eve n'a aucune information.

Toutes ces informations circulent sur le canal public et permettent aux utilisateurs de voir le déroulement des différentes phases d'annonce de base, de correction d'erreur et d'amplification de confidentialité visualisé grâce à l'interface illustrée ci-dessous.

b/ informations échangées sur le canal public.

Alice**********

**********Bases announcement**********

x + x + x x x + x + x x + x x x + x + + + x x + x

x x x + + x + + + x + x x x + x + + + + x x x x x

Bob**********

**********Bases announcement**********

+ x x + x + x x x x + x + + + + + x x + x x + + +

+ + x x + x + + + + x x + + + + x + + + + x x + +

Alice**Raw key containing the bits measured in the same bases**

Number of positions kept: 247 (clé tamisée)

0 1 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1

0 1 0 0 0 0 1 1 1 1 1 1 1 1 0 1 1 1 1 0 0 1 1 0 0

Bob**Raw key containing the bits measured in the same bases**

Number of positions kept: 247(clé tamisée)

0 1 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1

0 1 0 0 0 0 1 1 1 1 1 1 1 1 0 1 1 1 1 0 0 1 1 1 0

Alice**********Noise Estimation Results**********

Sample size:50 // Alice va extraire une petite série des bits de la clef plaine pour l'envoyer à Bob.

Observed noise rate:0.02 // le taux d'erreurs observées par l'émetteur et le récepteur.

Number of remaining bits: 200 // clef après estimation d'erreurs.

Remaining bits:

0 1 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0

0 0 1 1 1 1 1 1 1 1 1 0 0 1 0 0 1 0 1 1 0 0 1 0 0

Bob **********Noise Estimation Results**********

0 1 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0

0 0 1 1 1 1 1 1 1 1 1 0 0 1 1 0 1 0 1 0 0 0 1 0 0

Alice**** list of distinct probes****

139 33 46 59 245 185 34 164 158 19 79 166 238 37 84

115 91 119 211 55 20 175 39 116 232 111 81 137 17 11

196 125 65 180 96 171 212 156 127 72 186 246 109 191 225

106 215

****

---Cascade Statistics--- // La cascade effectue la correction d'erreurs.

There were 10 errors found and corrected:

* 2 in pass 0
* 2 in pass 1
* 0 in pass 2
* 6 in pass 3
* 0 in pass 4

The total number of disclosed bits is 71

Exchange of 10 parity bits for confirmation.

Confirmation of identical shared bits was successful.

The bits are identical except with probability: 9.765625

Alice**********

The reconciled key prior privacy amplification is: 200 (clé finale)

0 1 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0

0 0 1 1 1 1 1 1 1 1 1 0 0 1 1 0 1 0 1 0 0 0 1 0 0

Bob**********

The reconciled key prior privacy amplification is:

0 1 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0

0 0 1 1 1 1 1 1 1 1 0 0 1 1 0 1 0 1 0 0 0 1 0 0

A la fin, Alice et Bob possèdent tous les deux une clé commune de taille 200 bits, identique et secrète qui peut être utilisée pour un échange d'information cryptée.

Conclusion générale

La cryptographie quantique est d'ores et déjà une réalité commerciale exploitée par de nombreuses sociétés. Son but n'est pas tant d'assurer la confidentialité des données (il existe pour cela de nombreux algorithmes traditionnels très fiables), mais plutôt de résoudre l'autre grand défi de la cryptographie : l'échange d'une clé de chiffrement sur un réseau public.

Dans ce mémoire, nous avons donné une description détaillée du protocole DCQ BB84, le plus utilisé actuellement dans le domaine de la cryptographie quantique et nous avons présenté une implémentation de ce protocole en utilisant un simulateur construit sur le modèle client serveur.

Ce logiciel fournit un soutien général pour le codage de l'information quantique. Les systèmes de codage quantique pris en charge sont les protocoles BB84 et B92. Un autre type de codage peut facilement être ajouté avec un minimum de travail, permettant ainsi d'agrandir la bibliothèque de primitives quantiques.

Le logiciel peut être amélioré en rajoutant un certain interfaces graphiques, tels que le nombre de qubits/longueur de la clé finale, le taux d'erreurs détectées, le pourcentage de qubits interceptés par l'espion. Cela nous permettra de faire une analyse statistique et retirer les valeurs expérimentales qui seront utiles pour les tests dans la réalité

Bibliographie

[Amo 07] G. Amor, Etude d'un canal à chiffrement quantique : contraintes techniques et faisabilité, mémoire de fin d'études Ingénieur, Ecole Supérieure des Communications, Tunis, 2007.

[Bec 90] B. Beckett, Introduction aux méthodes de la cryptologie, Ed. Masson, 1990

[Ben 84] C. H. Bennett et G. Brassard, Quantum cryptography: public key distribution and coin tossing, Internationale Conference On Computers, Systems & Signal Processing, Bangalore, India, 1984.

[Ben 88] C. H. Bennett, G. Brassard et J. M. Robert, Privacy amplification by public discussion, SIAM Journal of Computing 17, 1988

[Ben 92] C. H. Bennett, Quantum cryptography using any two nonorthogonal states, Physical Review Letters 68, 1992

[Ben 92] C. H. Bennett, F. Bessette, G. Brassard, L. Salvail et J. Smolin, Experimental quantum cryptography, Journal of cryptology, vol. 5, n° 1, 1992

[Ben 95] C. H. Bennett, G. Brassard, C. Crépeau et U.M. Maurer, Generalized privacy amplification, IEEE Transactions on Information Theory 41, 1995.

[Bih 00] E. Biham, M. Boyer, P.O. Boykin, T. Mor et V. Roy-chowdurry, A proof of the security of quantum key distribution, Proceedings of the 32nd Annual ACM Symposium on Theory of Computing (ACM Press, New York), 2000.

[Bra 94] G. Brassard et L. Salvail, Secret-Key Reconciliation by Public Discussion, Proceedings of Eurocrypt 94, 1994.

[Bus 03] F. Bussières, Cryptographie quantique à plusieurs participants par multiplexage en longueur d'onde, mémoire de fin d'études de Master, Université de Montréal, 2003.

[Cha 99] H. K. Lo et H. F. Chau, Unconditional Security of Quantum Key Distribution over arbitrarily long distances, Science 283, 1999.

[Coh 99] C. Cohen-Tannoudji, B. Diu et F. Laloe, Mécanique Quantique, Ed. Hermann 1999.

[Ebr 06] T. Ebrahimi, F. Leprévost, B. Warusfel, Cryptographie et sécurité des systèmes et réseaux, Ed. Hermès, 2006

[Ina 01] H. Inamori, N. Lutkenhaus et D.Mayers, Unconditional security of practical quantum key distribution, 2001. http://arxiv.org/abs/quant-ph/0107017

[Jac 96] B. C. Jacobs et J. D. Franson, Quantum Cryptography in Free Space, Optics Letters 21, 1996.

[Kim 04] T. Kimura, Y. Nambu, T. Hatanaka, A. Tomita, H. Kosaka, and K. Nakamura, Single-Photon Interference Over 150-km Transmission Using Silica-Based Integrated-Optic Interferometers For Quantum Cryptography Criterion, Electronics Letters, 2004.

[Koa 03] M. Koashi et J. Preskill. Secure quantum key distribution with an uncharacterized source, Physical Review Letters 90, 2003.

[Ngh 05] B.T.Nghia, Rapport final du TIPE : cryptographie quantique, Institut de la francophonie pour l'informatique, 2005.

[Ngu 05] Th. M. Nguyen, Etudier et implémenter une simulation du protocole d'échange de clé quantique BB84, stage de fin d'études, Ecole Nationale Supérieure des Télécommunications, Paris, 2005.

[Sch 95] B. Schneier, Cryptographie appliquée, Ed. International Thomson Publishing, 1995

[Sho 00] P. W. Shor et J. Preskill, Simple Proof of Security of the BB84 Quantum Key Distribution Protocol, Physical Review Letters 85, 2000.

[Wie 83] S. Wiesner, Conjugate coding, Sigact News 15, 1983.

[Cha 03] H.Chaouche, D.Kadouche, sécurité réseaux, mémoire de fin d'études Ingénieur, Dpt Electronique, UMMTO, 2003.

[Har 05] S.Harchaoui, O.Beladjal, sécurités dans les réseaux sans fils, mémoire de fin d'études Ingénieur, Dpt Electronique, UMMTO ,2005.

[Bel 06] S.Belal, T.Toubal, sécurité dans les réseaux mobiles de 3^{éme} génération algorithme Kasumi, mémoire de fin d'études Ingénieur, Dpt Electronique, UMMTO, 2006.

Sites internet:

http://www.cki.au.dk/experiment/qryp to/doc/QuCrypt/bb84prot.html

http://www.tqc.iu.edu/News/Review_of_quantum_crypto.htm

http:// www.setit.rnu.tn/last-edition/setit 2007/R/185.pdf

http://repo.zenk-security.com/cryptographie %20. %20 Algorithmes%20%

http://csrc.nist.gov/encryption/tkencryption.html