Université Abou Bekr Belkaid
Tlemcen Algérie
جامعة أبي بكر بلقايد
تلمسان الجزائر

République Algérienne Démocratique et Populaire
Université Abou Bakr Belkaid– Tlemcen
Faculté des Sciences
Département d'Informatique

Mémoire de fin d'études

pour l'obtention du diplôme de Master en Informatique

Option: *Réseaux et système distribué (RSD)*

Thème

Transfert sécurisé des données visuelles (images) dans un réseau intranet selon l'architecture client/serveur

Réalisé par :

- **KARAM FOUAD**
- **IMOULOUDENE SALAH EDDINE**

Présenté le 24 Juin 2015 devant le jury composé de MM.

- **LABRAOUI NABILA** *(Président)*
- **BENAISSA Mohamed** *(Encadreur)*
- **BENMOUNA YOUCEF** *(Examinateur)*
- **BELHOUCINE AMINE** *(Examinateur)*

Année universitaire : 2014-2015

Remerciements

En préambule à ce mémoire, nous souhaitent adressons nos remerciements les plus sincères aux personnes qui nous ont apporté leur aide et qui ont contribué à l'élaboration de ce mémoire ainsi qu'à la réussite de cette formidable année universitaire. Nous tenons à remercier sincèrement Mr BENAISSA Mohamed, qui, en tant que encadreur de mémoire, s'est toujours montré à l'écoute et très disponible tout au long de la réalisation de ce mémoire, ainsi pour l'inspiration, l'aide et le temps qu'elle a bien voulu nous consacrer et sans qui ce mémoire n'aurait jamais vu le jour.

Nous exprimons également nos gratitudes aux membres du jury Mme LABRAOUI NABILA, Mr BENMOUNA YOUCEF et BELHOUCINE AMINE qui nous ont honorés en acceptant de juger ce modeste travail.

Nous tenons à la fin de ce travail à remercier ALLAH le tout puissant de nous avoir donné la foi et de nous avoir permis d'en arriver là.

Dédicace

Toutes les lettres ne sauraient trouver les mots qu'il faut. Tous les mots ne sauraient exprimer la gratitude, le respect, la reconnaissance…

Nous dédions ce mémoire à nous chers parents qui n'ont épargné aucun effort pour nous soutenir depuis le début de nos études jusqu'à leur fin.

Nous tenons par le présent travail à témoigner notre reconnaissance envers tous nos profs et notre encadreur en particulier, pour toute leur aide et leur disponibilité ainsi que pour leurs conseils.

Table des matières

Liste des Figures

Liste des Tableaux

Introduction générale

De nos jours, de plus en plus des images numériques sont transférées sur les réseaux informatiques. Les travaux présentés dans ce projet montrent comment des algorithmes de cryptage permettent la sécurisation des images. Pour ce faire, les images peuvent être cryptées au niveau des codages source afin de faire remonter cette fonctionnalité au niveau des couches hautes (applications). De cette manière, les fonctionnalités cryptage d'images sont insérées au niveau d'un logiciel. La protection est alors assurée pendant la transmission des images mais aussi pour l'archivage de ces données numériques. L'enjeu consiste ensuite à faire résister ces chiffrements à des traitements avals comme la compression. En effet, la quantité d'information (entropie) à transmettre augmente fortement entre l'image originale et l'image cryptée. Dans le cas particulier de certains types des images, de grandes zones homogènes apparaissent. Ces zones perturbent l'efficacité des algorithmes de chiffrement.

Les réseaux informatiques sont complexes et les écoutes illégales possibles. Il se pose donc un réel problème quant à la sécurité lors de la transmission de données. Pour des raisons confidentielles, le transfert des images ne peut se faire avec un tel risque et doit donc se protéger. La protection la plus adaptée pour ce type de communication réside dans la cryptographie.

Beaucoup de techniques de cryptage de texte ont été développées. Depuis l'antiquité, les hommes ont toujours essayé de coder des messages secrets pour se prévenir des oreilles malveillantes. Dans les premières esquisses de cette science du secret, la sécurité résidait dans la confidentialité de l'algorithme qui permettait le chiffrement et le déchiffrement. C'est au fil du temps qu'est apparue progressivement la notion de clef. Aujourd'hui, les systèmes de cryptage reposent sur des algorithmes mis à disposition de tous et c'est la clef, code secret particulier, qui est confidentielle et qui permet de crypter ou de décrypter le message.

Notre mémoire est structurée comme suite :

Le premier chapitre comporte une Etude de l'architecture client/serveur, le deuxième chapitre comporte une Introduction générale sur des algorithmes de chiffrement a clé secret et a clé publique, le troisième chapitre contient une Introduction générale sur les images et le quatrième chapitre, comporte une Etude comparative entre les algorithmes de cryptographie selon la vitesse et la visualité de l'opération du chiffrement des images.

Chapitre 1

Etude de l’architecture client/serveur

Chapitre 1 : Etude de l'architecture client/serveur

1. Introduction

Ces vingt dernière années ont vues une évolution majeurs des systèmes d'information, à savoir le passage d'une architecture centralisé à travers de grosses machines (des mainframes) vers une architecture distribuée basée sur l'utilisation de serveurs et de postes clients grâce à l'utilisation de PC et des réseaux.

Cette évolution a été possible essentiellement grâce à deux facteurs qui sont :

- la baisse de prix d'informatique personnelle.
- le développement des réseaux.

Bien que la formulation du modèle client-serveur dans les années 1960 et 1970, les informaticiens de Xerox PARC utilisés les termes serveur-hôte (server-host) et de l'utilisateur-hôte (user-host).
Un contexte dans le quelles chercheurs ont utilisé ces termes était dans la conception d'un langage de programmation de réseau informatique appelé Décode-Encode Langue (DEL).
Le modèle client-serveur de l'informatique est une structure d'applications distribuées qui partitionne les tâches ou des charges de travail entre les fournisseurs d'une ressource ou un service, appelés serveurs, et les demandeurs de services, appelés clients. Souvent, les clients et les serveurs communiquent via un réseau informatique sur matériel séparée, mais le client et le serveur peuvent résider dans le même système. Un hôte de serveur exécute un ou plusieurs programmes de serveurs qui partagent leurs ressources avec les clients. Un client ne partage aucune de ses ressources, mais demande le contenu ou le service d'un serveur. Les clients lancent donc des sessions de communication avec les serveurs qui attendent les demandes entrantes.
Exemples d'applications informatiques qui utilisent le modèle client-serveur sont Email, l'impression en réseau, et le World Wide Web.

2. Rôles

La caractéristique client-serveur décrit la relation de programmes associés à une application. Le composant serveur dispose d'une fonction ou d'un service à un ou plusieurs clients, qui initient des demandes pour de tels services.

Les serveurs sont classés par les services qu'ils fournissent. Par exemple, un serveur Web fournit des pages Web. Une ressource partagée peut être tout de logiciels et de composants électroniques de l'ordinateur serveur, des programmes et des données à des processeurs et des périphériques de stockage. Le partage des ressources d'un serveur constitue un service.

Si un ordinateur est un client, un serveur, ou les deux, est déterminée par la nature de l'application qui nécessite des fonctions de service. Par exemple, un seul ordinateur peut exécuter le serveur web et le logiciel de serveur de fichiers en même temps de servir des données différentes à des clients faisant différents types de demandes. Le logiciel client peut également communiquer avec le logiciel serveur dans le même ordinateur. La communication entre les serveurs, comme pour synchroniser les données, est parfois appelée inter-serveur ou de serveur à serveur de communication.

3. La répartition des tâches

Dans l'architecture client-serveur, une application est constituée de trois parties :

- L'interface utilisateur.
- Le logiciel des traitements.
- La gestion des données.

Le client n'exécute que l'interface utilisateur (souvent une interface graphique) ainsi, que la logique des traitements (formuler la requête), laissant au serveur de bases de données la gestion complète des manipulations de données.

La liaison entre le client et le serveur correspond à tout un ensemble complexe des logiciels appelé middleware qui se charge de toutes les communications entre les processus. [s1]

4. Les différents modèles de client/serveur

En fait, les différences sont essentiellement liées aux services qui sont assurés par le serveur.[s2] On distingue couramment :

4.1. Le client-serveur de donnée

Dans ce cas, le serveur assure des taches de gestion, stockage et de traitement de donnée .c'est le cas le plus connu de client- serveur est utilisé par tous les grands SGBD :

La base de données avec tous ses outils (maintenance, sauvegarde….) est installée sur un poste serveur.

Sur les clients, un logiciel d'accès est installé permettant d'accéder à la base de données du serveur tous les traitements sur les données sont effectués sur le serveur qui renvoie les informations demandées par le client.

4.2.Le client-serveur de présentation

Dans ce cas la présentation des pages affichées par le client est intégralement prise en charge par le serveur. Cette organisation présente l'inconvénient de générer un fort trafic réseaux.

4.3.Le client-serveur de traitement

Dans ce cas, le serveur effectue des traitements à la demande du client .Il peut S'agir de traitement particulier sur des données, de vérification de formulaire de saisie, de traitements d'alarmes Ces traitements peuvent être réalisés par des programmes installés sur des serveurs mais également intégrés dans des bases de données, dans ce cas, la partie donnée et traitement sont intégrés.

5. Notion de port

Lors d'une communication en réseau, les différents ordinateurs s'échangent des informations qui sont généralement destinées à plusieurs applications (le client mail et le navigateur internet par exemple).

Seulement ces informations transitent par la même passerelle. Il faut donc savoir pour quelle application telle information est destinée. On attribue donc des ports pour chaque application. Les informations sont multiplexées et passent par la passerelle. A leur arrivée (vers le serveur) ou à leur réception (vers votre machine) elles sont démultiplexées et chaque information distincte passe par le port qui lui est associé. Les informations sont ensuite traitées par l'application correspondante. [s3]

Un port est codé sur 16 bits, il y a donc 65536 ports.

L'adresse IP plus le port (exemple : **127.0.0.1:80**) est appelée socket.

Les ports ce sont vus attribuer une assignation par défaut pour aider à la configuration des réseaux.

Voici les principaux ports et le protocole les utilisant :

Port Service ou Application.

21	FTP
23	Telnet
25	MTP
53	DNS
80	HTTP
110	POP3
119	NNTP

Tableau 1 : ports et protocoles

Les ports 0 à 1023 sont les ports reconnus ou réservés et sont assignés par l'IANA (Internet Assigned Numbers Authority).
Les ports 1024 à 49151 sont appelés ports enregistrés et les ports 49152 à 65535 sont les ports dynamiques (ou privés).

6. Notion de protocoles

Un protocole est une série d'étapes à suivre pour permettre une communication harmonieuse Entre plusieurs ordinateurs.
Internet est un ensemble de protocoles regroupés sous le terme "TCP-IP" (**T**ransmission **C**ontrol **P**rotocol/**I**nternet **P**rotocol). Voici une liste non exhaustive des différents protocoles qui peuvent être utilisés :
HTTP : (Hyper Texte Transfert Protocol) : c'est celui que l'on utilise pour
Consulter les pages web.

- FTP : (File Transfert Protocol) : C'est un protocole utilisé pour transférer des fichiers.
- SMTP : (Simple Mail Transfert Protocol) : c'est le protocole utilisé pour envoyer des mails.
- POP : C'est le protocole utilisé pour recevoir des mails.
- Telnet : utilisé surtout pour commander des applications côté serveur en lignes
- IP (internet Protocol) : L'adresse IP vous attribue une adresse lors de votre connexion à un serveur.

  Les protocoles sont classés en deux catégories

  - Les protocoles où les machines s'envoient des accusés de réception (pour permettre une gestion des erreurs). Ce sont les protocoles "orientés connexion"

- Les autres protocoles qui n'avertissent pas la machine qui va recevoir les données sont les protocoles "non orientés connexion"

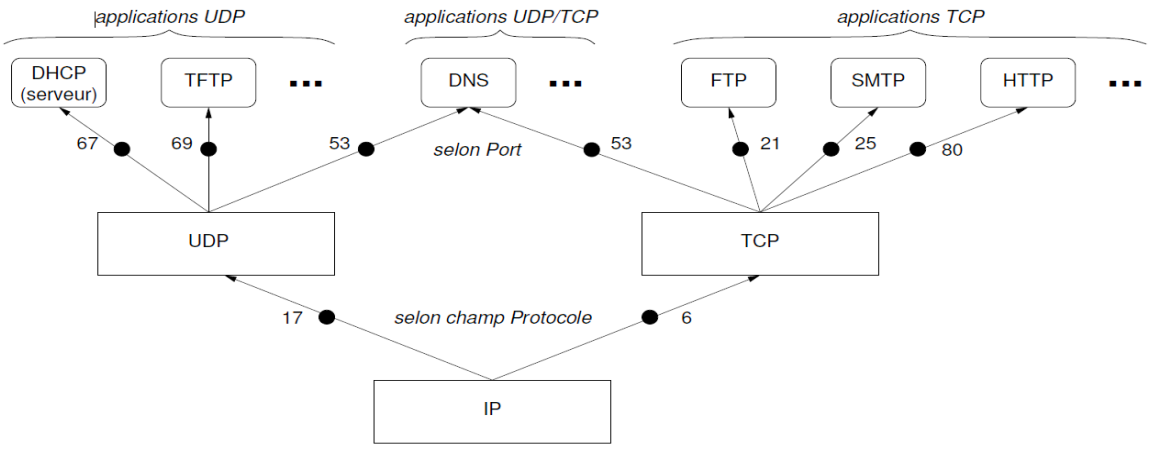


Figure 1 : protocoles et ports

7. Socket

Un socket est un port de communication ouvert au niveau des couches réseaux et permettant de faire passer des flux. La communication est en point à point en mode client-serveur. La communication est bi-directionnelle. La communication se fait en mode connecté (TCP) ou non connecté (UDP).

Exemple de Communication point à point bidirectionnelle :

- Les Sockets offrent une interface vers les protocoles de transport (TCP et UDP)
- Un Socket désigne l'extrémité d'une connexion
- Un Socket est associé à un port
- Une fois la connexion établie, la communication est similaire à une écriture sur un fichier. [1]

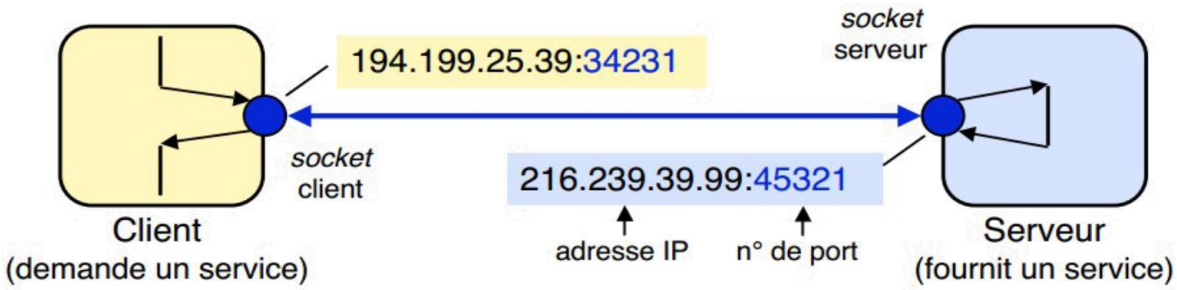


Figure 2 : socket

8. Les middlewares

On appelle middleware (ou logiciel médiateur en français), littéralement " élément du milieu", l'ensemble des couches réseau et services logiciel qui permettent le dialogue entre les différent composantes d'une application répartie. Ce dialogue se base sur un protocole applicatif commun, défini par l'API du middleware.

Middleware c'est une interface de communication universelle entre processus. Il représente véritablement la clef de voûte de toute application client/serveur.

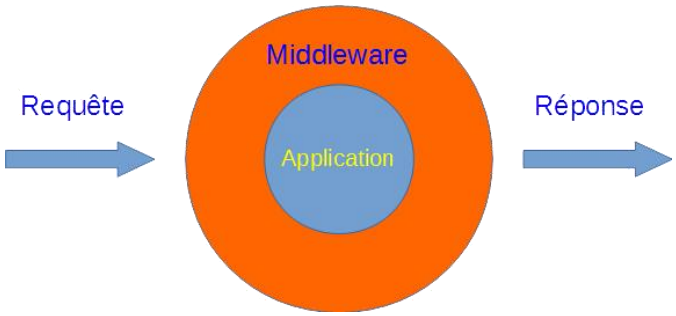


Figure 3 : middlewares

9. Les services des middlewares

Un middleware susceptible de rendre les services suivants :

- **Conversion** : Services utilisé pour la communication entre machine mettant en œuvre des formats de données différentes
- **Adressage** : Permet d'identifier la machine serveur sur laquelle est localisé le service demandé afin d'en déduire le chemin d'accès. Dans la mesure du possible.
- **Sécurité** : Permet de garantir la confidentialité et la sécurité des données à l'aide de mécanismes d'authentification et de cryptage des informations.

- **Communication** : Permet la transmission des messages entre les deux systèmes sans altération. Ce service doit gérer la connexion au serveur, la préparation de l'exécution des requêtes, la récupération des résultats et la déconnexion de l'utilisation. Le middleware masque la complexité des échanges inter-applications et permet ainsi d'élever le niveau des API utilisées par les programmes. Sans ce mécanisme, la programmation d'une application client/serveur serait complexe et difficilement évolutive.

10. RPC - Remote Procedure Call

Un RPC est un Protocole permettant de lancer une procédure d'un programme sur un ordinateur distant. Cette méthode permet au programmeur de réaliser assez simplement des programmes Client-Serveur. [s4]

11. Conclusion

Dans ce premier chapitre, nous avons présenté une étude sur l'architecture client-serveur, ainsi nous avons indiqués l'importance de la notion de protocoles, ports, sockets et middleware pour simplifier l'implémentation des applications client/serveur dans l'Internet. Dans le chapitre suivant nous présentons une introduction générale sur des algorithmes de chiffrements à clé secrète et à clé publique.

Chapitre 2

Cryptographie à clé secrète et à clé publique

Chapitre 2 : Cryptographie à clé secrète et à clé publique

1. Introduction

La cryptographie, ou art de chiffrer, coder les messages, est devenue aujourd'hui une science à part entière. Au croisement des mathématiques, de l'informatique, et parfois même de la physique, elle permet ce dont les civilisations ont besoin depuis qu'elles existent : le maintien du secret. Pour éviter une guerre, protéger un peuple, il est parfois nécessaire de cacher des choses.

La cryptographie étant un sujet très vaste, ce chapitre se focalisera essentiellement sur les méthodes de chiffrement dites modernes, c'est-à-dire celles étant apparues et utilisées après la Seconde Guerre mondiale. On passera en revue la saga du DES et de l'AES, en passant par le fameux RSA, le protocole le plus utilisé de nos jours. Ayant longtemps été l'apanage des militaires et des sociétés possédant de gros moyens financiers, la cryptographie s'est au fil du temps ouverte au grand public, et est donc un sujet digne d'intérêt. Toutes les méthodes de cryptographie seront présentées dans leur ordre chronologique d'apparition.

La cryptologie est la "science du secret", et regroupe deux branches : d'une part, la cryptographie, qui permet de coder les messages, et d'autre part, la cryptanalyse, qui permet de les décoder.

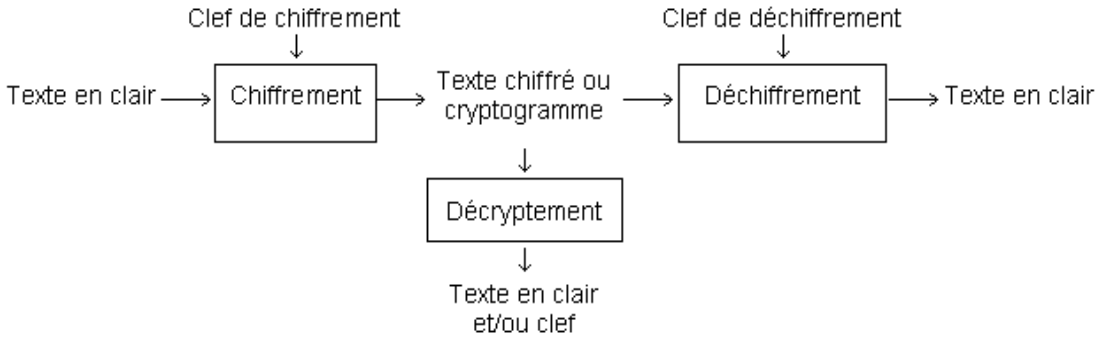


Figure 4 : chiffrement/déchiffrement

Terminologie

- **Texte en clair :** c'est le message à protéger.
- **Texte chiffré :** c'est le résultat du chiffrement du texte en clair.
- **Chiffrement :** c'est la méthode ou l'algorithme utilisé pour transformer un texte en clair en texte chiffré.
- **Déchiffrement :** c'est la méthode ou l'algorithme utilisé pour transformer un texte chiffré en texte en clair.
- **Clé :** c'est le secret partagé utilisé pour chiffrer le texte en clair en texte chiffré et pour déchiffrer le texte chiffré en texte en clair. On peut parfaitement concevoir un algorithme qui n'utilise pas de clé, dans ce cas c'est l'algorithme lui-même qui constitue la clé, et son principe ne doit donc en aucun cas être dévoilé.
- **Cryptographie :** cette branche regroupe l'ensemble des méthodes qui permettent de chiffrer et de déchiffrer un texte en clair afin de le rendre incompréhensible pour qui conque n'est pas en possession de la clé à utiliser pour le déchiffrer.
- **Cryptanalyse :** c'est l'art de révéler les textes en clair qui ont fait l'objet d'un chiffrement sans connaître la clé utilisée pour chiffrer le texte en clair. [2]
- **Cryptologie :** il s'agit de la science qui étudie les communications secrètes. Elle est composée de deux domaines d'étude complémentaires : la cryptographie et la cryptanalyse.
- **Décrypter :** c'est l'action de retrouver le texte en clair correspondant à un texte chiffré sans posséder la clé qui a servi au chiffrement. Ce mot ne devrait donc être employé que dans le contexte de la cryptanalyse.
- **Crypter :** en relisant la définition du mot décrypter, on peut se rendre compte que le mot crypter n'a pas de sens et que son usage devrait être oublié. Le mot cryptage n'a pas plus de sens non plus.
- **Coder, décoder :** c'est une méthode ou un algorithme permettant de modifier la mise en forme d'un message sans introduire d'élément secret. Le Morse est donc un code puisqu'il transforme des lettres en trait et points sans notion de secret. L'ASCII est lui aussi un code puisqu'il permet de transformer une lettre en valeur binaire.

2. Historique

2.1. Le Chiffre de César

Jules César était un général, homme politique et écrivain romain, né à Rome le 12 juillet ou le 13 juillet 100 av. J.-C. et mort le 15 mars 44 av. J.-C. Il aurait été assassiné par une conspiration, son propre fils Brutus lui portant le coup de grâce. César s'est illustré lors de la guerre des Gaules, ce qui a donné des siècles plus tard son personnage dans la bande dessinée Astérix le Gaulois. Il utilisait une méthode de chiffrement qui porte aujourd'hui son nom. [s5]

2.1.1. Principe

Le chiffre de César est la méthode de cryptographie la plus ancienne communément admise par l'histoire. Il consiste en une substitution mono-alphabétique : chaque lettre est remplacée ("substitution") par une seule autre ("mono-alphabétique"), selon un certain décalage dans l'alphabet ou de façon arbitraire. D'après Suétone, César avait coutume d'utiliser un décalage de 3 lettres : A devient D, B devient E, C devient F, etc. Il écrivait donc son message normalement, puis remplaçait chaque lettre par celle qui lui correspondait :

CLAIR	**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**
-> décalage = 3																										
CODE	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

Tableau 2 : Table de Sésar

Exemple : d'après cette méthode, « KARAM » devient donc « NDUDP ».

2.1.2. Sécurité

Niveau sécurité, le chiffre de César n'est pas fiable du tout, et ce pour deux raisons :

- Il n'existe que 26 façons différentes de crypter un message : puisqu'on ne dispose que de 26 lettres, il n'y a que 26 décalages possibles. Dès lors, des **attaques exhaustives** (tester toutes les décalages un à un) ne demanderaient que très peu de temps.
- Le chiffre de César est très vulnérable à l'analyse des fréquences.

3. Chiffrement à clé secret

3.1. Chiffrement et système cryptographique

Une fonction cryptographique, ou de chiffrement, est la donnée d'une transformation, en général bijective :

f : M --> C où M est l'ensemble des messages en clair et C est l'ensemble des messages chiffrés
Comme il s'agit d'une fonction bijective, l'opération permettant de passer de C à M est f^{-1}, c'est le déchiffrement. Ce principe de chiffrement/déchiffrement est dit "symétrique" ; car l'utilisation d'une bijection permet l'aller-retour très facilement.

L'emploi d'une bijection se justifie également parce qu'elle permet une correspondance univoque entre les éléments de l'ensemble. Autrement dit, à partir d'un message crypté, il n'est pas possible de tomber sur plusieurs possibilités de messages en clair.
Il est donc primordial, pour lire/écrire des messages cryptés :

- D'utiliser une fonction facilement inversible, en particulier pour un usage privé (il faut être capable de décrypter les messages sans avoir recours à des moyens techniques colossaux).
- De préserver cette fonction secrète, car si un "ennemi" se la procure, il lui suffira de l'inverser pour déchiffrer le message !

Ce dernier point implique un autre problème : l'utilisation d'une même fonction f pour crypter un message risque, à la longue, d'en dévoiler le secret. Ainsi, l'idéal serait de changer régulièrement de fonction de cryptage.
On définit donc un système cryptographique, ou de chiffrement, ou encore un chiffre, comme étant une famille finie 'F' de fonctions 'f' cryptographiques, chacune étant déterminée par un paramètre 'K', appelé clé secrète : $F = (f_K)$
On utilise ainsi la même "structure" de fonction de cryptage, mais avec chaque fois un paramètre K différent. [3] [4]

3.2. Le principe de Kerckhoffs

Plaçons-nous du point de vue du cryptanalyste, le spécialiste qui veut décoder le message. Idéalement, celui-ci doit trouver la clé 'K' et la transformation associée f_K. Cependant, la réalité est un peu différente. En vérité, si la fonction 'f' de cryptage est utilisée à grande échelle, il est illusoire de le considérer comme secret. On suppose donc que le cryptanalyste connaît

entièrement le système, mais qu'il ne sait pas quelle transformation f_K, c'est-à-dire qu'il ne connaît pas la clé secrète K !

C'est là le principe énoncé par Auguste Kerckhoffs à la fin du XIXe siècle, dans un article sur la cryptographie militaire :

La sécurité d'un crypto système ne doit reposer que sur le secret de la clé. Tous les autres paramètres sont supposés publiquement connus.

Ce principe est le fondement de la cryptographie à clé secrète. [s6]

3.3.Schémas de Feistel, ou chiffrement par blocs

Plutôt que d'utiliser des clés immenses telles que pour un chiffre à usage unique, on utilise le plus souvent des algorithmes qui ont une clé secrète relativement petite (de 80 à 128 bits, mais qui utilisent cette clé d'une façon apparemment si complexe qu'il est impossible à un ennemi d'en trouver la valeur.

Dès lors, puisque la clé est si petite, n'est-il pas aisé d'essayer toutes les possibilités jusqu'à parvenir au décryptement ? En fait, aucun ordinateur ne pourrait y arriver en un temps réaliste. En effet, si la clé était de 128 bits, c'est-à-dire une succession de 128 "0" et "1", il y aurait 2^{128} Clés possible.

L'objectif est donc le suivant : élaborer à partir du message M une suite aléatoire de chiffres, ou du moins qui paraisse aléatoire, que seule la détention de la clé K permet de déchiffrer. Concrètement, il s'agit de construire une fonction bijective "pseudo-aléatoire" :

- Elle doit être une bijection, car elle doit faire correspondre à chaque chiffre du message en clair un chiffre du message codé ; et car elle doit permettre de pouvoir, à partir d'un chiffre C, de remonter de façon univoque au chiffre correspond du message en clair.
- Elle doit être ou paraître aléatoire : en cryptographie, la perfection même est l'aléatoire, le message codé doit avoir l'air de découler directement du hasard, pour limiter les risques d'une attaque par analyse du texte chiffré, de ses redondances, etc.

La mise au point d'une fonction réunissant ces deux conditions posa problème aux cryptographes jusque dans les années 1950, lorsque Feistel montra qu'une fonction pseudo-aléatoire se transformait, par une méthode simple, en bijection. Actuellement, c'est la méthode de chiffrement à clé secrète la plus utilisée.

3.3.1. L'algorithme

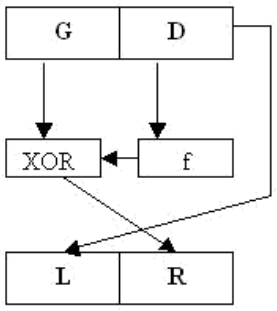


Figure 5 : Algorithme de Feistel

3.3.2. Explication

Soit une fonction f qui prend comme argument un mot de n bits.
L'algorithme de chiffrement va procéder en chiffrant des blocs de 2n bits, qu'on partage en 2 parties de n bits chacune : les parties gauche (G) et droite (D).
L'image du bloc (G,D) est le bloc (L,R), avec L=D et R = G XOR f(D).
Cette transformation est bijective, car si on a un couple (L,R), on retrouve bien (G,D) par D=L et G=R XOR f(L).
La partie droite n'a pas été transformée (juste envoyée à gauche). Il faut donc répéter le schéma de Feistel un certain nombre de fois (on parle de **tours**).

3.4. Le DES

3.4.1. Introduction

Le Data Encryption Standard (standard de chiffrement de données) a été publié en 1977, et fut ainsi le premier algorithme cryptographique à petite clé secrète (56 bits) à avoir été rendu public. Le DES consiste en un réseau de Feistel de 16 tours : le message à chiffrer est découpé en blocs de 64 bits, chacun d'eux étant séparé en deux sous-blocs de 32 bits.
L'algorithme du DES sera le plus utilisé dans le monde jusqu'en 1998. A cette époque, une association de particuliers fit construire, pour moins de 250 000 $ (somme dérisoire pour un Etat ou une organisation mafieuse), un processeur capable de casser le DES. A l'heure actuelle, trois jours suffisent aux ordinateurs pour le percer, et ce grâce à des attaques exhaustives !
On aura bien tenté d'améliorer le DES, en doublant la taille de sa clé (on parle alors de TDES), mais cette version n'était pas assez rapide. Le NIST (National Institute of Standards and

Technologies) lance donc un concours pour créer un successeur au DES, et ce sont les belges Joan Daemen et Vincent Rijmen qui seront retenus.

3.4.2. Description d'algorithme

Pour être chiffré, un bloc subit tout d'abord une permutation initiale, puis un algorithme complexe est appliqué en fonction de la clé (calcul médian), et enfin le bloc subit une permutation finale. Cette dernière permutation est l'inverse de la permutation initiale. De cette façon, l'algorithme de chiffrement et de déchiffrement est le même.
Le calcul médian dépendant de la clé peut être défini comme étant deux fonctions : une première appelée la fonction de chiffrement et une fonction de programmation de la clé. [s7]

3.4.3. Sécurité du DES

Depuis son invention, la sécurité du DES a fait l'objet d'études attentives. Des techniques spéciales telles que la cryptanalyse différentielles , ou linéaire , ont été inventées pour attaquer le DES , mais les attaques les plus efficaces proviennent d'une exploration exhaustive de l'espace des clés .Avec des matériels spécifique ou des grands réseaux de stations de travail , il est maintenant possible de déchiffrer les cryptogrammes venant du DES en quelques jours , voire même quelque heures .Au train ou la puissance des ordinateurs augmente , on s'attend à ce que le DES puisse bientôt être cassé par un simple PC.

3.5. L'AES

L'Advanced Encryption Standard [7], ou Rijndael, est donc officialisé le 2 octobre 2000. Joan Daemen et Vincent Rijmen devaient respecter les conditions suivantes :

- Une large portabilité
- Un chiffrement rapide
- Un algorithme simple à comprendre et surtout, libre de droits

L'AES fait subir au message quatre transformations successives, agissant directement sur les octets :

- Une transformation non linéaire d'un octet
- Un décalage de lignes
- Un brouillage de colonnes
- Une addition de clés

3.6. Le OU exclusif

Le OU exclusif ou proprement dit le XOR est un algorithme de cryptographie le plus simple a implémenter sur ordinateur, c'est un algorithme symétrique puisque l'algorithme de cryptage

est similaire à celui de décryptage, il fonctionne par une clé, qui doit être choisi selon le message à crypter, c'est l'étape la plus importante.

Il est relativement facile de crypter une image avec des techniques informatiques simples l'idée est d'additionner deux images, l'originale et une image clef.

Une image digitalisée peut être décrite comme une suite de pixels, chaque pixel ayant une couleur. Supposons pour simplifier que l'image est en niveau de gris. Le niveau de gris sera représenté par exemple par un nombre entier entre 0 est 255 (0 : noir ,255 : blanc).

Ces nombres sont codés dans l'ordinateur en binaire sur 8bits .on peut additionner deux bits à l'aide de la fonction XOR donnée par le tableau suivant :

XOR	0	1
0	0	1
1	1	0

Tableau 3 : Tableau de XOR

Ainsi, si le premier pixel de l'image originale est 01110011 et le premier pixel de l'image clef 10100101, l'addition des deux sera :

1er pixel de l'image original	01110011
1er pixel de l'image clef	10100101
1er pixel de l'image brouillée	11010110

Le grand intérêt de cette méthode réside dans le fait que si l'on additionne l'image brouillée et l'image clef, on retrouve l'image originale. Si l'on additionne le premier pixel de l'image brouillée au premier pixel de l'image clef on aura :

1er pixel de l'image	11010110
1er pixel de l'image clef	10100101
1er pixel de l'image original	01110011

3.7. Le Blowfish

3.7.1. Introduction

Blowfish a été conçu par Bruce Schneier en 1993 comme étant une alternative aux algorithmes existants, en étant rapide et gratuit.

3.7.2. Caractéristiques de Blowfish

- plus rapide que le DES.
- Il est un chiffrement Feistel, utilisant itérativement une fonction de chiffrement 16 fois.
- La grandeur des blocs est de 64 bits.
- Il peut prendre une longueur de clé variant entre 32 bits et 448 bits.

Depuis sa conception il a été grandement analysé et est aujourd'hui considéré comme étant un algorithme de chiffrement robuste. Il n'est pas breveté et ainsi son utilisation est libre et gratuite. Cet algorithme peut être optimisé dans les applications matérielles (puces), mais il est surtout utilisé dans des logiciels.

Il y a deux parties dans l'algorithme : une première partie qui manipule l'expansion de la clé et une deuxième partie qui manipule le chiffrement des données.

3.7.3. Principe de fonctionnement

Le principe de cet algorithme se divise en 2 grandes parties

- **Expansion de la clé**

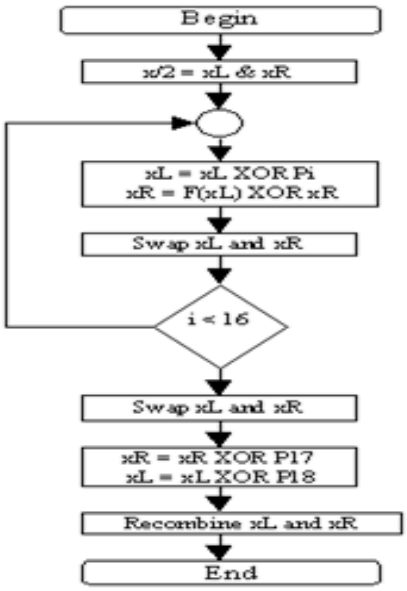
Cette partie concerne :

- la séparation de la clé originale en un ensemble de sous-clés totalisant 4168 octets.
- l'initialisation d'un tableau P et de quatre ***S-Boxes*** de 32 bits chacune.
- Le tableau P contient 18 sous-clés de 32 bits, chaque *S-Box* contient 256 entrées.

Les étapes suivantes sont utilisées pour calculer les sous-clés :

- Initialisation du tableau P et des S-Box avec une chaîne de caractères fixe (chiffres composant la constante PI).
- Utilisation de l'algorithme blowfish pour chiffrer la chaîne de caractères all-zero (chaîne de caractères fixe) en utilisant les sous-clés.
- La sortie est maintenant P[1] et P[2].
- Chiffrement des nouveaux P[1] et P[2] avec les sous-clés modifiées.
- La sortie est maintenant P[3] et P[4].
- Répéter 521 fois les deux dernières étapes afin de calculer les nouvelles sous-clés pour le tableau P et pour les quatre S-Box.

➢ Chiffrement



Début chiffrement
Divisé x en 2 : xL et xR
Pour i allant de 1 à 16 faire
xL = xL XOR P[i]
xR = F(xL) XOR xR
Permuter xL et xR
Fin Pour
Permuter xL et xR
xR = xR XOR P[17]
xL = xL XOR P[18]
x = xL + xR
Retourner x
Fin chiffrement

Figure 6 : chiffrement de blowfish

La fonction F( )

Début fonction F(Entrée : xL : 32 bits de données)

Divisé xL en 4 : a, b, c, d

Retourner ((S1,a + S2,b MOD 232) XOR S3,c) + S4,d MOD 232

Fin fonction F

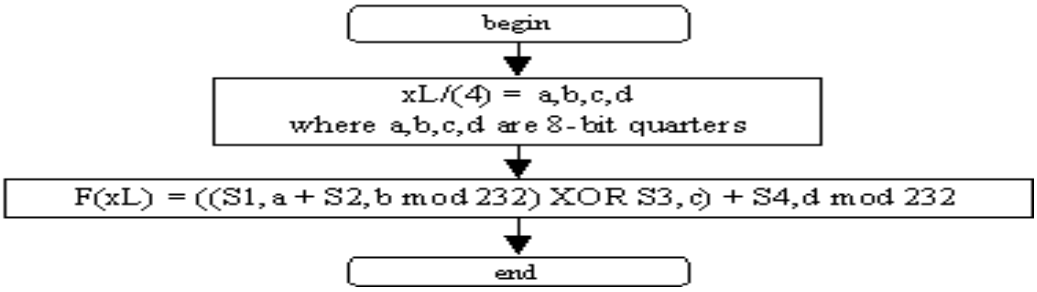


Figure 7 : fonction de chiffrement de blowfish

3.8. Conclusion sur la cryptographie à clé secrète

Malgré toutes ses évolutions et ses mises en œuvre, la cryptographie à clé secrète est toujours entravée par un défaut : la condition sine qua non de son succès est et restera le secret de sa clé (principe de Kerckhoffs). Bien qu'ayant pu au fil du temps réduire sa taille, les cryptographes ont toujours été confrontés au problème de la transmission de cette clé... Mais le progrès ne s'arrête jamais ! Si le problème est de conserver le secret de la clé, pourquoi ne pas le contourner en inventant un système qui la rend publique ?

4. Le chiffrement à clé publique

4.1. Diffie et Hellman

4.1.1. Clé "publique" et fonctions à sens unique

C'est en 1976 que Whitfield Diffie et Martin Hellman, de l'Université Stanford, proposent un principe de chiffrement entièrement nouveau : la cryptographie à clé publique, ou asymétrique. [5]

Expliquons

Alice doit recevoir un message de Bob, mais elle ne fait pas confiance au facteur qui pourrait ouvrir sa lettre. Comment peut-elle être sûre de recevoir ce message sans qu'il soit lu ?... Alice va d'abord envoyer à Bob un cadenas ouvert, dont elle seule possède la clé. Ensuite,

Bob va placer son message dans une boîte, qu'il fermera à l'aide de ce cadenas, avant de l'envoyer à Alice. Le facteur ne pourra donc pas ouvrir la boîte, puisque seule Alice possède la clé !

Ainsi, un système cryptographie à clé publique est en fait basé sur deux clés :

- Une clé publique, pouvant être distribuée librement, c'est le cadenas ouvert
- Une clé secrète, connue uniquement du receveur, c'est le cadenas fermé

C'est la raison pour laquelle on parle de chiffrement asymétrique.

En résumé, on dispose d'une fonction P sur les entiers, qui possède un inverse S. On suppose qu'on peut fabriquer un tel couple (P,S), mais que connaissant uniquement P, il est impossible (ou au moins très difficile) de retrouver S. Autrement dit, il faut déterminer mathématiquement des fonctions difficilement inversibles, ou "à sens unique".

Trouver de telles fonctions semblait ardu aux mathématiciens. En effet, comment imaginer une fonction qui soit à sens unique pour tout le monde, excepté pour son créateur qui peut l'inverser grâce à la connaissance d'une information particulière (la clé) ? Ce sont Diffie et Hellman qui ont les premiers donné une réponse à cette question.

4.1.2. Le protocole de Diffie et Hellman

Parallèlement à leur principe de cryptographie à clé publique, Diffie et Hellman ont proposé un protocole d'échanges de clés totalement sécurisé, basé sur des fonctions difficiles à inverser.

(1) Alice et Bob se mettent d'accord publiquement sur un très grand nombre premier "p" et sur un nombre "n" inférieur à "p". [6]

(2) Alice engendre une clé secrète "a" et Bob une clé secrète "b".

(3) Alice calcule l'élément public k_a et Bob l'élément public k_b :

$k_a = n^a \bmod p$

$k_b = n^b \bmod p$

(4) Alice transmet sa clé publique k_a à Bob, et Bob transmet sa clé publique k_b à Alice.

(5) Alice et Bob profitent ensuite de la commutativité de la fonction exponentielle pour établir leur secret commun :

$K_{Alice} = (k_b)^a = (n^b)^a \bmod p$

$K_{Bob} = (k_a)^b = (n^a)^b \bmod p$

$\Rightarrow K_{Alice} = K_{Bob} = n^{ab} \bmod p$

4.1.3. Sécurité du système

A priori, il n'y a pas moyen, à partir des informations transmises publiquement (p,n,n^a,n^b), de trouver n^{ab} sans caculer un logarithme modulo p, ou faire un quelconque calcul d'une complexité exagérée.

Ainsi, la sécurité du système est dite calulatoire et repose sur deux hypothèses :

- L'adversaire dispose d'une puissance de calcul limitée

- Avec cette contrainte de puissance et un temps limité, il n'est pas possible d'inverser la fonction exponentielle, ni de trouver n^{ab} à partir de p,n,n^a,n^b.

Remarque 1 : malgré tout cela, en 2001, des experts français ont réussi à inverser la fonction exponentielle modulaire pour un nombre p de 120 chiffres ! La sécurité d'un système dépend donc des progrès constants dans le domaine de la complexité algorithmique.
Remarque 2 : les fonctions à sens unique sont souvent issues de l'arithmétique modulaire, car elles se comportent de manière très irrégulière.

4.1.4. Les limites du système

Le schéma de Diffie-Hellman, bien qu'astucieux, reste un schéma de principe et souffre d'un inconvénient majeur : il n'assure pas les services de sécurité classiques que sont l'authentification mutuelle des deux intervenants, le contrôle de l'intégrité de la clé et l'anti-rejeu (vérifier qu'une information déjà transmise ne l'est pas à nouveau). L'ennemi peut donc facilement usurper l'identité d'Alice, en remplaçant son élément public par le sien.

4.2. Le RSA

4.2.1. Le principe

Le premier système à clé publique solide à avoir été inventé, et le plus utilisé actuellement, est le système RSA. Publié en 1977 par Ron Rivest, Adi Shamir et Leonard Adleman de l'Institut de technologie du Massachusetts (MIT), le RSA est fondé sur la difficulté de factoriser des grands nombres, et la fonction à sens unique utilisée est une fonction "puissance".[s8]

4.2.2. L'algorithme de chiffrement

Départ :

- Il est facile de fabriquer de grands nombres premiers p et q (+- 100 chiffres)
- Etant donné un nombre entier n = pq, il est très difficile de retrouver les facteurs p et q

(1) Création des clés

- La clé secrète : 2 grands nombres premiers p et q
- La clé publique : n = pq ; un entier e premier avec (p-1)(q-1)

(2) Chiffrement : le chiffrement d'un message M en un message codé C se fait suivant la transformation suivante :

$C = M^e \bmod n$

(3) Déchiffrement : il s'agit de calculer la fonction réciproque

$M = C^d \bmod n$

tel que $e.d = 1 \bmod [(p-1)(q-1)]$

4.2.3. La signature électronique

Après la confidentialité de la transmission d'un message subsiste un problème : son authenticité. Alice voudrait bien envoyer un message M à Bob de telle façon que celui-ci soit sûr qu'elle est réellement l'émettrice du message, et qu'un intrus ne tente pas de venir semer la confusion. [7]
Le système RSA fournit une solution à ce problème :
Rappelons les données :

- Alice seule détient la clé secrète d et diffuse la clé publique (n,e)
- Alice va se servir de la clé publique pour chiffrer le message M

(1) Alice accompagne son message chiffré de sa signature, qui correspond à : M^d

(2) Bob va donc voir si l'égalité $(M^d)^e \bmod n = M$ est vérifiée. Si c'est le cas, Alice est bien l'émettrice du message.

4.2.4. Sécurité du système : primalité, factorisation

Comme pour les protocoles fondés sur le logarithme discret, la sécurité du système RSA est calculatoire : elle dépend essentiellement de la difficulté de factoriser un entier qui est le produit de deux grands nombres premiers. Si on sait factoriser n, il est facile de trouver d. Il est à noter qu'il est conseiller d'utiliser des clés de 1024 bits (+- 309 chiffres décimaux).
Le principal objet des attaques est l'implémentation, la mise en oeuvre du RSA. C'est la manière de se servir du système qui peut poser problème, pas le système lui-même (par exemple, prendre une clé trop petite...).
Signalons enfin que le réel problème du RSA (et des autres systèmes à clé publique) n'est pas la sécurité, mais la lenteur. Tous les algorithmes à clé publique sont 100 à 1000 fois plus lents que les algorithmes à clé secrète, quelle que soit leur implémentation (logicielle ou matérielle)
Exemple : chiffrer BONJOUR

1) Alice crée ses clés :
   - La clé secrète : p = 53, q = 97 (Note : en réalité, p et q devraient comporter plus de 100 chiffres !)
   - La clé publique : e = 7 (premier avec 52*96), n = 53*97 = 5141

2) Alice diffuse sa clé publique (par exemple, dans un annuaire).
3) Bob ayant trouvé le couple (n,e), il sait qu'il doit l'utiliser pour chiffrer son message. Il va tout d'abord remplacer chaque lettre du mot BONJOUR par le nombre correspondant à sa position dans l'alphabet :

B = 2, O = 15, N = 14, J = 10, U = 21, R = 18

BONJOUR = 2 15 14 10 15 21 18

4) Ensuite, Bob découpe son message chiffré en blocs de même longueur représentant chacun un nombre plus petit que n. Cette opération est essentielle, car si on ne faisait pas des blocs assez longs (par exemple, si on laissait des blocs de 2 chiffres), on retomberait sur un simple chiffre de substitution que l'on pourrait attaquer par l'analyse des fréquences (*Voir chiffre de César*).

BONJOUR = 002 151 410 152 118

5) Bob chiffre chacun des blocs que l'on note B par la transformation $C = B^e \bmod n$ (où C est le bloc chiffré) :

$C_1 = 2^7 \bmod 5141 = 128$ $C_4 = 152^7 \bmod 5141 = 660$

$C_2 = 151^7 \bmod 5141 = 800$ $C_5 = 118^7 \bmod 5141 = 204$

$C_3 = 410^7 \bmod 5141 = 3761$

On obtient donc le message chiffré C : 128 800 3761 660 204.

5. Conclusion

Après avoir vu les avantages indéniables de la cryptographie à clé publique (transmission de la clé, sécurité, authentification), on est en raison de se demander quel pourrait être l'avenir des systèmes à clé secrète.

En effet, à l'heure actuelle, on utilise des crypto systèmes hybrides, qui couplent les avantages des deux principes, alliant la souplesse de gestion des clés de la cryptographie asymétrique et les performances (vitesse) de la cryptographie symétrique.

Il existe deux types de crypto systèmes hybrides :

- Soit, la cryptographie à clé publique sécurise le transport d'une clé symétrique
- Soit, les entités émettrice et destinatrice se mettent publiquement d'accord sur un secret commun et l'utilisent ensuite pour chiffrer les données grâce à un algorithme symétrique classique

Chapitre 3

Introduction générale sur les images

Chapitre 3 : Introduction générale sur les images

1. Introduction

Avec la parole, l'image constitue l'un des moyens les plus importants qu'utilise l'homme pour communiquer avec autrui. C'est un moyen de communication universel dont la richesse du contenu permet aux êtres humains de tout âge et de toute culture de se comprendre.

C'est aussi le moyen le plus efficace pour communiquer, chacun peut analyser l'image à sa manière, pour en dégager une impression et d'en extraire des informations précises.

De ce fait, le traitement d'images est l'ensemble des méthodes et techniques opérant sur celles-ci, dans le but de rendre cette opération possible, plus simple, plus efficace et plus agréable, d'améliorer l'aspect visuel de l'image et d'en extraire des informations jugées pertinentes.

Dans ce chapitre, nous découvrons quelques aspects théoriques liés aux images numériques.

2. Définition de l'image

L'image est une représentation d'une personne ou d'un objet par la peinture, la sculpture, le dessin, la photographie, le film, etc.

C'est aussi un ensemble structuré d'informations qui, après affichage sur l'écran, ont une signification pour l'œil humain.

Elle peut être décrite sous la forme d'une fonction I(x,y) de brillance analogique continue, définie dans un domaine borné, tel que x et y sont les coordonnées spatiales d'un point de l'image et I est une fonction d'intensité lumineuse et de couleur. Sous cet aspect, l'image est inexploitable par la machine, ce qui nécessite sa numérisation.[s9]

Figure 8 : Des images numériques

3. L'échantillonnage (le passage d'image analogique a l'image numérique)

L'échantillonnage est le procédé de discrétisation spatiale d'une image consistant à associer à chaque zone rectangulaire R(x, y) d'une image continue une unique valeur I(x, y).

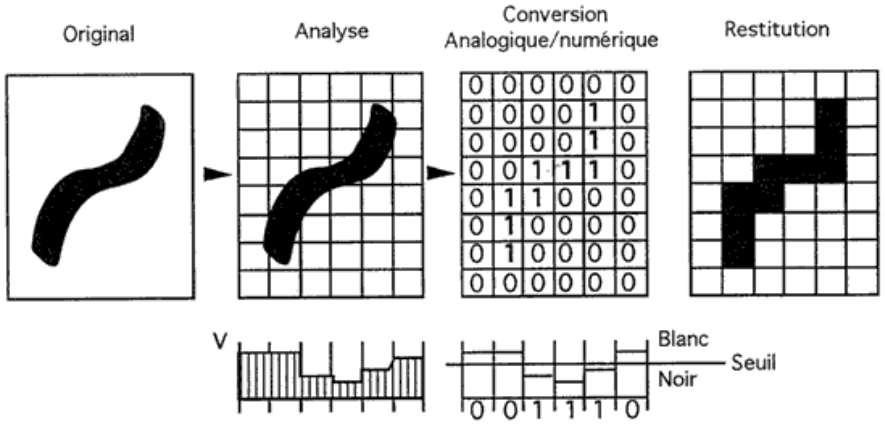


Figure 9 : Échantillonnage d'image

On parle de sous-échantillonnage lorsque l'image est déjà discrétisée et qu'on diminue le nombre d'échantillons.

4. Les images numériques

Sont des clichés électroniques d'une scène ou numérisés à partir de documents tels que photographies, manuscrits, textes imprimés, et œuvres d'art.

L'image numérique est échantillonnée et mappée comme une grille de points ou éléments d'images (picture element ou pixel).

A chaque pixel correspond une valeur tonale (noir, blanc, niveaux de gris ou couleur), exprimée en code binaire (zéros et uns). Les chiffres binaires ("bits") de chaque pixel sont stockés dans une séquence par l'ordinateur, et souvent réduits à une représentation mathématique (compressés). Les bits sont alors réinterprétés et lus par l'ordinateur afin de délivrer une version analogique en vue d'être affichée ou imprimée.

5. Types d'images

a. Les images matricielles ou « mosaïques de points » ou « images bitmap »

Figure 10 : Image matricielle (bitmap)

Ces images sont constituées d'une multitude de petits carrés colorés organisés en lignes et en colonnes pour former des ensembles généralement rectangulaires. Elles peuvent être obtenues à l'aide d'un appareil photographique numérique ou d'un scanner mais on peut aussi les créer de toutes pièces en utilisant un logiciel approprié ; c'est pourquoi nous employons dans l'en-tête de ce paragraphe le mot « image » et non le mot « photographie ».[10]

En pratique, les images matricielles sont les seules utilisables en photographie. Il est facile de comprendre que leur qualité et leur aptitude à reproduire de fins détails dépend directement du nombre de petits carrés élémentaires qui les composent. Ceux-ci, pour un usage normal, ne doivent pas être perçus individuellement.

L'agrandissement d'une image matricielle n'apporte aucune information supplémentaire. S'il est exagéré, la structure de la mosaïque apparaît et cela donne un effet désagréable ; on dit alors que l'image est « pixellisée », ce que nous justifierons par la suite.

b. Les images vectorielles

Contrairement aux images matricielles, les images vectorielles sont composée d'objets géométriques individuels (segments de droite, polygones, arcs de cercle, etc.) définis chacun par divers attributs de forme, de position, de couleur, etc. Pour les afficher à une taille donnée, il est nécessaire de les recalculer à chaque visualisation.

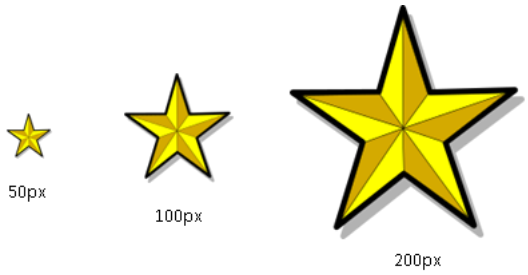


Figure 11 : Images vectorielles 1

L'intérêt de ces images est que l'on peut les agrandir autant que l'on veut sans perdre de qualité, car on évite les effets de pixellisation. Il est possible de « vectoriser » des dessins complexes, comme des tracés réalisés à main levée, des cartes géographiques, etc.

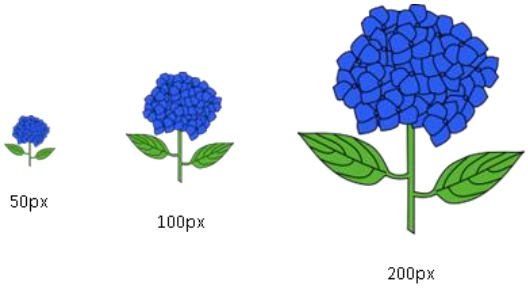


Figure 12: Images vectorielles 2

Cependant, les images vectorielles ne sont pas adaptées aux applications photographiques en raison de leur « gourmandise » en puissance de calcul et en mémoire, dès lors que le nombre des détails à reproduire est très élevé. Le logiciel libre ***Inkscape*** est un des outils qui permettent de créer de telles images.

6. Définition de Pixels

Une image est constituée d'un ensemble de points appelés **pixels** (pixel est une abréviation de ***PIC****ture* ***EL****ement*) Le pixel représente ainsi le plus petit élément constitutif d'une image numérique. L'ensemble de ces pixels est contenu dans un tableau à deux dimensions constituant l'image :

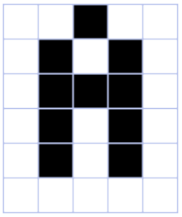


Figure 13 : Les pixels d'une image

Etant donné que l'écran effectue un balayage de gauche à droite et de haut en bas, on désigne généralement par les coordonnées [0,0] le pixel situé en haut à gauche de l'image, cela signifie que les axes de l'image sont orientés de la façon suivante :

- L'axe X est orienté de gauche à droite.
- L'axe Y est orienté de haut en bas, contrairement aux notations conventionnelles en mathématiques, où l'axe Y est orienté vers le haut.

7. La taille des images

Ce sont les mesures horizontales et verticales d'une image exprimées en pixels. Les dimensions en pixels peuvent être déterminées en multipliant la largeur et la longueur de l'image par le dpi. Un appareil photo numérique possède également des dimensions en pixels, exprimés par le nombre de pixels horizontaux et verticaux définissant sa résolution (p.ex. 2048 par 3072). Pour calculer la résolution en dpi, divisez une des dimensions en pixels du document par la dimension en pouces correspondante.

Taille de l'image = nb de pixels/longueur x nb de pixels/largeur

Par exemple un appareil photo qui produit une image de : 3008 x 2000

3008 x 2000 = 6 016 000 pixels = 6,016 millions de pixels

Il est dès lors facile de calculer l'agrandissement maximum possible avec ce fichier si on se fixe une résolution de 300dpi

3008/300 = 10,03 pouces x 2,54 = 25,48cm

2000/300 = 6,67" x 2,54 = 16,94cm

8. La Résolution

La résolution d'une image est définie par le nombre de pixels par unité de longueur.
Elle s'exprime en dpi (dots per inches) ou ppp (points par pouce). Un pouce = 2,54 centimètres.
La résolution d'une image numérique définit le degré de détail qui va être représenté sur cette image. Une image de résolution élevée compte un plus grand nombre de pixels (elle contient plus d'informations), elle est donc plus volumineuse qu'une image basse résolution de mêmes dimensions.
La question de la résolution se pose
- soit au moment de numériser une image (à quelle résolution scanner)
- soit au moment de créer un nouveau fichier image avec l'application adéquate (Photoshop par exemple)
Une autre question à se poser concernant la résolution est : que vais-je faire de cette image ?
En effet, la résolution d'une image dépend de celle du périphérique d'impression ou d'affichage (résolution de sortie).
Pour l'impression, une résolution de 300 dpi est la règle. Par contre pour les images destinées à l'écran une résolution de 72 dpi est suffisante (la résolution d'un moniteur est en général de 72 dpi).

9. Profondeur de couleur (profondeur de bit)

Est définie par le nombre de bits utilisés pour représenter chaque pixel. Plus la profondeur de bit est élevée, plus grand sera le nombre de teintes (niveaux de gris ou couleur) représenté. Les images numériques peuvent être produites en noir et blanc (deux couleurs), niveaux de gris ou couleur.Une image bitonale (deux couleurs) est représentée par des pixels de 1 bit chacun, pouvant représenter deux teintes (d'habitude le noir et le blanc), en utilisant la valeur 0 pour le noir et 1 pour le blanc.
Une image en niveaux de gris est composée de pixels possédant plusieurs bits d'informations, allant en général de 2 à 8 bits, ou davantage.
Une image couleur est typiquement représentée par une profondeur de bit variant de 8 à 24 bits ou plus. Dans une image 24 bits, les bits sont souvent divisés en 3 groupes : 8 pour le rouge, 8 pour le vert et 8 pour le bleu. Les combinaisons de ces bits servent à représenter les autres couleurs. Une image 24 bits offre 16,7 millions de valeurs de couleurs (2^{24}). De plus en plus, les scanners capturent chaque canal de couleur à 10 bits ou plus, et les réduisent à 8 bits afin de

compenser le "bruit" du scanner et présenter une image aussi proche que possible de la perception visuelle de l'être humain.

Figure 14 : Profondeur de couleur

Profondeur de couleur : De gauche à droite, image bitonale 1 bit, niveaux de gris 8 bits, couleur 24 bits.

Calculs binaires pour le nombre de teintes représentées par les profondeurs de bit courantes :

1 bit (2^1) = 2 tons

2 bits (2^2) = 4 tons

3 bits (2^3) = 8 tons

4 bits (2^4) = 16 tons

8 bits (2^8) = 256 tons

16 bits (2^{16}) = 65536 tons

24 bits (2^{24}) = 16,7 millions de tons

10. Représentation des couleurs

Nous l'avons vu une image apparait comme une matrice où chaque case contient des nombres associés à une couleur. Usuellement on distingue 3 grands types de couleurs pour une image numérique :

- Le noir et blanc ;
- Les niveaux de gris ;
- La couleur.

Ces types sont généralement à choisir lors d'une numérisation par scanner ou lors de la configuration d'un appareil photographique.

Figure 15 : Image originale

10.1. Image noir et blanc

Le noir et blanc est le plus simple. Le contenu de chaque case de la matrice est soit un 0 (noir) soit 1 (blanc). Le nombre de couleurs n'est que de 2 et le rendu de l'image le moins performant mais parfois suffisant dans le cadre par exemple de documents scripturaux.

Figure 16: Image noire et blanc

La taille du fichier obtenu, avec la même résolution (507x676) que celle de l'image initiale, est de 42,3 ko (au lieu de 980 ko !).

10.2. Niveaux de gris

Le codage dit en niveaux de gris permet d'obtenir plus de nuances que le simple noir et blanc.

Il offre des possibilités supplémentaires pour coder le niveau de l'intensité lumineuse. La couleur est codée souvent sur un octet soit 8 bits ce qui offre la possibilité d'obtenir 256 niveau de gris (0 pour le noir et 255 pour le blanc). On peut aussi le faire avec 16 niveaux de gris (4 bits).

000	008	016	024	032	040	048	056	064	072	080	088	096	104	112	120	128	136	144	152	160	168	176	184	192	200	208	216	224	232	240	248
000	008	016	024	032	040	048	056	064	072	080	088	096	104	112	120	128	136	144	152	160	168	176	184	192	200	208	216	224	232	240	248	255

Tableau 4 : 256 niveaux de gris

Figure 17 : Image 'fleur' 256 niveaux de gris

Figure 18 : Image 'fleur' 16 niveaux de gris

De plus pour l'image codée sur 256 niveaux de gris, la valeur du pixel de coordonnées (54,51) (Situé en haut à gauche « dans le ciel ») est de 120.

Plus le niveau de gris est élevé, meilleur est la distinction des détails sur l'image. L'usage de ce codage est utilisé fréquemment pour la presse écrite ou l'envoi par messagerie électronique de fichier d'image de taille réduite avec une perte de lisibilité de l'image moindre.

10.3. Image couleur

10.3.1. Principe :

La couleur d'un pixel est obtenue, comme le ferait un peintre, par le mélange de couleurs fondamentales. Il ne s'agit pas ici de décrire toutes les techniques utilisées. Nous allons décrire un des principes les plus couramment utilisé qui est celui de la synthèse additive.

10.3.2. Codage RVB

Le principe consiste à mélanger les 3 couleurs : rouge, vert et bleu (noté RVB ou RGB en anglais). A l'aide de ces 3 couleurs, on obtient toute une palette de nuances allant du noir au blanc. A chaque couleur est associé un octet (donc 256 niveaux de luminosité) de chacune des couleurs fondamentales.

Rouge	Vert	Bleu	Couleur
255	0	0	Rouge
0	255	0	Vert
0	0	255	Bleu
0	255	255	Cyan
255	0	255	Magenta
255	255	0	Jaune
0	0	0	Noir
128	128	128	Gris moyen
255	255	255	Blanc

Tableau 5 : Principe codage de la couleur

Un pixel 'couleur' est alors codé avec 3 octets et on a alors la possibilité d'obtenir 2 24 possibilités de couleurs soit de l'ordre de 16 millions de couleurs différentes.

On dit que les images obtenues sont en couleurs « vraies ». La qualité colorimétrique obtenue est celle d'une photographie argentique couleur.

Figure 19 : Valeurs RGB d'un pixel

Ainsi le pixel de coordonnées (54, 51) (en haut à gauche de l'image « dans le ciel ») a pour valeurs RVB respectivement (77, 118, 239).

Il existe d'autres formats de codage de la couleur (image à palette 256 couleurs, mode HAM…) pour lesquels on pourra se reporter utilement à leur description en ligne sur Internet.

11. Poids d'une image

Pour connaître le poids (en octets) d'une image, il est nécessaire de compter le nombre de pixels que contient l'image, cela revient à calculer le nombre de cases du tableau, soit la hauteur de celui-ci que multiplie sa largeur. Le poids de l'image est alors égal à son nombre de pixels que multiplie le poids de chacun de ces éléments.

Voici le calcul pour une image 640x480 en *True color* :

Nombre de pixels :
640x480=307200

Poids de chaque pixel
24 bits / 8 = 3 octets

Le poids de l'image est ainsi égal à :
307200x3=921600 octets
921600/1024=900 Ko

(Pour connaître la taille en Ko il suffit de diviser par 1024).

Voici quelques exemples (en considérant que l'image n'est pas compressée) :

Définition de l'image	Noir et blanc (1 bit)	256 couleurs (8 bits)	65000 couleurs (16 bits)	True color (24 bits)
320x200	7.8 Ko	62.5 Ko	125 Ko	187.5 Ko
640x480	37.5 Ko	300 Ko	600 Ko	900 Ko
800x600	58.6 Ko	468.7 Ko	937.5 Ko	1.4 Mo
1024x768	96 Ko	768 Ko	1.5 Mo	2.3 Mo

Tableau 6 : Poids des images

12.Formats d'image

Le format d'un fichier, que ce soit une image, de la musique, un tableau ou du traitement de texte, désigne la structure de ce fichier. Cette structure est l'organisation des données interne au fichier, et détermine le logiciel nécessaire à son traitement.

Les fichiers Excel se terminent par .xls, les fichiers Word par .doc, les fichiers Jpeg par. Jpg etc….

Concernant les images, il existe une grande variété de formats de fichiers, qui ont chacun des caractéristiques propres.

Consistent à la fois des bits comprenant les informations de l'image et d'en-tête concernant la lecture et l'interprétation du fichier. Les formats de fichiers varient en termes de résolution, profondeur de bit, capacités de couleurs et support pour la compression et les métadonnées.

Le plus connu est le Jpeg. Il s'est largement démocratisé car il compresse les images pour les enregistrer.

	Type (matriciel / vectoriel)	Compression des données	Nombre de couleurs supportées	Affichage progressif	Animation	Transparence
JPEG	matriciel	Oui, réglable (avec perte)	16 millions	Oui	Non	Non
JPEG2000	matriciel	Oui, avec ou sans perte	4 milliards	Oui	Oui	Oui
GIF	matriciel	Oui, Sans perte	256 maxi (palette)	Oui	Oui	Oui

PNG	matriciel	Oui, sans perte	Palettisé (256 couleurs ou moins) ou 16 millions	Oui	Non	Oui (couche Alpha)
TIFF	matriciel	Compression ou pas avec ou sans pertes	de monochrome à 16 millions	Non	Non	Oui (couche Alpha)
SVG	vectoriel	compression possible	16 millions	* ne s'applique pas *	Oui	Oui (par nature)

Tableau 7 : Tableau comparatif des Formats d'image

13.La compression

Pour limiter la taille des fichiers, il est possible de comprimer les images. Il existe deux grand types de compression d'image, ceux qui n'entraînent aucune perte de qualité, et ceux qui entraînent une perte de qualité. Dans les deux cas, plus l'image a une variété de couleur, plus la compression sera réduite. [8]

- Les compressions sans perte de qualité : il s'agit du format TIF qui utilise la compression LZW (ou Packbits). Dans ce format, tous les points de même couleur sont regroupés. Ainsi si 8 points ont une couleur identique, au lieu de stocker 8 fois la même information, ne sont stockées que l'adresse du point de départ, celle du point d'arrivée, et la couleur. Bien évidemment, plus l'image est simple, plus le taux de compression obtenu est important.
- Les compressions avec perte de qualité utilisent deux procédés :
  - la réduction du nombre de couleur : c'est le format GIF (Graphic Interchange Format) de Compuserve (256 couleurs). En fait, le format GIF utilise deux procédés conjointement, la réduction du nombre de couleurs, et le regroupement des pixels de couleur identique. Comme la réduction du nombre de couleurs uniformise certaines couleurs, le taux de compression obtenu est souvent très important.
  - Les algorithmes de compression qui regroupent des pixels qui ont des couleurs proches. Ainsi, le procédé utilisé est le même que pour TIF, mais ici, il y a acceptation d'une perte de qualité : c'est le format JPEG (Joint Photo Expert Group, extension JPG). Lors de l'enregistrement, il est possible de choisir le taux de compression, ou le niveau de perte de qualité (selon les logiciels). A 80% de perte, les différences avec l'original ne sont quasiment pas visibles lorsqu'elles sont visualisées sur un écran.

Sans compression

JPEG fin - Faible compression

JPEG moyen - compression moyenne

JPEG faible - compression très forte

Figure 20 : Exemples de compression JPEG

Exemples de compression JPEG : Les trois images ont la même taille et la même résolution, seul le degré de compression JPEG change.

14.Conclusion

Nous avons présentés dans ce chapitre, une introduction générale sur le format et le type des images utilisés dans le domaine informatique.

Dans le chapitre suivant, nous appliquons la cryptographie pour chiffrer et sécurisé les informations d'une image qui circule sur un réseau selon le modèle client-serveur.

Chapitre 4

Etude comparative entre les algorithmes de cryptographie

Chapitre 4 : étude comparative entre les algorithmes de cryptage

1. Introduction

Comme l'importance et la valeur des données échangées sur l'Internet ou d'autres types de médias est en augmentation, la recherche de la meilleure solution pour offrir la protection nécessaire contre les attaques des voleurs de données ainsi que la fourniture de ces services dans le cadre temps opportun est l'un des plus actifs sujets dans les communautés liées à la sécurité.

Ce chapitre tente de présenter une comparaison équitable entre les algorithmes les plus connus les plus utilisées dans le domaine de cryptage de données.

Ce chapitre est divisé en 2 parties :
Partie 1 : évaluation de performance des algorithmes de cryptage.
Partie 2 : évaluation de la qualité de cryptage de ces algorithmes.

2. Présentation de l'application

Afin d'étudier les algorithmes de cryptographie on a développé cette application de chiffrement reparti selon le modèle client/serveu**r** Pour sécuriser le transfert des images sur un réseau local.

Figure 21: Interface graphique de L'application

❶ Choix de type de connexion

❷ Adresse IP de serveur

❸ Le port de connexion

❹ Bouton de connexion

❺ Statuts de connexion

❻ Bouton de déconnexion

❼ Onglet des images

❽ Bouton de chargement des images

❾ Bouton d'enregistrement des images

⓲ zone d'affichage des images

❿ Bouton d'envoi des images au client

⓫ Bouton d'affichage d'histogramme

⓬ Bouton de génération et partage de clé avec le client

⓭ zone d'affichage de clé

⓮ Bouton de cryptage des images

⓯ Bouton de décryptage des images

⓰ Bar de statuts

⓱ Choix de l'algorithme de cryptage

Première partie

3. Paramètre de comparaison

Puisque notre principale préoccupation ici est la performance de ces algorithmes sous différents paramètres, la comparaison présentée prend en compte le comportement et les performances de la vitesse de chiffrement des images.

4. Les Algorithmes comparés

Cette section a l'intention de donner l'arrière-plan nécessaire de comprendre les différences principaux entre les algorithmes comparées.(*Pour plus de détail voir le chapitre 2*)

DES : (Data Encryption Standard)

Le Data Encryption Standard est un algorithme de chiffrement symétrique utilisant des clés de 56 bits. Son emploi n'est plus recommandé aujourd'hui, du fait de sa lenteur à l'exécution et de son espace de clés trop petit permettant une attaque systématique en un temps raisonnable. Quand il est encore utilisé c'est généralement en Triple DES, ce qui ne fait rien pour améliorer ses performances. DES a notamment été utilisé dans le système de mots de passe UNIX.

3DES : (Triple DES)

Le Triple DES est un algorithme de chiffrement symétrique par bloc, enchaînant 3 applications successives de l'algorithme DES sur le même bloc de données de 64 bits, avec 2 ou 3 clés DES différentes.

AES :(Advanced Encryption Standard)

Advanced Encryption Standard ou AES, aussi connu sous le nom de Rijndael, est un algorithme de chiffrement symétrique. Il remporta en octobre 2000 le concours AES, lancé en 1997 par le NIST et devint le nouveau standard de chiffrement pour les organisations du gouvernement des États-Unis. Il a été approuvé par la NSA dans sa suite B des algorithmes cryptographiques.

RC2

RC2 est un chiffrement de bloc conçu par Ronald Rivest en 1987. L'abréviation « RC » signifie Ron's Code ou Rivest Cipher. Le cryptologue est à l'origine des autres RC.

RC4

RC4 est un algorithme de chiffrement à flot conçu en 1987 par Ronald Rivest, l'un des inventeurs du RSA, pour les Laboratoires RSA. Il est supporté par différentes normes, par exemple dans SSL ou encore WEP.

Blowfish

Blowfish est un algorithme de chiffrement symétrique par blocs conçu par Bruce Schneier en 1993, Utilise une taille de bloc de 64 bits et la clé de longueur variable peut aller de 32 à 448 bits. Elle est basée sur l'idée qu'une bonne sécurité contre les attaques de cryptanalyse peut être obtenue en utilisant de très grandes clés pseudo-aléatoires.

XOR

L'opérateur XOR est extrêmement fréquent en tant que composant des chiffrements plus complexes. Par lui-même, en utilisant une clé constante répété, un chiffre XOR simple peut être trivialement cassé. Son principal avantage est qu'il est simple à mettre en œuvre, et que l'opération XOR est peu coûteux en calculs. Un simple chiffrement XOR est donc parfois utilisé pour cacher des informations dans les cas où aucune sécurité particulière n'est requise.

RSA

Le chiffrement RSA (nommé par les initiales de ses trois inventeurs) est un algorithme de cryptographie asymétrique, très utilisé dans le commerce électronique, et plus généralement pour échanger des données confidentielles sur Internet. Cet algorithme a été décrit en 1977 par Ronald Rivest, Adi Shamir et Leonard Adleman.

5. Configuration d'évaluation

Cette section décrit l'environnement d'évaluation et les composants du système utilisés.
Comme mentionné cette évaluation utilise les classes prévues dans l'environnement java pour évaluer l'exécution de tous les algorithmes.

L'implémentation de XOR utilisé ici est celui fourni par nous sous le nom XOR.java.
*Le (*Tableau 8*)* montre les paramètres des algorithmes utilisés dans cette expérience.

Algorithme	*Taille de clé (Bits)*	*Taille de block (Bits)*
DES	***64***	***64***
3DES	***192***	***64***
AES	***128***	*****
RC2	***128***	*****
RC4	***128***	*****
Blowfish	***128***	***64***
XOR	***512***	*****
RSA	***512***	***53***

Tableau 8 : Configuration des algorithmes

6. Méthodologie d'évaluation de performance

Cette section décrit les techniques de simulation et les choix faits pour évaluer les performances des algorithmes comparés.

En plus de cela, cette section sera discutée des paramètres de méthodologie liés comme : les paramètres du système, les facteurs d'évaluation, et les paramètres initiaux d'évaluation.

7. Les paramètres du système

L'évaluation est effectuée en utilisant :

Processeur	Intel(R) Pentium(R) Dual CPU E2200 @ 2.20GHz
Fabricant	Intel
Vitesse	2.2 GHz
Nombre de cœurs	2
Mémoire	2.1 GB
Système d'exploitation	Microsoft Windows 7 Ultimate Edition Service Pack 1 (build 7601), 32-bit
Service Pack	1
Taille	32 Bit
Edition	Ultimate
IDE	NetBeans IDE 8.0.2 (Build 201411181905)
JDK	1.7.0_60
JRE	Java(TM) SE Runtime Environment 1.7.0_60-b19

Tableau 9 : Les paramètres du système

8. Les facteurs d'évaluation

Puisque les caractéristiques de sécurité de chaque algorithme que leur force contre les attaques cryptographiques sont déjà connues et discutées.

Le facteur choisi ici de déterminer la performance est la vitesse de l'algorithme pour crypter/décrypter des images de différentes tailles.

9. Procédure d'évaluation

En considérant différentes tailles des images (*voir* Tableau 10) les algorithmes sont été évalués en termes de temps nécessaire pour crypter et décrypter différentes images.

Toutes les implémentations étaient exactes pour s'assurer que les résultats seront relativement justes et précise.

Résolution d'image	128 x 128	256 x 256	512 x 512	1024 x 1024
Taille d'image (KB)	48	192	768	3072

Tableau 10 : Caractéristiques des images utilisées

Le programme d'évaluation utilisée (Illustré à la Figure 22) accepte une seul entrée qui est l'Algorithme à utiliser. Et utiliser les paramètres indiqués dans le tableau (Tableau 8) en tant que paramètres par défaut.

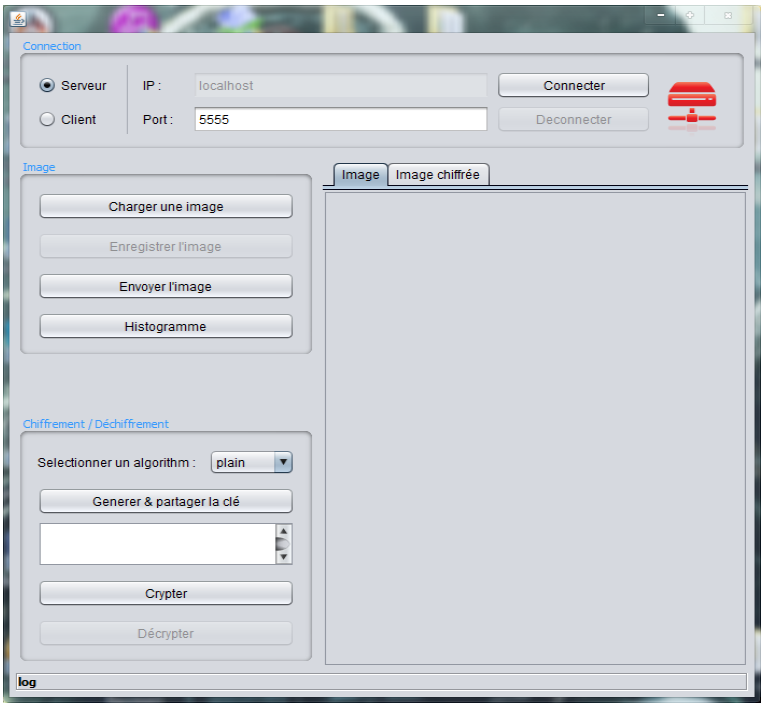


Figure 22 : L'application d'évaluation

10.Résultats d'évaluation

Cette section indique les résultats obtenus à partir de l'exécution du programme d'évaluation en utilisant différentes tailles d'image.

Les résultats montrent l'impact de la variation de la charge de données sur chaque algorithme.

Algorithmes	image 128x128	image 256x256	image 512x512	image 1024x124
DES	2.789	10.616	42.914	171.657
DESede	7.803	30.717	122.863	491.865
AES	1.291	4.614	18.745	75.461
RC2	2.475	9.341	37.63	151.491
RC4	0.692	2.322	9.101	36.815
Blowfish	1.559	5.396	21.618	86.828
XOR	0.506	2.524	11.301	47.208
RSA	98.883	389.037	1559.436	6328.048

Tableau 11 : Temps de cryptage

Algorithmes	image 128x128	image 256x256	image 512x512	image 1024x124
DES	2.783	10.715	45.819	181.455
DESede	7.829	30.912	124.327	502.555
AES	1.355	4.909	20.178	88.362
RC2	1.774	6.588	27.103	116.501
RC4	0.697	2.298	9.08	37.164
Blowfish	1.571	5.563	23.036	98.436
XOR	0.502	2.513	11.281	47.286
RSA	1078.423	4207.164	17021	68797

Tableau 12 : Temps de décryptage

algorithme	vitesse de cryptage KB/ms
DES	18.084
DESede	6.15
AES	41.612
RC2	20.553
RC4	83.686
Blowfish	35.581
XOR	67.556
RSA	0.493

Tableau 13 : vitesse de cryptage

algorithme	vitesse de décryptageKB/ms
DES	17.676
DESede	6.13
AES	38.059
RC2	28.335
RC4	84.574
Blowfish	33.338
XOR	68.079
RSA	0.054

Tableau 14 : vitesse de décryptage

Comparaison Entre RSA et les algorithmes symétriques

D'après le graphe ci-dessus, nous pouvons conclure que le temps de cryptage / décryptage est mieux en utilisant les techniques de cryptage / décryptage à clé symétrique par rapport au RSA, parce que le cryptage / décryptage à clé symétrique ne nécessite pas autant de cycles de CPU que le cryptage / décryptage à clé asymétrique, donc nous pouvons dire que c'est généralement plus rapide.

Et de l'autre côté les techniques de chiffrement asymétrique sont près de1000 fois plus lent que les techniques symétriques, car ils nécessitent plus de puissance de traitement de calcul

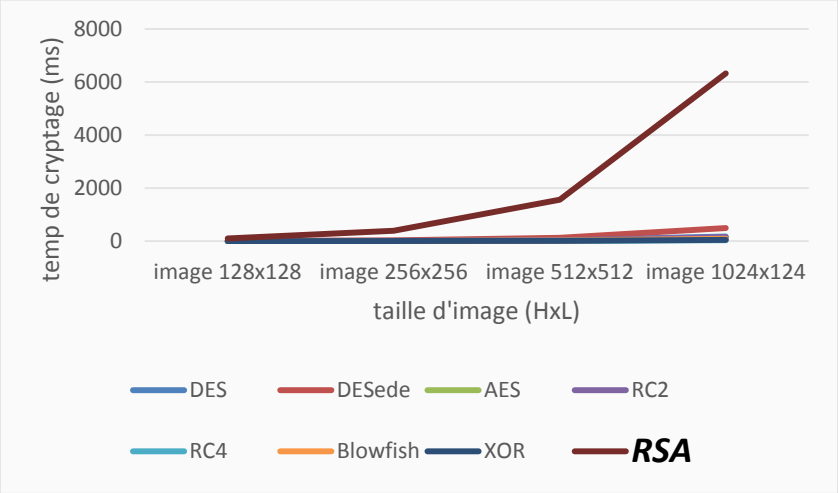


Figure 23 : comparaison de temps de cryptage entre RSA et les algo symétrique

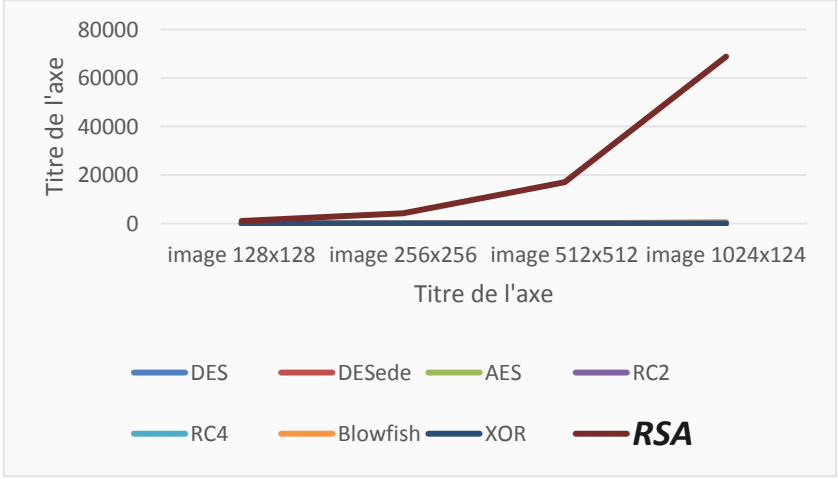


Figure 24 : comparaison de temps de décryptage entre RSA et les algo symétrique

Comparaison Entre les algorithmes symétriques

Les résultats montrent la supériorité de l'algorithme RC4 et XOR sur les autres algorithmes en termes de temps de traitement. Il montre également que RC2 consomme plus de ressources lorsque la taille de bloc de données est relativement grande. Un autre point peut être remarqué ici que 3DES exige toujours plus de temps que DES en raison de son cryptage triple caractéristique de phase. Blowfish surpassé d'autres algorithmes de chiffrement comme RC2 et DES. Enfin, il est constaté que RC2 a une faible performance et faible débit en comparaison avec les autres algorithmes qui utilise une clé de la même taille (128 bit). DES et 3DES sont connus pour avoir des trous de vers à leur mécanisme de sécurité, Blowfish et AES n'ont pas à ce jour.

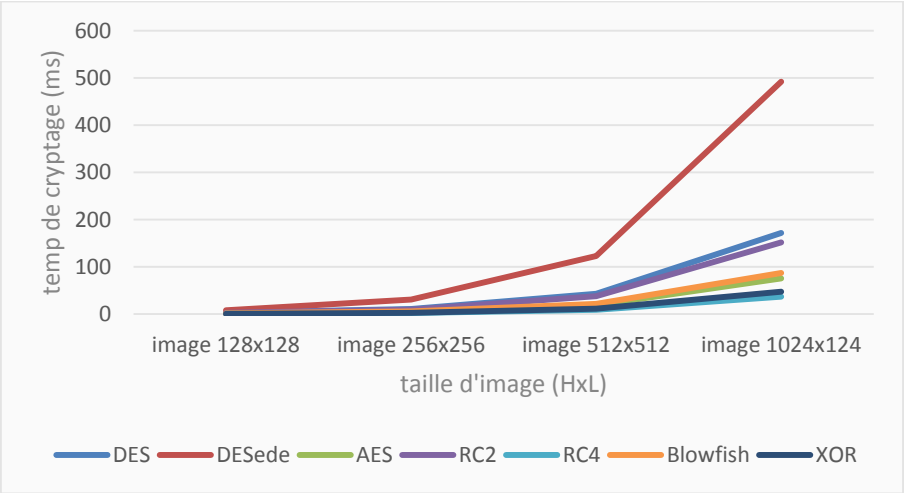


Figure 25 : Comparaison de temps de cryptage entre les algo symétrique

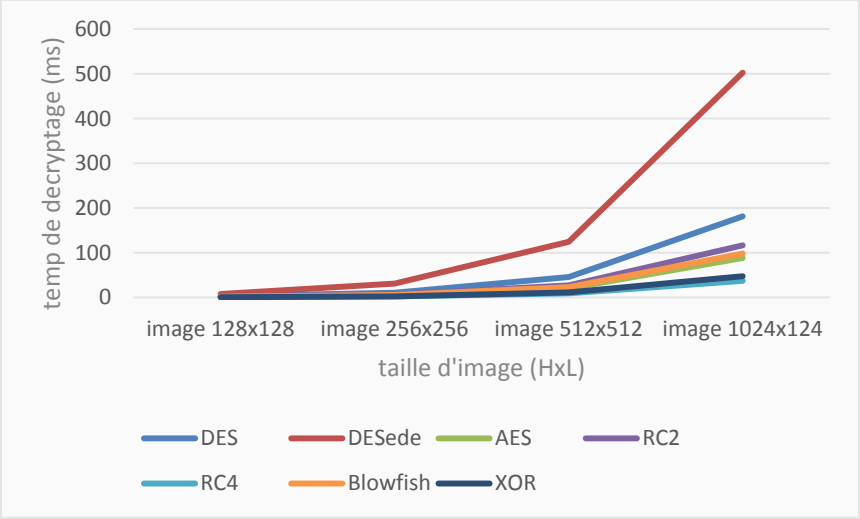


Figure 26 : Comparaison de temps de décryptage entre les algo symétriques

11.Résultat général

Les résultats de simulation pour ce point de compassion sont présentés (Figure 25*)* à l'étape de cryptage. Les résultats montrent la supériorité des algorithmes RC4 et XOR par rapport aux autres Algorithmes en termes de temps de traitement. Il peut également être noté ici ; que 3DES a une faible performance en termes de consommation d'énergie et le débit par rapport à DES. Il exige toujours plus de temps que DES en raison de ses caractéristiques de chiffrement de phase triple.

RSA Comparé à chiffrement symétrique, RSA impose une charge de calcul élevée, et tend à être beaucoup plus lent. Ainsi, il n'est généralement pas utilisé pour protéger les données de grande taille. Au lieu de cela, son force principale est sa capacité à établir un canal sécurisé sur un support non sécurisé.

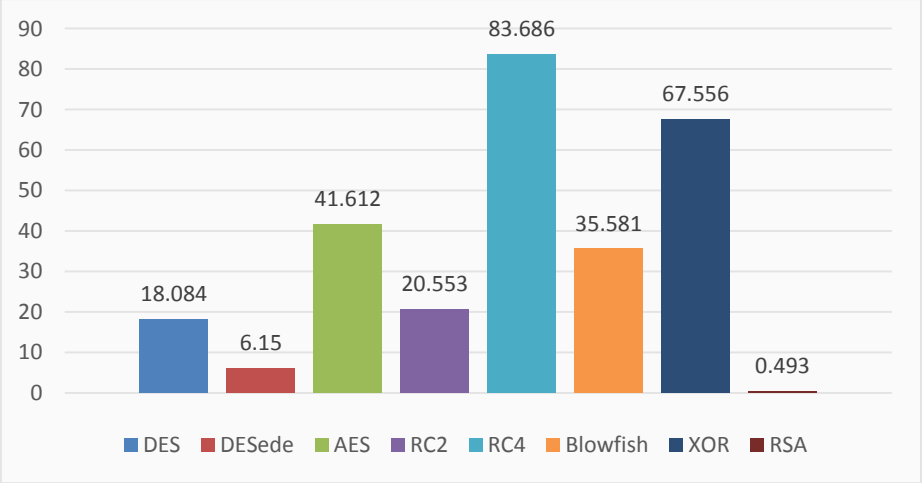


Figure 27 : vitesse de cryptage (KB/ms)

Les résultats de simulation pour ce point de compassion sont présentés (Figure 26) la phase de décryptage. Nous pouvons trouver dans le décryptage que RC4 et XOR sont les mieux que les autres algorithmes dans le débit et la consommation d'énergie. Enfin, Triple DES (3DES) nécessite encore plus de temps que DES.

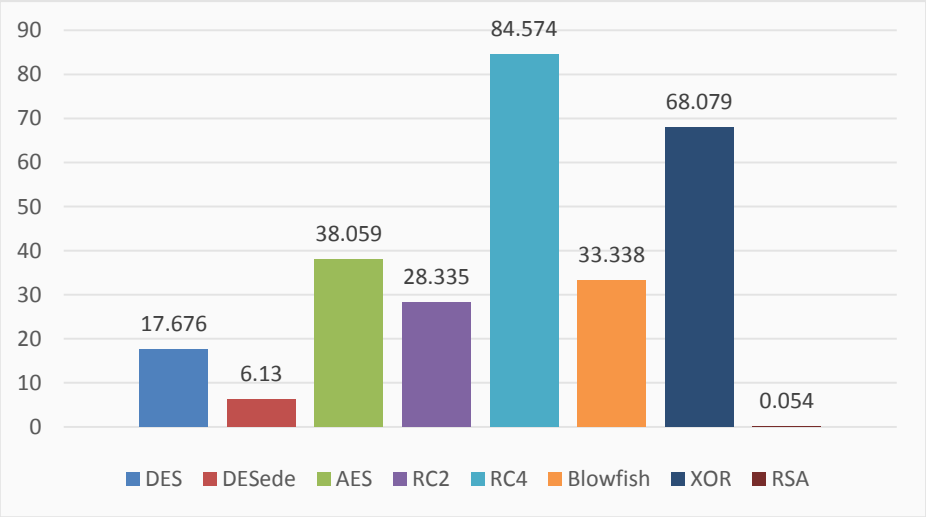


Figure 28 : vitesse de décryptage (KB/ms)

12. Résultat sur la vitesse de chiffrement et déchiffrement des images

Cette première partie présente une évaluation de la performance des algorithmes de chiffrement sélectionnés. Les algorithmes sélectionnés sont AES, DES, 3DES, RC2, RC4, Blowfish, XOR, RSA. Plusieurs points peuvent être conclus à partir des résultats de simulation.

D'abord ; il n'y a pas de différence significative entre les résultats de l'analyse du cryptage et décryptage.

Deuxièmement ; en cas de changement de la taille des donnée (résolution d'image), il a été conclu que RC4 a de meilleures performances que d'autres algorithmes de chiffrement commun utilisés, suivies par XOR.

Troisièmement ; Les résultats de l'évaluation a montré que Blowfish et AES a des performances décentes. Et comme ils n'ont pas des points faibles de sécurité connue jusqu'à présent, ils peuvent être considérés comme des excellents algorithmes de cryptage standard

Quatrièmement ; Le principal inconvénient de RSA, c'est qu'il est plus lent que le cryptage symétrique et nécessite plus de puissance de traitement pour crypter et décrypter les données. Mais pour obtenir les avantages des deux méthodes, une technique hybride est généralement utilisée. Dans cette technique, le chiffrement asymétrique est utilisé pour échanger la clé secrète, le cryptage symétrique est ensuite utilisé pour transférer des données entre l'émetteur et le récepteur.

En outre, nous constatons que 3DES a encore un manque de performance par rapport à l'algorithme DES.

Deuxième partie

13. Paramètre de comparaison

La méthode de matrice de cooccurrence est largement utilisée dans le monde du traitement d'image (Haralick et al. 1973). Elle présente une grande simplicité de mise en œuvre et donne de bons résultats sur la plupart des types d'images.

C'est ce qui justifie notre choix de l'utiliser pour évaluer la qualité de cryptage

14. Définition de matrice de cooccurrence

Du fait de leur richesse en information de texture, les matrices de cooccurrences sont devenues les plus connues et les plus utilisées pour extraire ces caractéristiques de textures.

Elles estiment des propriétés des images relatives à des statistiques de second ordre.

Une matrice de cooccurrence mesure la probabilité d'apparition des paires de valeurs de pixels situés à une certaine distance dans l'image. Elle est basée sur le calcul de la probabilité P (i, j, δ, θ) qui représente le nombre de fois où un pixel de niveau de couleur i apparaît à une distance relative δ d'un pixel de niveau de couleur j et selon une orientation θ donnée.

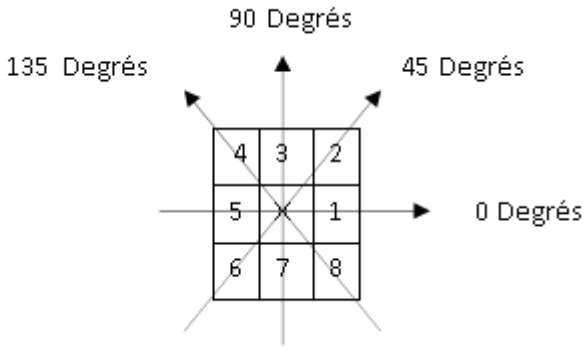


Figure 29: Plus proches voisins du pixel 'x' selon 4 directions

Les directions angulaires θ classiquement utilisées sont 0, 45, 90 et 135 degrés. Les relations de voisinage entre pixels, nécessaires au calcul des matrices, sont illustrées en (Figure 29) ; par exemple, les plus proches voisins de 'x' selon la direction θ =135 degrés sont les pixels 4 et 8. Les caractéristiques extraites à partir de ces matrices contiennent des informations notamment sur l'homogénéité, les dépendances linéaires entre les niveaux de gris, le contraste et la complexité de cette image.

Les matrices obtenues selon ces quatre directions sont alors calculées comme dans (1), (2), (3) et (4) où (*k*, *l*) sont les coordonnées d'un pixel de niveau de couleur $i \in [0, n^{max}-1]$ et (*m*, *n*) celles du pixel de niveau de couleur $j \in [0, n^{max}-1]$

$$P(i,j,\delta,0) = \left|\left\{((k,l),(m,n)) \in (N\times M)^2 tq(k-m=0,|l-n|=\delta,I_{k,l}=i,I_{m,n}=j)\right\}\right| \quad (1)$$

$$P(i,j,\delta,45) = \left|\left\{\begin{array}{l}((k,l),(m,n)) \in (N\times M)^2 tq \\ (k-m=\delta,l-n=-\delta) \vee (k-m=-\delta,l-n=\delta), I_{k,l}=i,I_{m,n}=j\end{array}\right\}\right| \quad (2)$$

$$P(i,j,\delta,90) = \left|\left\{((k,l),(m,n)) \in (N\times M)^2 tq(|k-m|=\delta,l-n=0,I_{k,l}=i,I_{m,n}=j)\right\}\right| \quad (3)$$

$$P(i,j,\delta,135) = \left|\left\{\begin{array}{l}((k,l),(m,n)) \in (N\times M)^2 tq \\ (k-m=\delta,l-n=\delta) \vee (k-m=-\delta,l-n=-\delta,I_{k,l}=i,I_{m,n}=j\end{array}\right\}\right| \quad (4)$$

La (Figure 30) montre un exemple de calcul des *P(i, j)* à partir d'une petite image 4×4 composée de quatre niveaux de gris (0, 1, 2, 3). Cet exemple se limite au cas δ =1 et θ = 0. L'élément (2, 3) de la matrice *P*(1, 0) est égal à 4, cela signifie qu'il existe 4 configurations dans l'image ou un pixel de niveau de gris 2 est séparé horizontalement d'un autre pixel de niveau de gris 3 par une distance 1. Ces configurations sont représentées en trait gris sur l'image.

0	0	1	2
0	1	3	2
0	2	3	2
1	2	3	0

Image

$$P(i,j,1,0°) = \begin{array}{c} \\ \\ 2 \\ \\ \end{array}\begin{array}{c} \\ \\ \\ 3 \\ \\ \end{array}\begin{pmatrix} 2 & 2 & 1 & 1 \\ 2 & 0 & 2 & 1 \\ 1 & 2 & 0 & 4 \\ 1 & 1 & 4 & 0 \end{pmatrix}$$

$$P(i,j,1,45°) = \begin{pmatrix} 2 & 1 & 0 & 0 \\ 1 & 2 & 1 & 0 \\ 0 & 1 & 0 & 5 \\ 0 & 0 & 5 & 0 \end{pmatrix}$$

$$P(i,j,1,90°) = \begin{pmatrix} 4 & 2 & 0 & 1 \\ 2 & 0 & 1 & 1 \\ 0 & 1 & 6 & 0 \\ 1 & 1 & 0 & 4 \end{pmatrix}$$

$$P(i,j,1,135°) = \begin{pmatrix} 0 & 1 & 2 & 2 \\ 1 & 0 & 1 & 1 \\ 2 & 1 & 0 & 2 \\ 2 & 1 & 2 & 0 \end{pmatrix}$$

Figure 30 : Exemple de matrices de cooccurrence construites à partir d'une image 4 × 4 composée de 4 niveaux de gris

La plupart des images sont codées sur 256 niveaux de gris, par conséquent, la taille des matrices de cooccurrence est de 256 × 256. On s'aperçoit ainsi que ces matrices comptabilisent une très grosse quantité d'informations difficile à exploiter directement. C'est pourquoi, un certain nombre d'auteurs comme Zucker [9] ont essayé d'extraire de l'information de ces matrices afin de mettre en évidence la structure des textures. Mais c'est Haralick et al [10] qui ont proposé les premiers 14 paramètres, caractérisant les textures, issus de ces matrices. Voici 6 paramètres considérés comme étant les plus utilisés et les plus pertinents [11] :

14.1. L'énergie $ENE = \sum_i \sum_j \left(P_{ij}(\delta,\theta)^2\right)$

Ce paramètre mesure l'uniformité de la texture. Il atteint de fortes valeurs lorsque la distribution des niveaux de gris est constante ou de forme périodique. Dans ce dernier cas, les valeurs élevées d'énergie sont obtenues pour les matrices $P(d, \theta)$ lorsque (d, θ) correspond à la période.

14.2. Le contraste $CST = \sum_i \sum_j \left((i-j)^2 P_{ij}(\delta,\theta)\right)$

La valeur en est d'autant plus élevée que la texture présente un fort contraste. Ce paramètre est fortement non corrélé à l'énergie.

14.3. L'entropie : $ENT = -\sum_i \sum_j \left(\log P_{ij}(\delta,\theta) P_{ij}(\delta,\theta)\right)$

Ce paramètre mesure le désordre dans l'image. Contrairement à l'énergie, l'entropie atteint de fortes valeurs lorsque la texture est complètement aléatoire (sans structure apparente). Elle est fortement corrélée (par l'inverse) à l'énergie.

14.4. La variance : $\mathrm{VAR} = \sum_i \sum_j ((i-\mu)^2 P_{ij}(\delta,\theta))$

La variance mesure l'hétérogénéité de la texture. Elle augmente lorsque les niveaux de gris différent de leur moyenne. La variance est indépendante du contraste.

14.5. La corrélation : $COR = \sum_i \sum_j \left(\frac{(i-\mu)(j-\mu)P_{ij}(\delta,\theta)}{\sigma^2}\right)$

Corrélation mesure la dépendance linéaire (relativement à (δ, θ)) des niveaux de gris de l'image. La corrélation n'est corrélée ni à l'énergie, ni à l'entropie.

14.6. La dissemblance

Ce paramètre mesure la dissemblance de l'histogramme (s'il est bien équilibré autour de sa moyenne ou bien s'il est plus orienté à gauche ou à droite par rapport à sa moyenne).

14.7. Le moment inverse : $IDM = \sum_i \sum_j \frac{P_{ij}(\delta,\theta)}{1+(i-j)^2}$

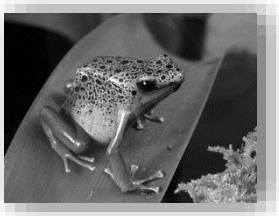
IDM (Inverse Différence Moment) mesure l'homogénéité de l'image. De nombreuses études ont été menées afin de caractériser, classifier, modéliser les textures à l'aide de ces paramètres ([12], [13], [14], [15]).

La méthode d'extraction de ces paramètres basée sur le calcul des matrices de cooccurrence est une des méthodes les plus proches de la notion de texture. Elles mettent effectivement en avant les relations qui existent entre les pixels de l'image en faisant intervenir l'aspect local (les niveaux de gris) et l'aspect spatial ((δ, θ)).

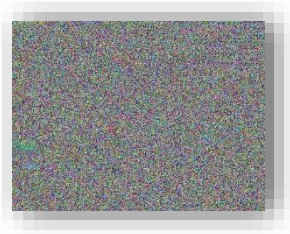
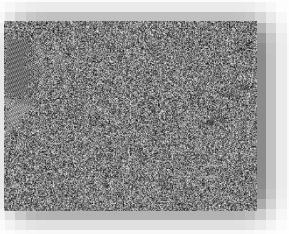
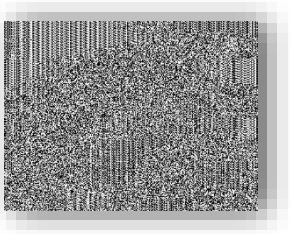
Dans la plupart des études, les auteurs calculent ces matrices de cooccurrence sur les images brutes ou filtrées, donc toujours à partir des niveaux de gris. Davis, Johns et Aggarwal [16] ont généralisé cette utilisation à tous les types d'information (incluant les niveaux de gris), ils ont transformé les images en codant certains pixels par une autre information que celle des niveaux de gris. Ensuite, ils ont calculé les matrices de cooccurrence à partir de ces nouvelles images codes.

15. Résultats d'évaluation

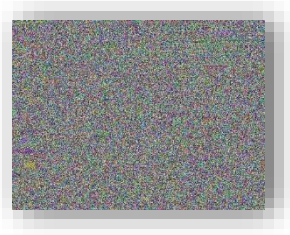
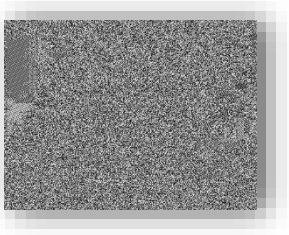
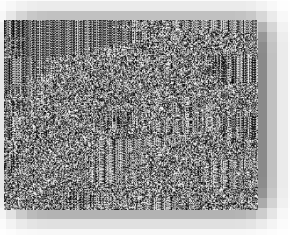
a. Les images proposées

image			
type	couleur	niveau de gris	noir et blanc
nom	nature	Grenouille	perroquet
CST	579.842	256.263	3634.283
COR	0.924	0.945	0.886
DIS	13.251	7.045	14.252
IDM	0.3164	0.4212	0.9441

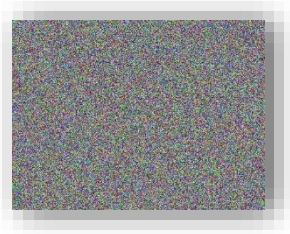
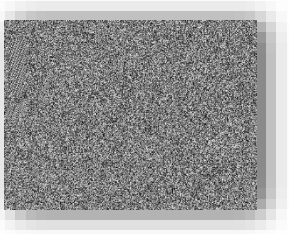
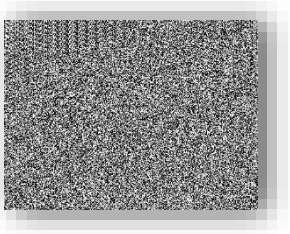
b. Résultats DES :

images			
nom	nature_DES	grenouille_DES	perroquet_DES
CST	5150.091	11112.402	31297.369
COR	0.01	-0.018	-0.019
DIS	58.205	86.512	129.059
IDM	0.0167	0.0116	0.1436

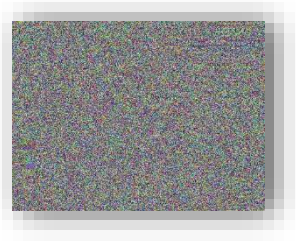
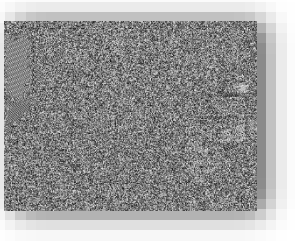
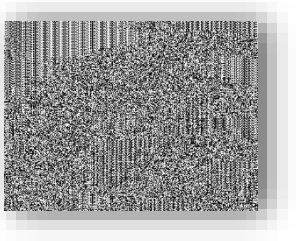
c. Résultats 3DES :

images			
nom	nature_3DES	grenouille_3DES	perroquet_3DES
CST	5146.939	10877.857	30904.316
COR	0.007	-0.009	-0.006
DIS	58.184	85.128	127.553
IDM	0.0167	0.0116	0.1455

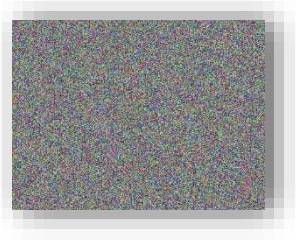
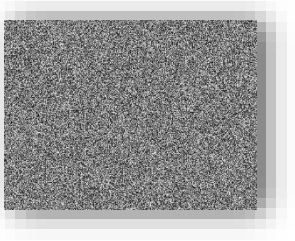
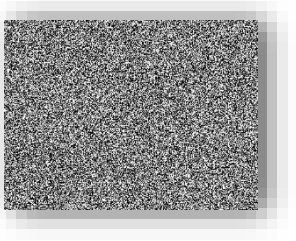
d. Résultats AES :

images			
nom	nature_AES	grenouille_AES	perroquet_AES
CST	5119.255	11206.749	30720.671
COR	0.016	-0.013	-0.002
DIS	58.066	86.384	126.988
IDM	0.0165	0.0121	0.1471

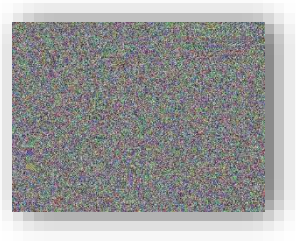
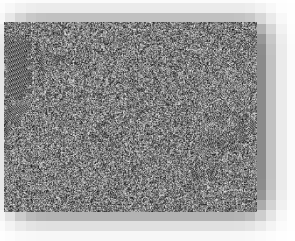
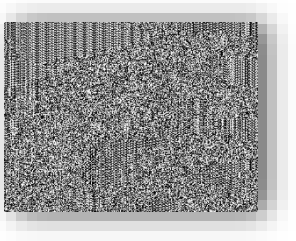
e. Résultats RC2 :

images			
nom	nature_RC2	grenouille_RC2	perroquet_RC2
CST	5148.213	11052.687	31739.64
COR	0.011	-0.009	-0.034
DIS	58.216	86.006	130.821
IDM	0.0166	0.0114	0.1428

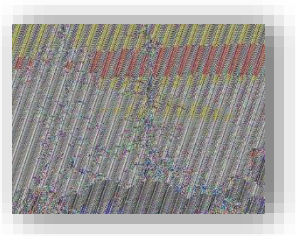
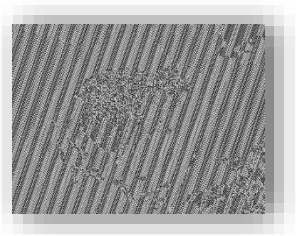
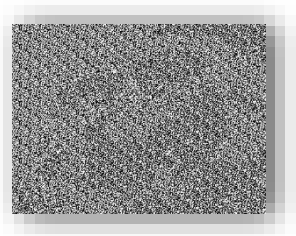
f. Résultats RC4 :

images			
nom	nature_RC4	grenouille_RC4	perroquet_RC4
CST	5129.817	11119.316	30565.142
COR	0.014	-0.008	0.003
DIS	58.121	86.097	126.364
IDM	0.0165	0.0118	0.1471

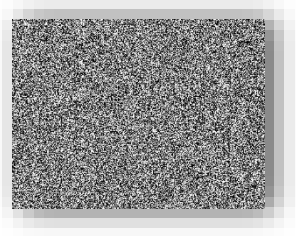
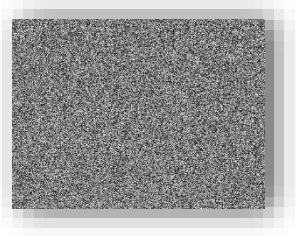
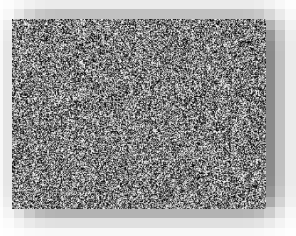
g. Résultats Blowfish :

images			
nom	nature_Blowfish	grenouille_Blowfish	perroquet_Blowfish
CST	5130.638	11436.975	30068.076
COR	0.01	-0.02	0.02
DIS	58.136	87.442	124.335
IDM	0.0163	0.0116	0.1513

h. Résultats XOR :

images			
nom	nature_XOR	grenouille_XOR	perroquet_XOR
CST	4449.836	12187.109	30609.205
COR	0.076	-0.14	0.003
DIS	52.676	92.233	126.381
IDM	0.0203	0.0085	0.1502

i. Résultats RSA :

images			
nom	nature_RSA	grenouille_RSA	perroquet_RSA
CST	5181.501	11033.21	30725.083
COR	0.006	0.00005	-0.001
DIS	58.422	85.696	126.968
IDM	0.0165	0.0121	-0.0018

16. Comparaison entre les algorithmes

a. Le Contraste :

Figure 31 : diagramme de contrast

Tous les algorithmes semblent de donner des résultats similaire, mais ce qui est remarqué c'est que XOR avoir le constrast le plus bas dans l'image en couleur alors qu'il possède le constrast le plus haut dans l'image en niveaux de gris.

On peut remarquer aussi que le meilleur résultat dans l'image en couleur est celle de RSA ainsi que RC2 dans l'image noire et blanc.

b. La Corrélation :

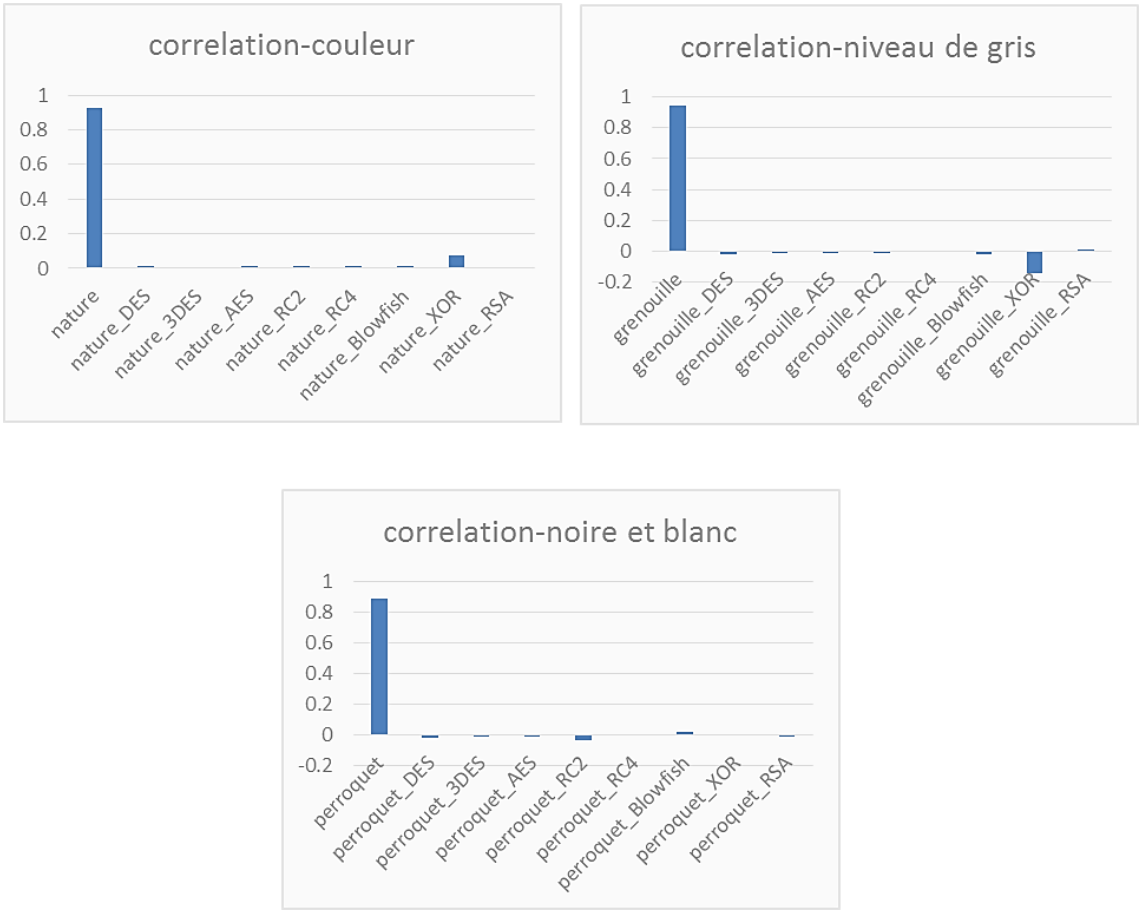


Figure 32 : diagramme de correlation

En vois ici que tous les algorithmes a donnée des résultats respectueux concernant la corrélation sauf pour XOR dans le cas d'image en couleur contrairement à l'image en niveau de gris où il avait le meilleur résultat.

En peut vois aussi que RC2 a le meilleur résulte dans l'image noire et blanc ainsi que RSA dans l'image en couleur.

c. La Dissemblance :

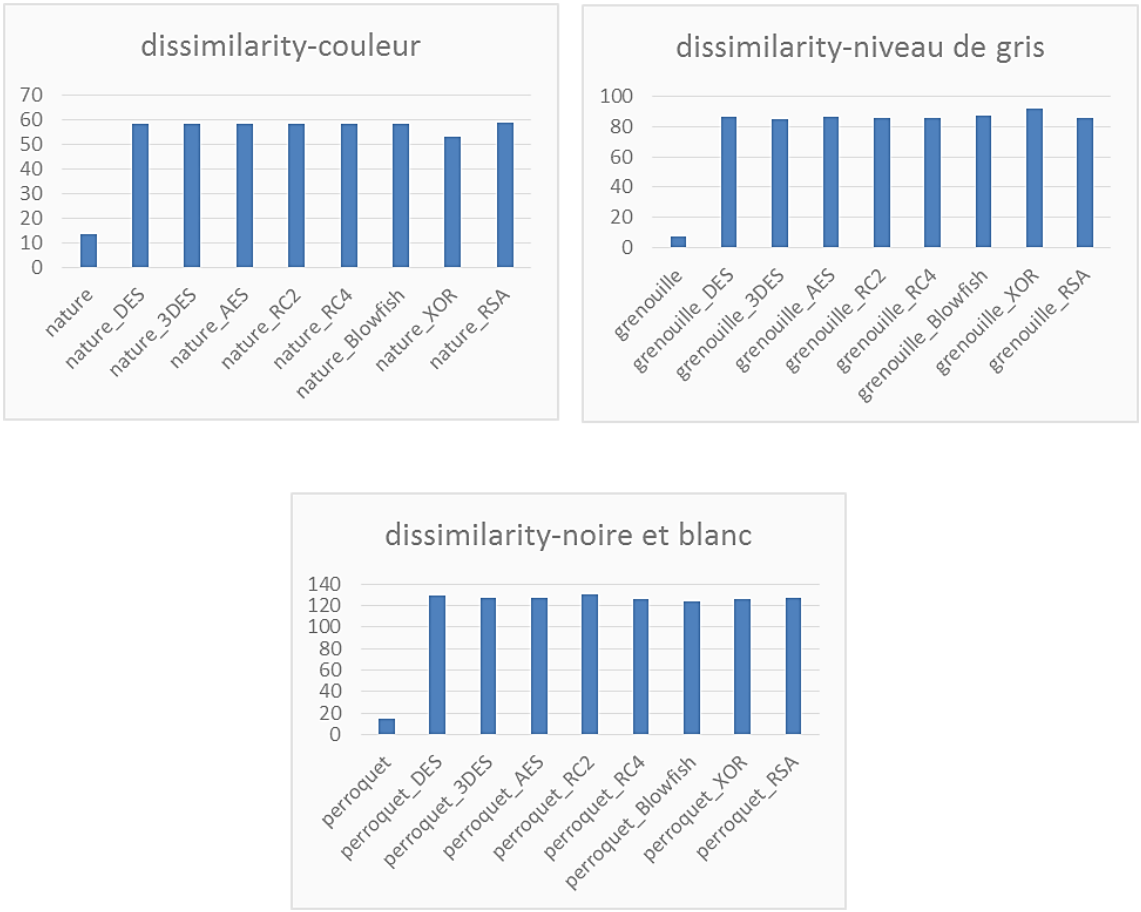


Figure 33 : diagramme de dissemblance

Tous les algorithmes ont largement changé les couleurs des pixels qui donne des histogrammes totalement différentes de celle donnée par l'image originale ce qui justifier les résultats obtenu par l'analyse de La Dissemblance.

d. L'homogénéité ou moment différentiel inverse :

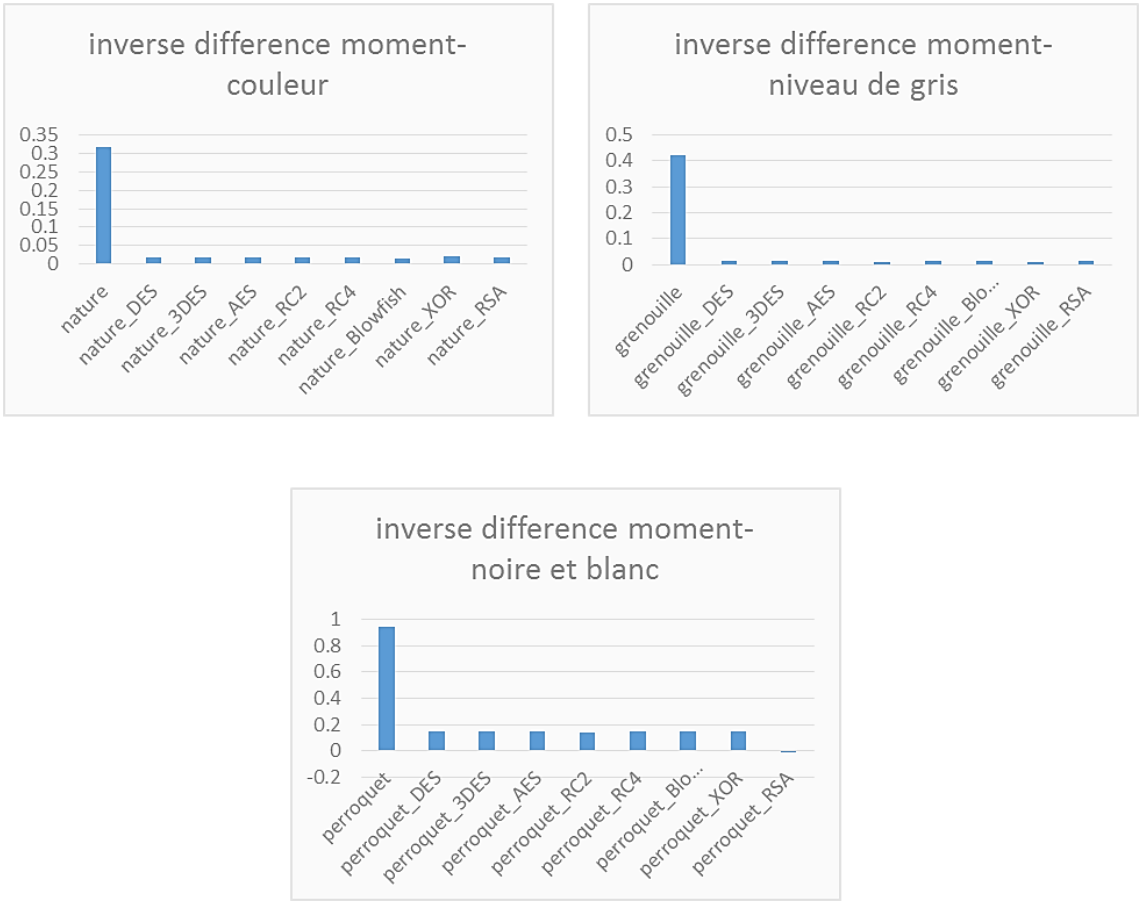


Figure 34 : diagramme de L'homogénéité

Faible valeur de ce paramètre signifie qu'il y a moins de régions homogènes par rapport à l'image originale.

Le meilleur résultat été donne par Blowfish

Dans l'image en couleur et par XOR dans l'image en niveau de gris et par RSA dans l'image en noir et blanc.

17. Conclusion

A partir des résultats obtenus on peut conclure que les algorithmes les plus performants dans le cryptage des images en couleur sont RSA et Blowfish mais puisque RSA a un temps d'exécution considérablement lent en prend Blowfish comme le meilleur choit, alors que XOR est le moins performant dans ce cas.

On peut conclure aussi que contrairement aux images en couleur, XOR est très efficace et peut être suffisante pour le cryptage des images en niveau de gris.

Dans le cas d'image noir et blanc on a multiple candidat d'être le meilleur algorithme : RSA, DES et RC2 mais le choix évident c'est RC2 à cause de ça vitesse.

Finalement chaque algorithme a des points forts et des points faibles mais déférente algorithme sont favorables dans déférente situation.

Conclusion générale

La sécurité des images est un domaine en pleine expansion, vu la quantité importante d'images qui circule dans divers réseaux de télécommunications, particulièrement Internet. Cette large utilisation des images, impose la mise en œuvre de systèmes cryptographiques capable de fournir un bon niveau de sécurité tout en assurant une performance dans la qualité de l'opération de chiffrement.

Le cryptage s'est imposé petit à petit, où il garantit un bon compromis entre niveau de sécurité et vitesse de traitement. Nous sommes intéressés dans notre projet de fin d'étude aux algorithmes cryptographiques appliqués dans la protection des images dans un réseau de communication.

Au cours de cette mémoire, nous avons étudié et implémenté les différents algorithmes de chiffrements dans une architecture Client/serveur.

Nous sommes basés sur deux critères de comparaisons pour évaluer l'opération de chiffrement qui sera appliqué sur des images :

Le premier critère c'est La vitesse de l'opération de chiffrement.

et le deuxième critère c'est La qualité visuelle de l'opération de chiffrement.

D'après les résultats obtenus avec les différentes comparaisons effectuées nous pouvons conclure que les algorithmes symétriques sont plus performants en termes de vitesse de chiffrement par rapport aux algorithmes à clé publique mais avec une faible qualité visuelle dans l'opération de chiffrement, par contre les algorithmes asymétriques donnent une meilleure qualité de chiffrement mais ils nécessitent un temps considérable dans le cryptage des images.

Comme perspective de notre travail, nous souhaitons pour les prochains projets de fin d'étude, élargir notre travail pour chiffrer des images médicaux.

Bibliographie

Référence

[1] A Jones and J Ohlund , Network Programming for Microsoft Windows Second Edition ,MSPress2002.

[2] M. Stamp, Informatique security principle and practice , Wiley 2013.

[3] B. Prenee ,Understanding Cryptography , Springer 2010.

[4] D. Stinson , Cryptography: Theory and Practice ,CRC Press 1995.

[5] P. Noizat , Bitcoin book , Spring 2012.

[6] S. Bruce, Cryptographie appliquée- Algorithmes, protocoles,Wiley 1997.

[7] W. Stallings, Cryptography-network-security-5th-edition, Prentice Hall 2011.

[8] See R. Steinmetz, K. Nahrstedt. Multimedia: Computing, Communications & Applications. Innovative Technology Series. Prentice Hall P T R. 1995.

[9] S.W. Zucker and D. Terzopoulos. Finding structure cooccurrence matrices for texture analysis. Computer Vision Graphics and Image Processing, 12:286–308, 1980.

[10] R.M. Haralick. Statistical and structural approachs to texture. Proceedings of the IEEE In Proceedings of the IEEE, Vol. 67, No. 5, pages 786–804, may 1979.

[11] A. Baraldi and F. Parmiggiani. An investigation of the textural characteristics associated with gray level cooccurrence matrix statistical parameters. IEEE Transactions on Geoscience and Remote Sensing, 33(2):293–304, march 1995.

[12] J. Parkkinen, K. Selk inaho, and E. Oja. Detecting texture periodicity from the cooccurrence matrix. Pattern Recognition Letters, Vol. 11, pages 43–50, 1990.

[13] C.C. Gotlieb and H.E. Kreyszig. Texture descriptors based on co-occurence matrices. Computer Vision, Graphics, and Image Processing, Vol. 1, pages 70–86, 1990.

[14] G. Lohmann. Analysis and synthesis of textures: a co-occurrence-based approach. Computers & Graphics, Vol. 19, Issue 1, pages 29-36, January-February 1995.

[15] I.M. Elfadel and R.W. Picard. Gibbs random fields, cooccurrences, and texture modeling. IEEE Trans. on Pat. Anal. and Mach. Intel., Vol. 16, Issue 1, pages 24–37, january 1994.

[16] L.S. Davis, S.A. Johns, and J.K. Aggarwal. Texture analysis using generalized cooccurrence matrices. IEEE Trans. on PAMI, 1(3):251–259, july 1979.

Référence site web

[s1] nt.impmc.upmc.fr/impmc/Enseignement/ye/informatique/systemes/chap7/73.html

[s2] http://www.newagepublishers.com/samplechapter/001709.pdf

[s3] w3.polytech.univ-montp2.fr/~karen.godary/M1/Trans_Client_Serveur.pdf

[s4] http://www.marthendiaye.com/cours/Coursclserv.pdf

[s5] http://www.cryptage.org/chiffre-cesar.html

[s6] http://www.cryptage.org/cle-secrete.html#chiffrement

[s7] http://www.uqtr.ca/~delisle/Crypto/prives/blocs_des.php

[s8] http://villemin.gerard.free.fr/Crypto/RSA.htm

[s9] http://www.mycube.fr/quest-ce-quune-image-numerique

[s10] http://www.montpellier.iufm.fr/technoprimaire/c2i/revisions/formats_image.pdf

ملخص

الهدف الرئيسي من مشروعنا هو دراسة خوارزميات التشفير المختلفة المستخدمة لتشفير وحماية البيانات المتداولة في شبكات الحاسوب.

وينقسم التشفير الحديث إلى فئتين:

التشفير المتناظر الذي يستخدم نفس المفتاح لتشفير وفك تشفير الرسائل وله ميزة كونه سريع

التشفير غير المتناظر الذي يتطلب مفاهيم أساسية في الرياضيات والذي يستخدم مفتاح عام للتشفير ومفتاح خاص لفك التشفير والذي لديه ميزة أمن المفاتيح.

Résumé

L'objectif principal de notre projet est d'étudier les différents algorithmes de cryptage utilisés pour le chiffrement et la protection des données circulée dans les réseaux informatique.

La cryptographie moderne se décomposés en deux classes :

La cryptographie symétrique qui utilise la même clé pour chiffrer et déchiffrer des messages et qui a l'avantage d'être rapide.

Le cryptage asymétrique nécessitant des notions essentiel en mathématiques et qui utilise une clé publique pour chiffrer et une clé privée pour déchiffrer et qui a l'avantage de la sécurité des clés

Abstract

The main goal of our project is to study the different encryption algorithms used to encrypt and protect data that circulate in computer networks.

Modern cryptography is decomposed into two classes:

Symmetric cryptography uses the same key to encrypt and decrypt messages and has the advantage of being fast.

Asymmetric encryption requiring essential concepts in mathematics and which uses a public key to encrypt and a private key to decrypt and which has the advantage of key security.