Abou Bekr Belkaid University
Tlemcen ALgeria
جامعة أبي بكر بلقايد
تلمسان الجزائر

**République Algérienne Démocratique et Populaire**

**Université Abou Bakr Belkaid– Tlemcen**

**Faculté des Sciences**

## Mémoire de fin d'études

**Pour l'obtention du diplôme de Master en Informatique**

***Option:*** Réseaux et Systèmes Distribués (RSD)

*Thème*

# Le cryptage des vidéos par l'algorithme AES

**Réalisé par :**

- **M**elle Baaziz Rekia
- **M**elle Ghadi Fatiha

| | |
|---|---|
| - Mr.Lehsaini Mohamed | **Président** |
| - Mme. Meziane Tani Souad | **Examinateur** |
| - Mr. Benaissa Mohamed | **Encadreur** |

**Année universitaire**

2023-2024


بسم الله الرحمن الرحيم
١٤٣٨

# Remerciements

*Nous tenons avant tout de remercier le bon DIEU qui nous a donné la volonté et le courage pour réaliser ce modeste travail.*

*Du fond du cœur nous remercions nos chers parents qui nous ont toujours guidé, encouragé et qui ont fait de leurs mieux pour que nous arrivons là aujourd'hui.*

*Nous remercions vivement Mr BENAISSA Mohammed notre Encadreur pour la précieuse assistance, sa disponibilité et son soutien qu'il nous à apporter tout au long de ce projet.*

*Que les membres de jury trouvent ici l'expression de nos sincères remerciements pour l'honneur qu'ils nous font en acceptant de juger notre travail.*

*Nous remercions* sincèrement tous les enseignants qui nous ont enseigné et utilisé leurs compétences pour nous soutenir dans notre poursuite apprendre.

# Dédicace

*À nos martyrs héroïques*

*Aux créateurs de la gloire du 7 octobre*

*Au fier peuple de Gaza*

*Tout amour et respect.*

*Rekia*

# Dédicace

*A mes chers parents,*

*A mes frères et sœurs,*

*A tous mes ami(e)s,*

*A ceux qui m'aiment,*

*Je dédie ce travail.*

*Fatiha*

# Table des matières


Liste de Figures

Liste des tableaux

INTRODUCTION GÉNÉRAL ..... 1

CHAPITTRE 1 : les différents algorithmes de chiffrement symétriques et asymétriques

**1 Introduction** ..... 4

**2 Définition de la cryptographie** ..... 4

**3 A quoi sert la CRYPTOGRAPHIE** ..... 4

3.1 Confidentialité ..... 4

3.2 Authentification ..... 4

3.3 Intégrité ..... 5

3.4 Non-répudiation ..... 5

**4 La cryptographie** ..... 5

4.1 Cryptographie symétrique ..... 5

4.2 Cryptographie Asymétrique ..... 6

4.3 Cryptage symétrique vs asymétrique ..... 6

**5 Les différentes Algorithmes ..... 7**

5.1 Algorithmes symétriques ..... 7

5.1.1 Algorithme DES(Data encryption standard) ..... 7

5.1.2 Le Triple DES ..... 9

5.1.2.1 Définition ..... 9

5.1.3 IDEA ..... 10

5.1.4 BLOWFISH ..... 11

5.1.5 RC4 ..... 12

5.2 Algorithme Asymétriques ..... 12

5.2.1 RSA ..... 12

5.2.2 Protocole d'échange de clé publique DIFFI-HELLMAN ..... 13

**6 Conclusion** ..... 14

CHAPITER2 : les concepts et les notions de base des vidéos

**1 Introduction** ..... 16

**2 Type des vidéos** ..... 16

2.1 Type analogique ..... 16

2.2 Vidéo numérique ..... 17

**3 Couleur de l'espace** ..... 17

3.1 Représentation RVB (Rouge-Vert -Bleu) ..... 17

3.2 Représentation YCbCr ..... 18

**4 Formats d'échantillonnage** ..... 19

**5 Formats vidéo numériques** ..... 20

**6 Conteneurs de références** ..... 21

**7 Compression vidéo numérique** ..... 22

7.1 Types de compression ..... 22

7.1.1 Compression sans perte ..... 22

7.1.2 Compression avec perte ..... 23

**8 Normes de codage vidéo** ..... 28

**9 Conclusion** ..... 31

CHAPITRE 3 : Le principe de base de fonctionnement de l'algorithme AES

**1 INTRODUCTION** ..... 33

**2 FONCTIONNEMENT D'AES** ..... 33

2.1 SubBytes ..... 34

2.2 ShiftRows ..... 34

2.3 MixColumns ..... 34

2.4 KeyExpansion ..... 35

2.5 AddRoundKey ..... 36

**3 CHIFFREMENT ET DECHIFFREMENT** ..... 36

**4 CHIFFRAGE** ..... 37

4.1 PROSESSUS DE CHIFFREMENT ..... 37
4.2 LA TRONSFORMATION subBytes ..... 38
4.3 LA TRANSFORMATION ShiftRows ..... 40
4.4 LA TRONSFORMATION MixColumns ..... 41
4.5 Addition de la transformation de clé ronde ..... 42
4.6 Key schedule Expansion ..... 43

**5 DÉCRYPTAGE ..... 44**
5.1 Processus de décryptage ..... 44
5.2 Transformation InvSubBytes ..... 45
5.3 Transformation InvShiftRows ..... 46
5.4 Transformation InvMixColumns ..... 47

**6 Conclusion ..... 48**
Chapitre 4 ..... 50
ETUDE LA BIBLIOTYEQUE FFmpeg ..... 50

**1 Introduction ..... 51**

**2 Présentation de l'outil FFmpeg ..... 51**
2.1 Définition ..... 51
2.2 Installation du FFmpeg ..... 51
2.2.1 Installation sous Windows ..... 51
2.2.2 Installation Sous linux ..... 52
2.3 Principe de fonctionnement ..... 53
2.4 Les bibliothèques ..... 53
2.4.1 Définition ..... 53
2.4.2 Bibliothèques utilisées par FFmpeg ..... 54
2.5 Les formats ..... 54
2.6 Les codecs ..... 55
2.7 Les filtres ..... 56
2.8 Principales Commandes FFmpeg ..... 56
2.9 Liste des traitements dont le FFmpeg est capable ..... 57
2.10 CRF (Constant Rate Factor) ..... 59
2.11 QP (quantization parameter) ..... 59

**3 Conversion de Formats Vidéo et Audio Utilisant FFmpeg.....59**

3.1 Commande de base.....59

3.1.1 Conversion de MKV en MP4..... 60

3.1.2 Conversion de MOV en MP4..... 60

3.1.3 Conversion de WebM en MP4 .....61

3.1.4 Conversion de MP4 en MP3 .....61

3.2 Commande avancée.....61

3.2.1 Changement de Codec Vidéo/Audio..... 62

3.2.2 AAjustement de la Qualité Vidéo/Audio ..... 62

**4 Conclusion .....63**

CHAPITTRE 05 : IMPLEMENTATION DE L'ALGORITHME AES

**1 Introduction.....65**

**2 Les Algorithmes comparés .....65**

2.1 AES .....65

2.2 BLOWFISH.....65

2.3 XOR 65

2.4 MASK 66

**3 Présentation de l'Environnement et du Matériel et Outils Utilisés 66**

3.1 GTK..... 66

3.1.1 Définition..... 66

3.1.2 l'iInstallation..... 66

3.2 GStreamer..... 66

3.2.1 Definition..... 66

3.2.2 Installation.....67

3.3 OpenSSL.....67

3.3.1 Définition.....67

3.3.2 Installation.....67

3.4 FFMPEG.....67

3.4.1 Définition.....67

3.4.2 Installation.....67

**4 Les paramètres du système ..... 68**

**5 Etude comparative entre les différents algorithmes de cryptage 69**

5.1 Critère sur la vitesse et le temps de chiffrement .....72

de chaque algorithme.....72

5.2 Critère sur la qualité de l'opération de chiffrement de chaque algorithme ..73

**6 F Présentation de l'application .....74**

**7 Conclusion .....79**

Conclusion générale..... 80

Résumé .....81

Abstract.....81

**ملخص**.....81

Bibliographie..... 82

# Liste de Figures

Figure 1.1 : Schéma de fonctionnement de la cryptographie symétrique ..... 5

Figure 1.2 : Schéma de fonctionnement de la cryptographie asymétrique ..... 6

Figure 1.3 : fonctionnement du DES ..... 9

Figure 1.4 : Triple DES ..... 10

Figure 1.5 : Fonctionnement d'IDEA ..... 11

Figure 2:1 : Les types de balayage ..... 16

Figure 2:2 : Représentation des couleurs RVB ..... 18

Figure 2.3 : Profils d'échantillonnage 4 : 2 : 0, 4 : 2 : 2 et 4 : 4 : 4. ..... 20

Figure 2.4 : L'évolution des formats de la vidéo numérique: **de SD au 8K**. ..... 21

Figure 2.5 : Redondances dans la vidéo ..... 24

Figure 2.6 : Une vidéo MPEG est une séquence de groupe d'images (GOP) ..... 25

Figure 2.7 : Exemple de groupe d'images avec l'ordre d'affichage. ..... 26

Figure 2.8 : Étapes de codage typiques utilisées dans la compression MPEG ..... 26

Figure 2.9 : Evolution des normes de codage vidéo ..... 28

Figure 3.1 : Opération ShiftRows ..... 34

Figure 3.2 : Calcul de MixColumns ..... 35

Figure 3.3 : Schéma de fonctionnement de l'étape KeyExpansion ..... 36

Figure 3.4 : Etapes de l'algorithme AES ..... 37

Figure 3.5 : Le processus de chiffrement de l'AES ..... 38

Figure 3.6 : Notation matricielle de s-box ..... 39

Figure 3.7 : Application de S-box à chaque Byte de l'État ..... 39

Figure 3.8 : Valeurs S-box pour toutes les 256 combinaisons ..... 40

Figure 3.9 : Décalage cyclique des trois dernières ..... 41

Figure 3.10 : Mélange de colonnes de l'État ..... 42

Figure 3.11 : Fonctionnement OU exclusif des mots ..... 43

Figure 3.12: Processus de Décryptage ..... 45

Figure 3.13: Application de la boîte S inverse à chaque octet de l'État ..... 46

Figure 3.14 Valeurs S-box inverses pour toutes les 256 combinaisons ..... 46
Figure 3.15 : Décalage cyclique inverse des trois dernières rangées de l'État ..... 47
Figure 3.16 Opération de Inverse Mix Column sur l'état ..... 48
Figure 4:1: Principe de fonctionnement du FFmpeg ..... 53
Figure 4:2: Résultat de la commande ffmpeg -formats ..... 55
Figure 4:3: Principaux codecs du ffmpeg ..... 55
Figure 5:1 : Comparaison temps de chiffrement des 03 Vidéos ..... 72
Figure 5:2: la qualité de chiffrement des vidéos ..... 73

## Liste des tableaux

Tableau 2-1 : Les différents formats de la vidéo numérique ..... 20

Tableau 2-2 : Conteneurs de vidéo. ..... 22

Tableau 2-3 : Les caractéristique de la norme h.264/avc. ..... 30

Tableau 3-1 : Tableau du nombre d'itérations par rapport à la clé..... 33

Tableau 3-2 : Bloc-clé- Combinaisons rondes .. ..... 44

Tableau 4-1 : Signification des commandes ..... 58

Tableau 4-2 : Options d'encodage de la vidéo ..... 58

Tableau 4-3 : Options d'encodage de l'audio..... 59

Tableau 5-1 : Les paramètres du système ..... 68

Tableau 5-2 : Résultats Comparatifs du Chiffrement par AES,BLOWFISH ,XOR et MASK- Vidéo 01- ..... 69

Tableau 5-3 : Résultats Comparatifs du Chiffrement par AES,BLOWFISH ,XOR et MASK- Vidéo 02- ..... 70

Tableau 5-4 : Résultats Comparatifs du Chiffrement par AES,BLOWFISH ,XOR et MASK- Vidéo 03- ..... 71

Tableau 5-5 : Temps de cryptage ..... 72

Tableau 5-6 : la qualité de chiffrement des vidéos ..... 73

## Listes des acronymes et abréviations

**AES** : Advanced Encryption Standard

**DES** : Data encryption standard

**IDEA** : International Data Encryption Algorithm

**RVB** : Rouge Vert Bleu

**MPEG** : Moving Pictures Experts Group

**XOR** : Exclusive OR

**DCT** : Discrete Cosine Transform

**FFMPEG** : Fast Forward MPEG

**ECB** : Electronic Codebook :

**CBC** : Cipher Block Chanining

**CFB** : Cipher Feedback

**OFB** : Output Feedback

**IEC/ISO** : International Electrotechnical Commission /International Organization for Standardization

**ITU-T** : International Telecomminication Union-Telecommunication Standardization Sector

**RC4** : Rivest Cipher 4

**RSA** : Rivest-Shamir-Adleman

**RLE** : Run-Length Encoding

**AVC** : Advanced Video Coding

**CABAC** : Context- Adaptive Binary Arithmetic Coding

**CAVLC** : Context-Adaptive Variable Length Coding

**NIST** : Institut national des normes et de la technologie

**FIPS** : norme fédérale de traitement de l'information

**CRF** : Constant Rate Factor

**QP** : quantization parameter

**MKV** Matroska Video

# INTRODUCTION GÉNÉRALE

L'augmentation exponentielle de la production et de la diffusion de contenus vidéo numériques, couplée à la nécessité croissante de protéger ces données contre des accès non autorisés, a mis en lumière l'importance cruciale du cryptage des vidéos. Parmi les diverses techniques de cryptage disponibles, l'algorithme de cryptage symétrique AES (Advanced Encryption Standard) s'est distingué comme une méthode efficace et largement adoptée pour sécuriser les données numériques, y compris les vidéos.

Dans le contexte du cryptage des vidéos, l'AES présente plusieurs avantages significatifs. Premièrement, il est capable de traiter de grandes quantités de données à une vitesse élevée. Deuxièmement, grâce à sa robustesse contre les attaques cryptographiques connues, l'AES garantit une protection fiable des contenus vidéo contre les tentatives de décryptage non autorisées. Troisièmement, son implémentation est largement supportée par le matériel moderne, ce qui permet une intégration facile dans les systèmes de traitement vidéo.

L'application de l'AES au cryptage des vidéos implique généralement de diviser la vidéo en blocs de données de taille appropriée, puis de chiffrer chaque bloc indépendamment en utilisant une clé symétrique. Ce processus assure que même si une partie de la clé ou du texte chiffré est compromise, il reste extrêmement difficile pour un attaquant de déchiffrer l'intégralité de la vidéo sans la clé appropriée.

Le premier chapitre fournit une étude des différents algorithmes de chiffrement symétriques et asymétriques, le deuxième chapitre se concentre sur l'étude des concepts et les notions de base des vidéos.

Le troisième chapitre on a étudié le principe de base de fonctionnement de l'algorithme AES.

Le quatrième chapitre c'est étude de la bibliothèque FFMPEG et Enfin, le dernier chapitre Implémentation de l'algorithme AES avec un test et une étude comparative sur les performances de cet algorithme avec d'autre algorithme simple par exemple Blowfish, Mask et XOR.

# *CHAPITTRE 1 : Les différents algorithmes de chiffrement symétriques et asymétriques*

# 1 Introduction

La cryptographie joue un rôle essentiel dans la sécurisation des communications et des données numériques. Elle permet de garantir la confidentialité, l'intégrité et l'authentification des informations échangées. Au cœur de la cryptographie se trouvent les algorithmes de Chiffrement, qui sont des techniques mathématiques permettant de rendre les données inintelligibles pour toute personne non autorisée.

# 2 Définition de la cryptographie

La cryptographie, en tant qu'art du chiffrement et du codage des messages, s'est développée pour devenir une discipline à part entière de nos jours. En fusionnant les domaines des mathématiques, de l'informatique, et parfois même de la physique, elle répond à un besoin fondamental des sociétés depuis leur existence : la préservation du secret. Que ce soit pour prévenir les conflits, assurer la protection des populations, ou garantir la confidentialité des informations, le recours à la cryptographie est parfois indispensable pour dissimuler des éléments cruciaux.**[1]**

# 3 A quoi sert la CRYPTOGRAPHIE

Les communications échangées entre A et B sont sujettes à un certain nombre de menaces. La cryptographie apporte un certain nombre de fonctionnalités permettant de pallier ces menaces, pour Confidentialité, Authentification, Intégrité, Non-répudiation. **[1]**

## 3.1 Confidentialité

La confidentialité consiste à empêcher l'accès aux informations qui transitent à ceux qui n'en sont pas les destinataires. Ils peuvent lire les messages cryptés transmis sur le canal mais ne peuvent pas les déchiffrer.

## 3.2 Authentification

Il faut pouvoir détecter une usurpation d'identité. Par exemple, A peut s'identifier en prouvant à B qu'elle connaît un secret S qu'elle est la seule à pouvoir connaitre.

### 3.3 Intégrité

Il s'agit de vérifier que le message n'a pas subi d'altérations lors de son parcours. Elle concerne plutôt une modification volontaire et malicieuse e l'information provoquée par un tiers lors de transfert sur le canal. Ces modifications sont en générales masquées par le tiers pour être difficilement détectables.

### 3.4 Non-répudiation

C'est une protection protagonistes d'un échange entre les informations, et non plus contre un tiers. Si A envoie un message M, elle ne doit pas pouvoir prétendre ensuite devant B qu'elle ne l'a pas fait, ou alors qu'elle a envoyé M'et que le message a été mal compris.

## 4 La cryptographie

### 4.1 Cryptographie symétrique

La cryptographie symétrique, également connue sous le nom de cryptographie à clé secrète, représente la forme la plus ancienne de cryptographie (. Ce type de chiffrement repose sur l'utilisation d'une clé secrète, bien que certains chiffrements symétriques, comme le chiffre de César, ne nécessitent pas de clé. Dans les systèmes utilisant une clé, le processus est le suivant : l'émetteur chiffre les données à l'aide d'une clé, généralement une chaîne de caractères. Sans cette clé, il devient extrêmement difficile, voire impossible (selon le niveau de sécurité du chiffrement et la complexité de la clé), de déchiffrer le message chiffré pour retrouver le message original. Ainsi, l'émetteur doit partager la clé avec les destinataires s'il souhaite que ces derniers puissent lire le message en clair.**[2]**


Figure 1.1 : Schéma de fonctionnement de la cryptographie symétrique

## 4.2 Cryptographie Asymétrique

La cryptographie asymétrique, également appelée cryptographie à clé publique, fonctionne de manière radicalement différente de la cryptographie symétrique. Alors que la cryptographie symétrique peut être comparée à un coffre-fort auquel seules les personnes possédant la clé peuvent accéder, la cryptographie asymétrique pourrait être comparée à une boîte aux lettres où l'on peut déposer des informations, et seule la personne possédant la clé peut accéder au contenu de la boîte. La boîte aux lettres serait la clé publique (donc accessible à tous), tandis que la clé pour l'ouvrir serait la clé privée. En effet, dans la cryptographie asymétrique, il y a une clé publique et une clé privée.

Les nombres premiers sont essentiels pour rendre les algorithmes de cryptographie asymétrique presque indéchiffrables. Par exemple, la multiplication de 5 x 7 est assez facile, tout comme l'opération inverse de retrouver 35 à partir de facteurs premiers. Cependant, factoriser 1591 est beaucoup plus complexe, même si calculer 37 x 43 est très facile. C'est sur cette difficulté de factorisation que reposent les algorithmes asymétriques.**[2]**

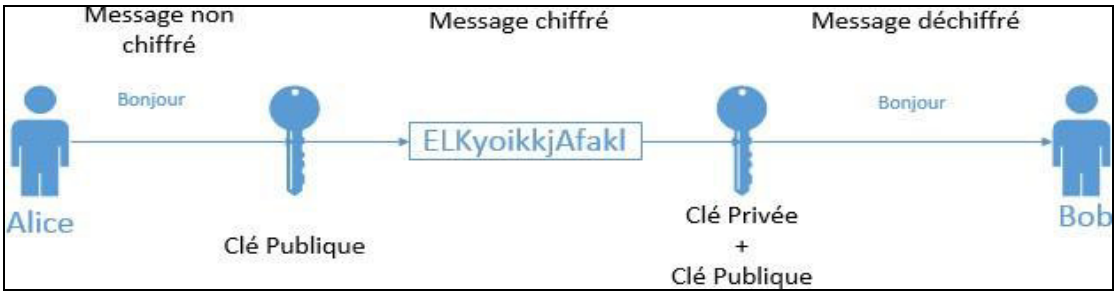



Figure 1.2 : Schéma de fonctionnement de la cryptographie asymétrique

## 4.3 Cryptage symétrique vs asymétrique

Le chiffrement symétrique et asymétrique sont deux techniques de cryptographie utilisées pour protéger les données contre l'interception et la lecture illicite. Le chiffrement symétrique utilise une seule clé pour chiffrer et déchiffrer les données, ce qui le rend rapide et efficace pour transmettre de grandes quantités de données. Cependant, la clé doit être partagée entre l'envoyeur et le destinataire avant l'échange des messages, ce qui peut poser

des problèmes de sécurité. Les algorithmes de chiffrement symétrique populaires incluent AES, DES et 3DES.

Le chiffrement asymétrique, quant à lui, utilise deux clés reliées mathématiquement entre elles : une clé publique et une clé privée. La clé publique peut être partagée librement, tandis que la clé privée reste secrète. Cette technique est plus lente que le chiffrement symétrique, mais elle offre des avantages supplémentaires tels que l'authentification et la non-répudiation en plus de la confidentialité. Les algorithmes de chiffrement asymétrique incluent RSA, Diffie-Hellman. En général, le chiffrement symétrique est utilisé pour le chiffrement de données à grande échelle, tandis que le chiffrement asymétrique est utilisé pour les communications sécurisées et les transactions financières.**[3]**

# 5 Les différentes Algorithmes

## 5.1 Algorithmes symétriques

Ce sont les algorithmes basés sur le chiffrement et déchiffrement à clé secrète.

### 5.1.1 Algorithme DES(Data encryption standard)

#### 5.1.1.1 Définition

le DES (Data Encryption Standard), adopté en 1977 par le NIST (National Institute of Standards and Technology), est un algorithme de chiffrement symétrique qui opère sur des blocs de données de 64 bits. Il utilise une clé de 64 bits, dont 8 bits sont utilisés pour la parité, laissant ainsi 56 bits effectifs pour le chiffrement. Le DES peut fonctionner selon différents modes de chiffrement, tels que :ECB,CBC,CFBet OFB **[4]**

#### 5.1.1.2 Fonctionnement

•**Chiffrement symétrique** : La même clé est utilisée pour chiffrer et déchiffrer les données.

•**Chiffrement par bloc** : Les données sont divisées en blocs de 64 bits qui sont ensuite chiffrés individuellement.

•**Algorithme de Feistel** : Le DES utilise un réseau de Feistel, qui consiste en une série de tours d'opérations appliquées à chaque bloc de données.

Le fonctionnement du DES repose sur une série de 16 rondes de transformation, chacune comprenant plusieurs opérations. Tout d'abord, le bloc de 64 bits est divisé en deux blocs de 32 bits : la droite (D) et la gauche (G).

Au cours de chaque ronde, la partie droite est traitée en utilisant une fonction F qui prend en entrée 32 bits, une sous-clé de 48 bits (K) et produit en sortie 32 bits. Cette fonction est composée d'un ou exclusif (XOR) avec la sous-clé, d'une permutation initiale des bits (IP), d'une expansion des bits de 32 à 48 bits, d'un produit de fonction (S-box) qui prend en entrée 6 bits et produit en sortie 4 bits, et enfin d'une permutation finale des bits (FP).

Après chaque application de la fonction F, la partie droite et gauche sont mélangées en utilisant une opération de XOR. À la fin des 16 rondes, les blocs droite et gauche sont combinés pour former le bloc chiffré.

Le déchiffrement du DES se fait en utilisant la même série de rondes, mais dans l'ordre inverse, avec des sous-clés différentes. Les opérations de permutation initiale et finale sont également inversées.**[4]**

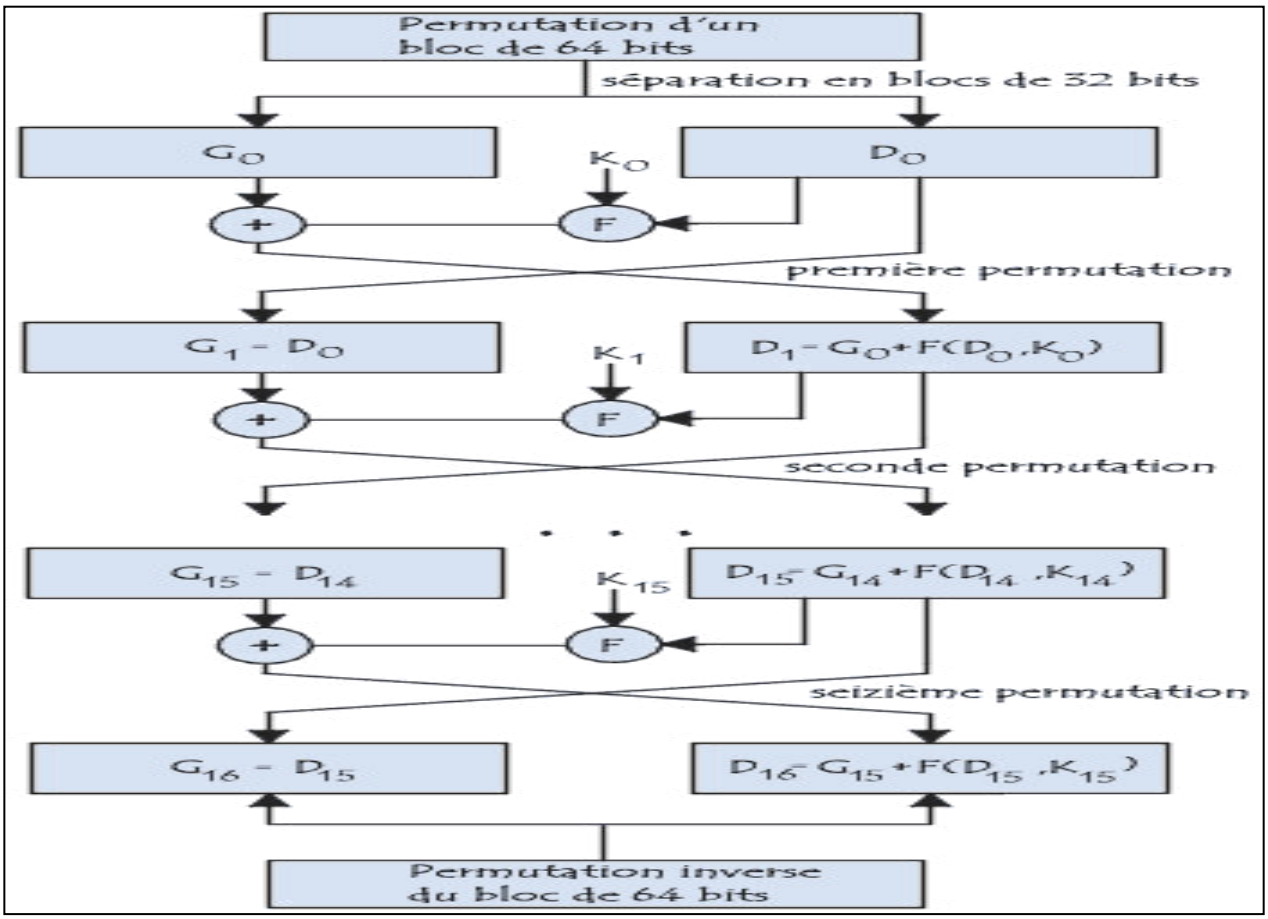



Figure 1.3 : fonctionnement du DES

### 5.1.2 Le Triple DES

#### 5.1.2.1 Définition

Le Triple DES a été conçu pour améliorer les performances du DES en résolvant la faiblesse de la longueur de sa clé. Pour ce faire, il combine plusieurs chiffrements DES pour créer un système global avec une longue clé de 112 et 168 bits, utilisant deux ou trois clés différentes. Bien que généralement utilisé avec deux clés différentes, le Triple DES peut être utilisé avec trois clés distinctes. Le mode d'utilisation standard est le mode EDE (chiffrement, déchiffrement, chiffrement), ce qui le rend compatible avec le DES lorsqu'il utilise trois fois la même clé. Une clé Triple DES est donc composée de deux DES et fait 112 bits, ce qui rend le 3DES résistant à une attaque exhaustive. Cependant, une variante à trois clés différentes pourrait être vulnérable à une attaque utilisant un des messages intermédiaires.**[5]**

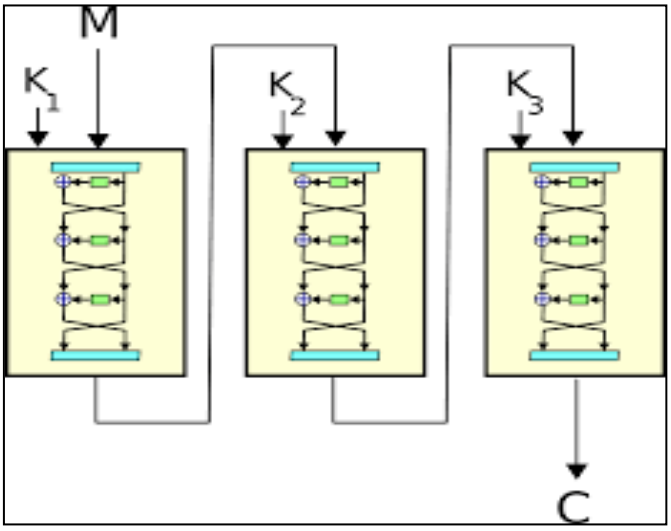



Figure 1.4: Triple DES

### 5.1.3 IDEA

#### 5.1.3.1 Définition

L'algorithme IDEA (International Data Encryption Algorithm) est un algorithme de chiffrement symétrique par blocs qui utilise une clé de 128 bits pour chiffrer ou déchiffrer des données divisées en blocs de 64 bits. Il a été développé dans les années 1990 en Suisse et est considéré comme un algorithme de chiffrement sécurisé.

#### 5.1.3.2 Fonctionnement

Le fonctionnement de l'IDEA repose sur une série de huit rondes de transformation, suivies d'une dernière ronde de mélange des données. Chaque ronde utilise une clé de 128 bits et effectue une série d'opérations telles que l'addition modulo 2, la multiplication modulo 216, et le ou exclusif bit à bit.

Au début de chaque ronde, les données sont divisées en deux blocs de 32 bits : le bloc de gauche (X) et le bloc de droite (Y). Dans chaque ronde, les blocs X et Y sont mélangés en utilisant une opération de multiplication modulo 216 ou de ou exclusif bit à bit, suivie d'une opération d'addition modulo 2 avec une sous-clé de 16 bits. Les résultats sont ensuite mélangés à nouveau en utilisant une opération de ou exclusif bit à bit ou de multiplication modulo 216, suivie d'une opération d'addition modulo 2 avec une autre sous-clé de 16 bits.

Après les huit rondes, les données sont mélangées une dernière fois en utilisant une opération de multiplication modulo 216 ou de ou exclusif bit à bit, suivie d'une opération d'addition modulo 2 avec la dernière sous-clé de 16 bits.**[5]**

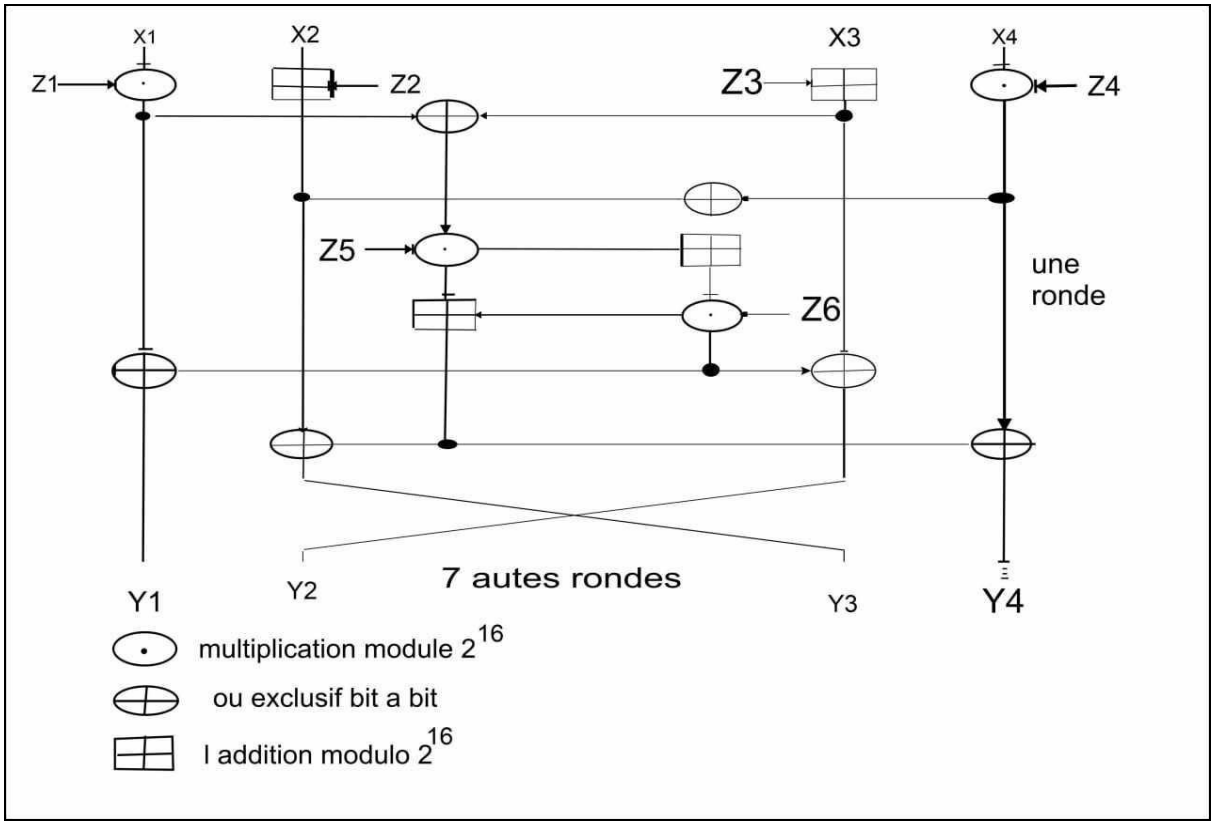



Figure 1.5 : Fonctionnement d'IDEA

### 5.1.4 BLOWFISH

#### 5.1.4.1 Définition

Blowfish est un algorithme de chiffrement symétrique par blocs conçu par Bruce Schneier en 1993. Il utilise une clé de longueur variable allant de 32 bits à 448 bits et un bloc de données de 64 bits. Blowfish est considéré

Comme un algorithme de chiffrement sécurisé et a été largement utilisé dans les applications de chiffrement.**[5]**

#### 5.1.4.2 Fonctionnement

Le processus de chiffrement de Blowfish suit une structure classique de schémas de Feistel, où une entrée de 64 bits est divisée en quatre quartiers de 16 bits chacun. Ces quartiers servent d'entrée aux S-boxes, qui produisent des sorties transformées. Ces sorties sont combinées avec les trois autres quartiers via des fonctions logiques spécifiques avant d'être regroupées pour former un nouveau quartier de 32 bits. Cette

étape est répétée jusqu'aux 16 tours du schéma de Feistel. Le résultat final est obtenu après avoir concaténé les quatre derniers quartiers.

Pour le déchiffrement, on inverse l'ordre des étapes de chiffrement et applique un XOR entre les deux premières entrées du tableau P sur le bloc chiffré. Les entrées du tableau P sont ensuite utilisées dans l'ordre inverse pour retrouver le plaintext initial.**[6]**

### 5.1.5 RC4

#### 5.1.5.1 Définition

L'algorithme RC4 est un algorithme de chiffrement symétrique par flot, qui a été développé par Ron Rivest en 1987. Il est utilisé dans de nombreux protocoles de sécurité, tels que SSL et WEP. **[5]**

#### 5.1.5.2 Fonctionnement

Le fonctionnement de l'algorithme RC4 est relativement simple. Il utilise une clé de chiffrement de longueur variable (entre 40 et 2048 bits) pour générer une séquence de bits pseudo-aléatoires, appelée flux. Ce flux est ensuite combiné avec les données à chiffrer en utilisant une opération XOR. Le flux est généré en utilisant une série de permutations et de substitutions sur un tableau d'octets initialisé avec la clé de chiffrement.

L'algorithme RC4 est considéré comme relativement rapide et efficace, mais il a été compromis par des attaques de cryptanalyse. En particulier, il est vulnérable aux attaques par injection de clé, qui permettent à un attaquant de récupérer la clé de chiffrement à partir de données chiffrées. En conséquence, il est généralement recommandé d'utiliser des algorithmes de chiffrement plus forts et plus récents, tels que AES.**[7]**

## 5.2 Algorithme Asymétriques

### 5.2.1 RSA

#### 5.2.1.1 Définition

L'algorithme RSA est un algorithme de chiffrement asymétrique, inventé par Ron Rivest, Adi Shamir et Leonard Adleman en 1977. Il utilise une paire de clés, une clé

publique et une clé privée, pour chiffrer et déchiffrer les données. La clé publique peut être partagée librement, tandis que la clé privée doit être gardée secrète

#### 5.2.1.2 Fonctionnement

Le fonctionnement de l'algorithme RSA est basé sur la difficulté de factoriser de grands nombres premiers. Pour générer une paire de clés RSA, on choisit deux nombres premiers distincts p et q, et on calcule leur produit n = p * q. On calcule ensuite la fonction d'Euler de n, qui est égale à (p-1) * (q-1). On choisit ensuite un entier e tel que 1 < e < $\varphi$(n) et que e soit premier avec $\varphi$(n). La clé publique est alors constituée de la paire (n, e).

Pour chiffrer un message, on convertit d'abord le message en un entier m tel que 0 ≤ m < n. On calcule ensuite le message chiffré c = m^e mod n. Pour déchiffrer le message, on utilise la clé privée, qui est constituée de la paire (n, d), où d est l'inverse modulaire de e modulo $\varphi$(n). On calcule alors le message déchiffré m = c^d mod n.

L'algorithme RSA est largement utilisé pour la sécurité des communications sur Internet, notamment pour le chiffrement des données de cartes de crédit et des mots de passe. Il est également utilisé pour la signature numérique, qui permet de garantir l'authenticité et l'intégrité des données. Cependant, RSA est vulnérable à certaines attaques, notamment les attaques par force brute et les attaques par factoring. Pour cette raison, des clés de plus en plus longues sont utilisées pour renforcer la sécurité.**[5]**

### 5.2.2 Protocole d'échange de clé publique DIFFI-HELLMAN

#### 5.2.2.1 Définition

Le protocole d'échange de clé publique Diffie-Hellman est un algorithme de chiffrement asymétrique qui permet à deux parties de s'échanger une clé secrète sur un canal de communication non sécurisé. Le protocole utilise des opérations mathématiques simples pour générer une clé secrète partagée entre les deux parties, sans que la clé ne soit jamais transmise sur le canal de communication.

#### 5.2.2.2 Fonctionnement

Le fonctionnement de l'algorithme Diffie-Hellman est le suivant :

1. Les deux parties conviennent d'un nombre premier p et d'un générateur g.
2. Chaque partie choisit un nombre secret, a pour la première partie et b pour la seconde partie.
3. Chaque partie calcule g^a mod p ou g^b mod p, respectivement.
4. Les deux parties échangent les résultats de leur calcul.
5. Chaque partie calcule la clé secrète en utilisant le résultat de l'autre partie et son propre nombre secret. La clé secrète est calculée comme suit :

   K = (g^ab) mod p.

La clé secrète ainsi générée peut être utilisée pour chiffrer et déchiffrer les données échangées entre les deux parties. Le protocole Diffie-Hellman est considéré comme sûr, car il est basé sur la difficulté de calculer le logarithme discret d'un nombre premier. Cependant, il est important de noter que le protocole ne fournit pas d'authentification, ce qui signifie qu'un attaquant peut potentiellement se faire passer pour l'une des parties et intercepter les données échangées .**[8]**

## 6 Conclusion

La sécurité absolue n'existe pas, seule une sécurité relative peut être atteinte. Dans ce chapitre, nous avons traités et étudiés les différents algorithmes de chiffrement symétrique et asymétrique. Le choix de l'algorithme est déterminant. Dans le chapitre suivant, nous présentons les notions de base sur les données visuelles vidéo.

# *CHAPITER2 : les concepts et les notions de base des vidéos*

# 1 Introduction

Dans ce chapitre, nous allons aborder l'importance des vidéos numériques et la valeur des informations qu'elles contiennent. Nous commencerons par présenter les différents types de vidéos et leurs caractéristiques, telles que l'espace de couleur et le format.

Ensuite, nous expliquerons les étapes de compression de l'information vidéo et citerons certaines normes de compression vidéo normalisées par les organismes de normalisation IEC/ISO et ITU-T. Enfin, nous conclurons en résumant les points clés abordés dans ce chapitre.

# 2 Type des vidéos

Il existe deux types de vidéo : vidéo analogique et vidéo numérique. **[9]**

## 2.1 Type analogique

La vidéo analogique, représentant l'information comme un flux continu de données analogiques, destiné à être affichées sur un écran de télévision, basé sur le principe du balayage qui propose deux types : balayage entrelacé et balayage progressif (voir la figure 2.1).

Il existe plusieurs normes pour la vidéo analogique. Les trois principales sont : PAL, NTSC, SECAM. **[9]**

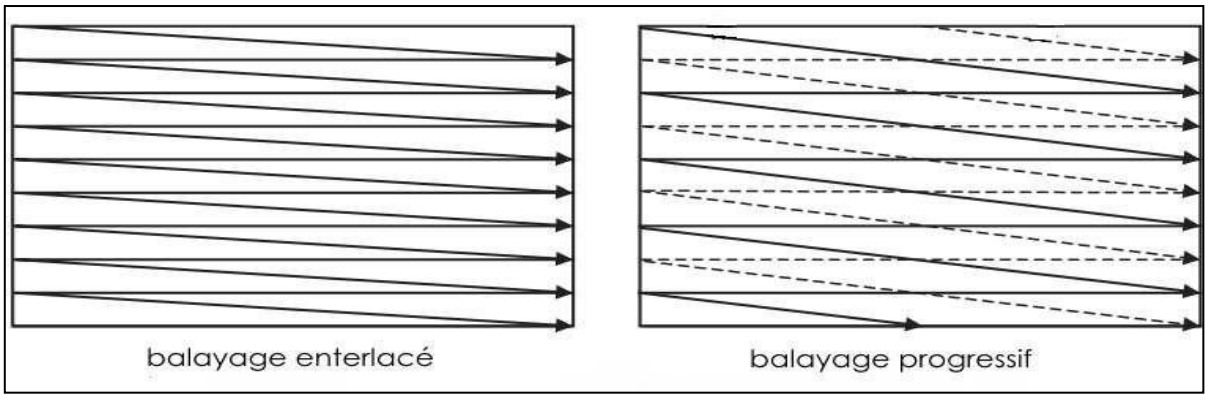



Figure 2:1 : Les types de balayage [10]

## 2.2 Vidéo numérique

Une séquence vidéo numérique est une série d'images fixes animées qui représentent une information ou une scène de manière temporelle. Chaque séquence vidéo possède des caractéristiques spatiales telles que la résolution, l'espace colorimétrique utilisé, le format de sous-échantillonnage des couleurs (pour le système YCrCb), et la résolution temporelle exprimée en nombre d'images par seconde (fps - Frames per second). **[10]**

# 3 Couleur de l'espace

Une image numérique est une représentation sous forme de tableau en deux dimensions composées d'échantillons appelés pixels. La précision de l'image détermine le nombre de niveaux d'intensité pouvant être représentés, mesurée en nombre de bits par échantillon. En fonction de cette précision, les images peuvent être catégorisées comme suit :

Images binaires : 1 bit par échantillon

Infographie : 4 bits par échantillon

Images en niveaux de gris : 8 bits par échantillon

Images en couleur : 16, 24 bits ou plus par échantillon.

Selon la théorie trichrome, la perception des couleurs est générée en stimulant de manière sélective trois types de récepteurs dans l'œil.**[11]**

## 3.1 Représentation RVB (Rouge-Vert -Bleu)

Dans un système de représentation des couleurs RVB, tel qu'illustré dans la figure 2.2, une couleur est obtenue en combinant trois couleurs primaires : le rouge, le vert et le bleu. La ligne diagonale où R = G = B représente les valeurs de gris variant du noir au blanc. **[10]**

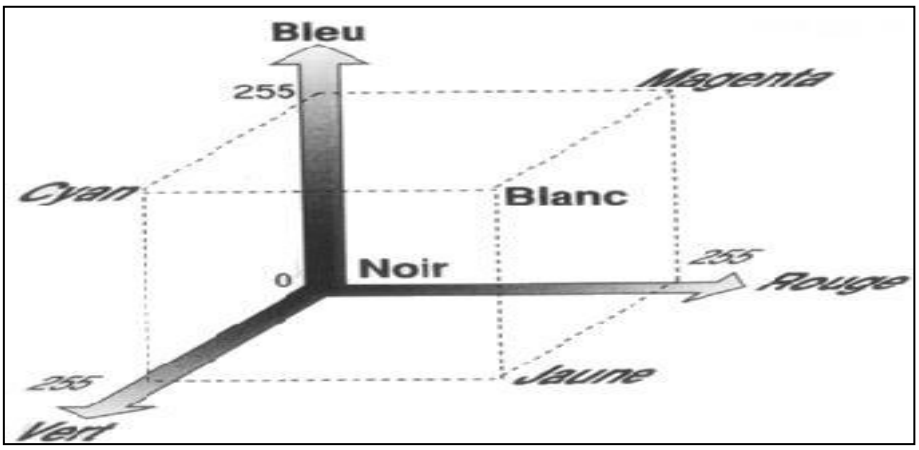



Figure 2.2 : Représentation des couleurs RVB

Une autre méthode de représentation des images en couleur implique la séparation des composants de luminance et de chrominance. La composante de luminance, notée Y, offre une version en niveaux de gris de l'image selon l'équation (1.1), tandis que les deux composantes de chrominance, U et V, fournissent des informations supplémentaires permettant de convertir l'image en niveaux de gris en une image couleur grâce aux équations (1.2) et (1.3). La représentation YUV est souvent préférée pour la compression d'images et de vidéos en raison de sa nature plus adaptée à ces processus. La transformation précise de la représentation RVB en YUV est définie par la norme CCIR 601. **[10]**

$$Y = 299R + 0.587G + 0.114B \quad (1.1)$$

$$U = 0.564(B - Y) \quad (1.2)$$

$$V = 0.713(B - Y) \quad (1.3)$$

## 3.2 Représentation YCbCr

Le format YCbCr est largement employé pour la compression d'images. Dans ce format, la composante Y est similaire à celle du système YUV. Cependant, les composantes U et V, qui décrivent la chrominance, subissent une mise à l'échelle et une conversion en valeurs zéro-centrées

pour donner respectivement Cb et Cr, comme indiqué dans les équations (1.4) et (1.5).

**Cb = U/2 + 0.5** **(1.4)**

**Cr = V/1.6 +0.5** **(1.5)**

De cette manière, les composantes de chrominance Cb et Cr sont toujours comprises dans la plage [0,1]. **[11]**

## 4 Formats d'échantillonnage

La Figure 2.3 présente trois schémas d'échantillonnage pour Y, Cb et Cr :
L'échantillonnage 4:4:4 indique que les trois composantes (Y, Cb et Cr) ont la même résolution, avec un échantillon de chaque composante à chaque position de pixel.
Pour l'échantillonnage 4:2:2 (parfois appelé YUY2), les composantes de chrominance ont la même résolution verticale que la luminance mais la moitié de la résolution horizontale. Ce schéma est utilisé pour une reproduction des couleurs de haute qualité.
Dans le format d'échantillonnage 4:2:0 (appelé "YV12"), Cb et Cr ont chacun la moitié de la résolution verticale de Y. Les chiffres 4:2:0 ne sont pas interprétés de manière logique, mais ont été historiquement choisis comme code pour identifier ce schéma d'échantillonnage spécifique et le distinguer des schémas 4:4:4 et 4:2:2. [12]

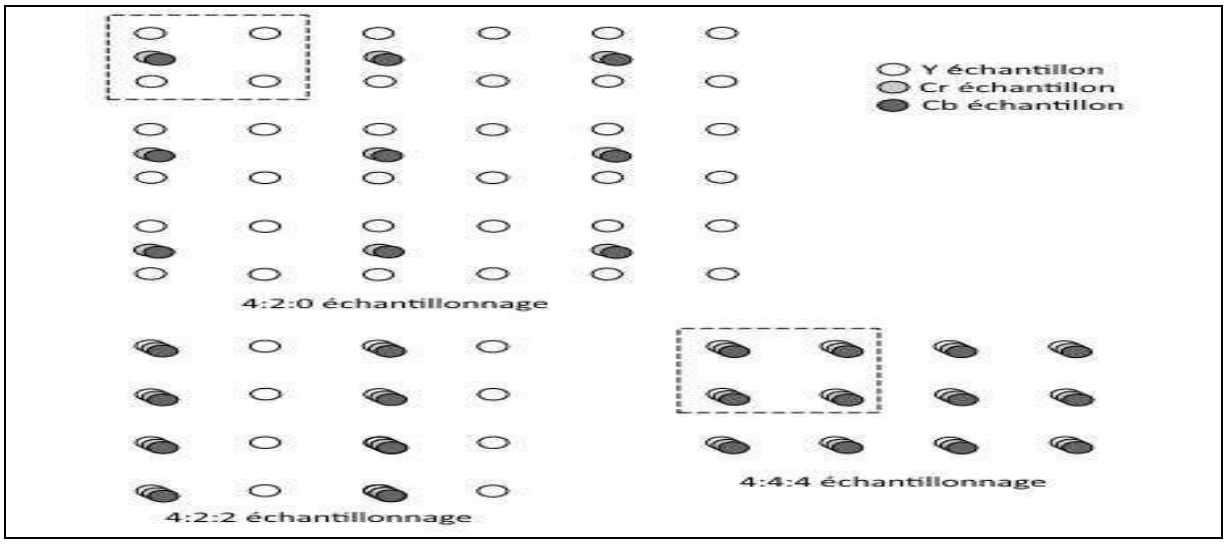



Figure 2.3 : Profils d'échantillonnage 4 : 2 : 0, 4 : 2 : 2 et 4 : 4 : 4. [12]

## 5 Formats vidéo numériques

Le groupe de spécialistes de l'UIT (SGXV) a proposé trois formats recommandés : le format d'entrée standard (SIF), le format d'échange commun (CIF) et une version à débit réduit du CIF appelée quart CIF (QCIF) . Ces formats, présentés dans le Tableau 2.1, offrent une gamme complète de formats vidéo numériques couramment utilisés dans les applications mobiles. **[10]**

| Description | SIF | CIF | QCIF |
|---|---|---|---|
| **Résolution horizontale (Y) pixels** | 352 | 360(352) | 180(176 ) |
| **Résolution verticale (Y) pixels** | 240/288 | 288 | 144 |
| **Résolution horizontale (Cr, cb) pixels** | 176 | 180(176) | 90(88) |
| **Résolution verticale (Cr, cb) pixels** | 120/144 | 144 | 72 |
| **Bits/pixel (bpp)** | 8 | 8 | 8 |
| **Mode d'affichage** | Progressif | Progressif | Progressif |
| **Trames par seconde (fps)** | 30 | 30, 15, 10, 7.5 | 30, 15, 10,7.5 |

*Tableau 2-1 : Les différents formats de la vidéo numérique*

La Figure 2.4 présente différents formats pour la transmission d'images numériques provenant de la télévision à tube, tels que la définition standard (SD) avec une résolution de 720x576 pixels, utilisée par les DVD. Le format 720p, considéré comme une Haute Définition (HD) intermédiaire, affiche une résolution de 1280x720 pixels et est principalement utilisé par les services de VOD (Vidéo à la demande). Le Full HD, une autre définition de la HD, offre une résolution de 1920x1080 pixels et est utilisé par des dispositifs tels que les téléviseurs, les Blu-Ray, le Quad HD, l'Ultra HD ou 4K (avec une résolution de 3840x2160 pixels). Le format cinéma propose une résolution de 4096x2160 pixels, tandis que le format 8K offre une définition avec 16 fois plus de pixels que le Full HD et 4 fois plus que la 4K, soit une résolution de 7680x4320 pixels. **[13]**

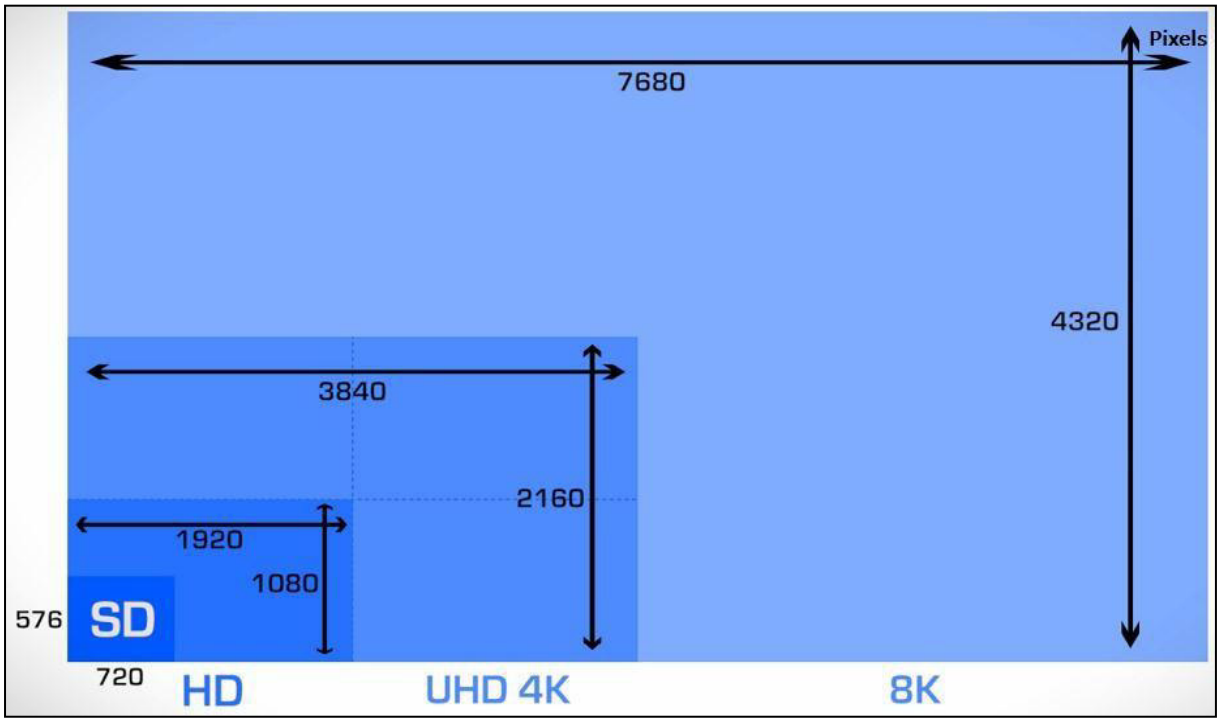



Figure 2.4 : L'évolution des formats de la vidéo numérique: de SD au 8K. [13]

## 6 Conteneurs de références

Les vidéos numériques sont contenues dans des fichiers qui sont appelés des conteneurs. Reconnaissable par leur extension de fichier (voir la Table 2.2).

| Conteneur | Extension |
|---|---|
| Audio Video Interleave | .avi |
| MPEG 1/2/4 | .mpg , .mpeg |
| Matroska | .mkv , .mka , .mks |
| QuickTime | .mov |
| 3gp | .3gp , .3g2 |
| Stream | .ts |
| Flash Video | .flv |

Tableau 2-2 : Conteneurs de vidéo

## 7 Compression vidéo numérique

La compression vidéo consiste à transformer un signal vidéo en un format qui nécessite moins d'espace de stockage ou de bande passante pour la transmission. Étant donné les exigences élevées en termes de transmission et de stockage vidéo (allant jusqu'à 270 Mbits/s pour la définition standard et 1,5 Gbit/s pour la haute définition), la compression vidéo est une technologie cruciale pour diverses applications telles que la télévision numérique (terrestre, câble, satellite), le stockage/reproduction optique, la télévision mobile, la vidéoconférence et la diffusion en continu sur Internet. **[10]**

### 7.1 Types de compression

Il existe deux types de compression la compression avec perte et la compression sans perte :

#### 7.1.1 Compression sans perte

La compression sans perte implique la compression des données sans altération de leur contenu. Les techniques de compression sans perte garantissent la préservation intégrale des informations. Lorsque des données sont compressées sans perte, il est possible de récupérer exactement les données d'origine à partir des données compressées. Ce type de compression est couramment utilisé pour

des applications où toute différence entre les données originales et les données reconstruites est inacceptable. Parmi les méthodes de compression sans perte, on retrouve le codage en longueur, le codage Huffman et la méthode Lempel-Ziv-Welch (LZW).

### 7.1.2 Compression avec perte

Le type de compression le plus répandu pour la vidéo est celui qui offre des taux de compression considérablement plus élevés. Cependant, ce choix implique un compromis : à mesure que le taux de compression augmente, la qualité de la vidéo compressée diminue. Les codecs avec perte, bien qu'inadaptés aux données informatiques, sont largement utilisés dans les normes MPEG en raison de leur capacité à offrir des taux de compression plus élevés que les codecs sans perte.

#### 7.1.2.1 Redondance vidéo

La compression est obtenue en supprimant les informations redondantes de la vidéo. Il existe quatre principaux types de redondances qui sont généralement explorés par les algorithmes de compression : **[14]**

- **Redondance perceptuelle** : l'information de la vidéo qui ne peut pas être facilement perçues par l'observateur humain et qui, par conséquent, peuvent être supprimées sans altération significative de la qualité de vidéo.
- **Redondance temporelle** : Les pixels des images vidéo successives montrent une forte similitude. Par conséquent, bien que le mouvement puisse déplacer les blocs de pixels, il ne modifie pas leurs valeurs ni leur corrélation (comme illustré dans la figure 2.5 (b)). Ce phénomène est également connu sous le nom de redondance inter-codage.
- **Redondance spatiale** : il existe une corrélation significative entre les pixels situés autour du même voisinage dans une image (comme figure 1.5 (a)). On l'appelle aussi redondance intra-codage.
- **Redondance statistique** : Ce type de redondance est lié à la relation statistique dans les données vidéo (bits et octets).

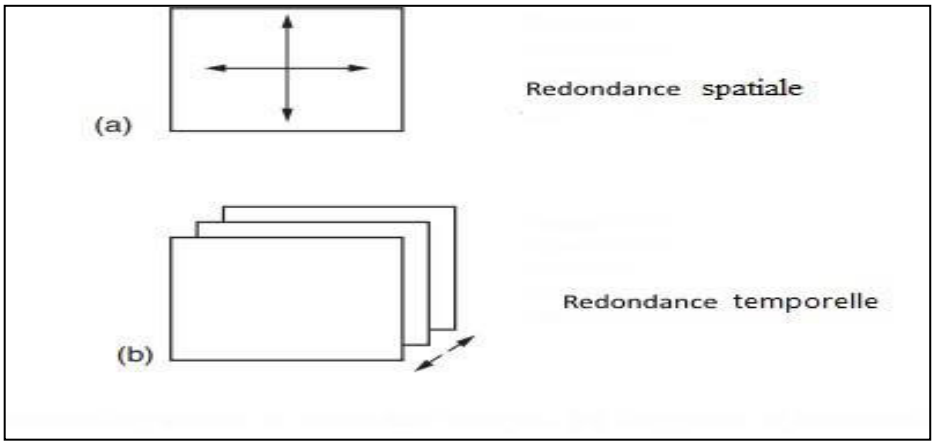



Figure 2.5 : Redondances dans la vidéo

**a**) Redondance spatiale

**b**) Redondance temporelle

### 7.1.2.2 Vidéo MPEG

The Motion Pictures Expert Group (MPEG), créé par l'ISO, élabore des normes pour diverses applications multimédias intégrant vidéo et son. Les vidéos MPEG se composent d'une séquence de groupes d'images (GOP), se démarquant par leur taille réduite par rapport à d'autres formats grâce à des techniques de compression avancées. La Discrete Cosine Transform (DCT) est utilisée pour la compression, exploitant la similarité entre les images d'une séquence pour un taux de compression élevé. La compression MPEG vise à éliminer la redondance spatiale et temporelle, avec trois types de trames : les trames I (Intra Frames ou key frames), les trames P (Inter Frames ou Delta Frames), et les trames B (Bidirectionnel frames).

- **I frames** : Les trames I sont des images intra-codées indépendantes, divisées en macroblocs, comme la montre dans

**La figure 2.6**. Chaque macrobloc est compressé à l'aide de DCT suivi d'une quantification et codage à longueur variable. La sortie du flux codé est appelée flux MPEG.

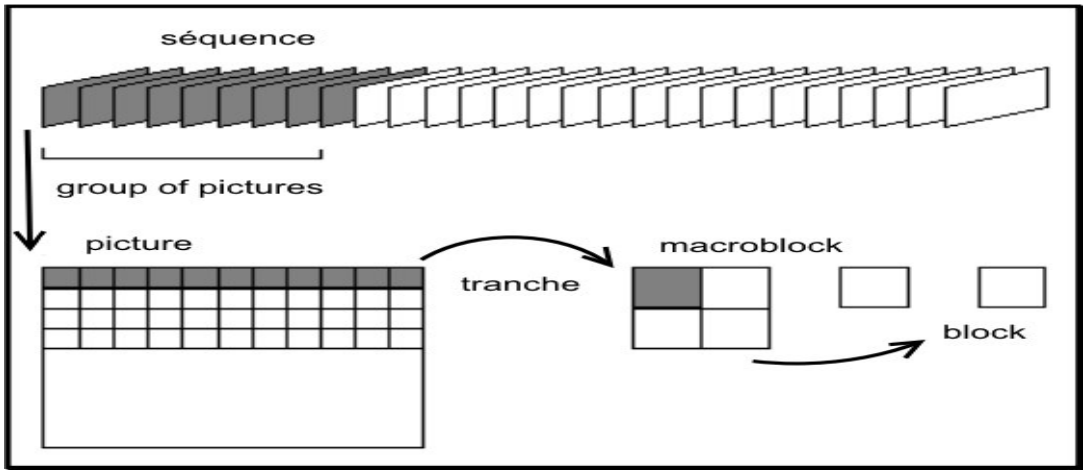



Figure 2.6: Une vidéo MPEG est une séquence de groupe d'images (GOP)**[15]**

Chaque image est divisée en tranches et chaque tranche est ensuite divisée en macroblocs.

- **Trames P et B** : Les trames P sont des trames inter-codées prédictivement à partir des trames I ou P précédentes, tandis que les trames B sont codées de manière bidirectionnelle en utilisant les trames I et/ou P précédentes et suivantes. Dans la Figure **2.7**, on observe que la trame de prédiction en arrière suit la trame prédite, ce qui implique de retarder le décodage des images B jusqu'à l'apparition de la prochaine image P ou B dans le flux. L'ordre d'affichage ne correspond pas à l'ordre de codage ; les trames sont disposées dans le flux MPEG de manière à ce que les trames de référence précèdent celles qui les référencent. Ainsi, la séquence de trames est transmise dans l'ordre suivant : I P B B B P B B B. Le décodeur doit alors réorganiser les images reconstruites, les trames P et B étant définies par leur déplacement par rapport aux trames I correspondantes et aux résidus d'erreur.

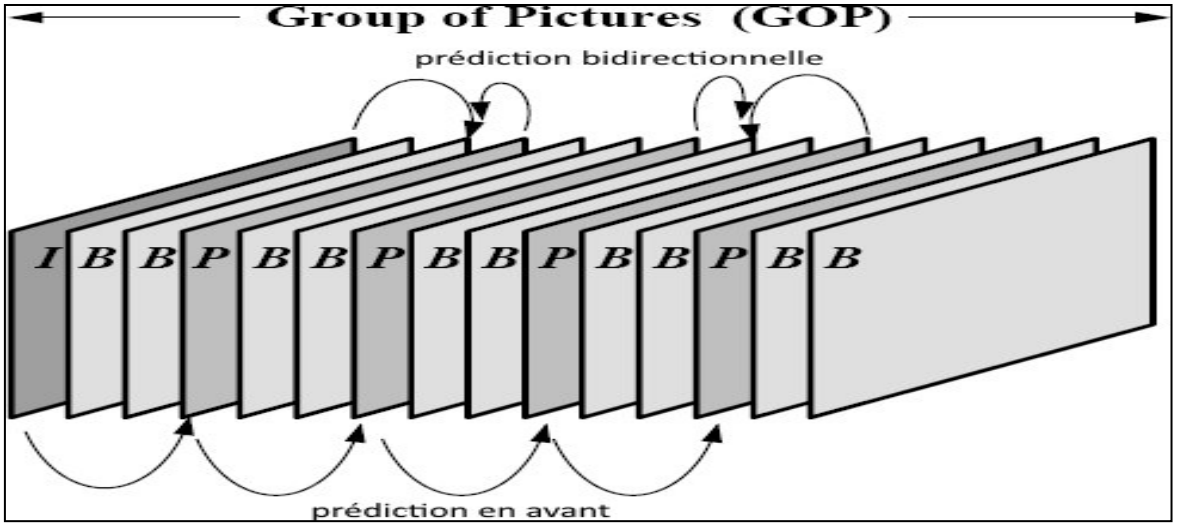



Figure 2.7 : Exemple de groupe d'images avec l'ordre d'affichage. [15]

### 7.1.2.3 Codage MPEG

Le processus de codage MPEG implique les quatre étapes présentées dans la **figure 2.8**. Préalablement, la vidéo est convertie de l'espace colorimétrique RVB en YUV et subit un sous-échantillonnage dans le domaine UV. Ce sous-échantillonnage en YUV a pour objectif de réduire la résolution des composantes de couleur.

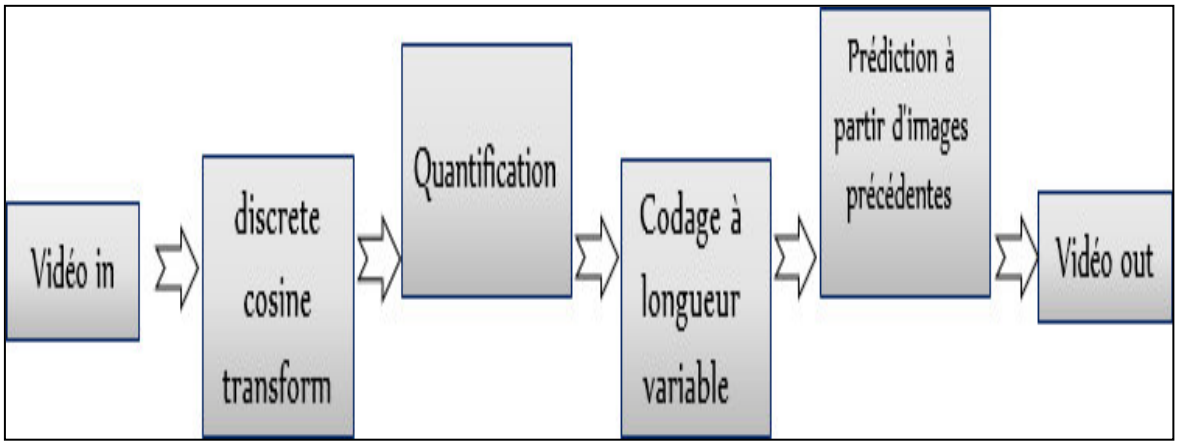



Figure 2.8: Étapes de codage typiques utilisées dans la compression MPEG

#### 7.1.2.3.1 DCT

Une Discrete Cosine Transform exprime une séquence finement composée des points de données en termes d'une somme des fonctions cosinus oscillant à différentes fréquences. Les DCT sont importants pour des nombreuses

applications en sciences et en ingénierie, de la compression avec perte d'audio et d'images (où de petites composants haute fréquence peut être éliminés). Il est choisi de facto dans la plupart des normes MPEG transformer car il a été prouvé être une approximation de la transformée optimale K-L au premier ordre de Markov modèle source . La transformation 2D a une dimension de 8 * 8, et dans le contexte de la compression spatiale en MPEG, la DCT effectue la conversion des données du domaine spatial vers le domaine fréquentiel.

#### 7.1.2.3.2 Quantification

L'étape principale de perte dans le processus de compression MPEG consiste à arrondir la plupart des composantes de fréquence élevée à zéro, tandis que les autres deviennent de petits nombres positifs ou négatifs. Cette méthode permet de réduire considérablement la quantité de bits nécessaires pour les stocker.

#### 7.1.2.3.3 Codage à longueur variable

Une fois les coefficients DCT quantifiés, ils subissent un processus de codage entropique, une méthode de compression de données sans perte. Les coefficients sont organisés en zigzag avant d'être encodés à l'aide de l'algorithme RLE (Run-Length Encoding), qui regroupe les fréquences en insérant des zéros de longueur de codage, suivi de l'application du codage de Huffman sur les données restantes.

#### 7.1.2.3.4 Prédiction à partir d'images précédentes

Cela correspond à une estimation ou compensation de mouvement, visant à réduire la redondance temporelle. Cette technique de compression est appliquée aux images P et B. Elle consiste à calculer la différence entre le macrobloc actuel et le macrobloc de prédiction le plus proche dans le cadre de référence, puis à transmettre ces données résiduelles (erreur de prédiction). Les données résiduelles subissent une nouvelle décorrélation spatiale via une transformation 2D après la décorrélations temporelle induite par le mouvement. Les vecteurs de mouvement décrivant le meilleur appariement du macrobloc sont des données

sémantiques, non des données pixel, indiquant le décalage entre le macrobloc actuel et son meilleur appariement dans le flux binaire.

## 8 Normes de codage vidéo

Les normes de codage vidéo ont progressé sous les appellations H.26x et MPEG-x. Les codecs H.26x sont préconisés par le secteur de normalisation des télécommunications de l'ITU-T, orientés vers des applications telles que la vidéoconférence et la téléphonie vidéo. D'autre part, les produits MPEG-x sont le fruit de l'ISO/IEC, visant principalement à répondre aux besoins de stockage vidéo (CD-ROM, DVD), à la diffusion télévisuelle et à la diffusion en continu sur Internet. Bien que les deux comités de normalisation aient généralement travaillé de manière indépendante sur des normes distinctes, des exceptions ont conduit à des normes conjointes telles que H.262/MPEG-2 et H.264/MPEG-4 Partie 10 (v10). L'évolution des normes de codage vidéo des deux organisations depuis 1984 est résumée dans la figure 2.9, mettant en lumière leurs collaborations. De plus, la figure 11 illustre l'évolution du codage d'images fixes résultant des efforts conjoints de l'ITU-T et de l'ISO/IEC.

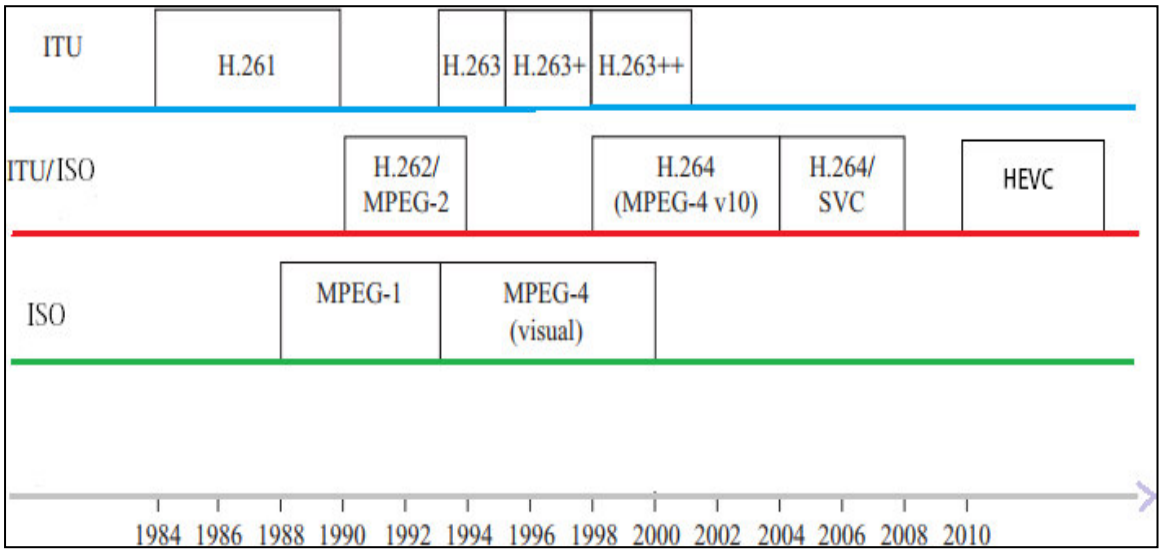



Figure 2.9 : Evolution des normes de codage vidéo

de l'ITU-T et de l'ISO / IEC comités.

- **La norme H.264 / MPGE 4**

La norme H.264/MPEG-4 AVC est le fruit d'une collaboration entre le comité de normalisation VCEG de l'ITU-T et les comités de normalisation MPEG ISO/IEC. Cette norme offre une compression nettement supérieure à ses prédécesseurs, permettant un encodage efficace des vidéos entrelacées et non entrelacées. Même à des débits binaires élevés, elle garantit une qualité visuelle plus satisfaisante que les standards précédents. De plus, elle propose des options de codage flexibles et une organisation des données codées pour renforcer la résilience aux erreurs ou aux pertes. AVC fait référence à Advanced Video Coding, et la Table 2.3 détaille les caractéristiques de cette norme.

| Catégorie | Description |
|---|---|
| Type de frame | Pour la norme h.264/avc on retrouve les mêmes types d'images que dans les normes précédentes (I, P ou B) |
| Division de macrobloc | La norme H.264/AVC prend en charge cinq types de tranches tranches de type I (Intra), les tranches de type P (prédictif), les tranches de type B (Bi-prédiction)et enfin les tranches de types SP (commutation P)et SI (commutation I). |
| Prédiction | La norme H264/AVC propose sept modes de prédictions. La prédiction d9un macrobloc peut être effectuée en le considérant dans son intégralité ou en le divisant en sous-blocs. Ainsi, pour le signal de luminance nous distinguons les partitions 16_16, 16_8, 8_16, 8_8, 8 _ 4, 4 _ 8 et 4 _ 4. Les deux composantes de chrominance peuvent aussi être partitionnées en sous-blocs allant de |

| | |
|---|---|
| | 8 _ 8 échantillons jusqu'à 2 _ 2 échantillons |
| **Quantification** | La norme H264/AVC applique sur les coefficients de la transformée une quantification avec 52 niveaux ( quantification. Pour chaque niveau un paramètre de quantification QP (quantization parametre) est associé. Le choix du pas est important pour un bon réglage du débit. Par la suite, les coefficients de chaque bloc de 4*4 sont ordonnés en appliquant le balayage en zigzag. |
| **Le codage entropique** | Le codage entropique peut être réalisé de deux manières différentes : -CAVLC (Context-Adaptive Variable Length Coding) : est une alternative pour le codage des tables de coefficients de transformation. -CABAC (Context- Adaptive Binary Arithmetic Coding) : est une combinaison d'une technique adaptative de codage arithmétique binaire qui permet un degré élevé d'adaptation et une réduction plus importante de redondance |
| **Filtre anti-blocs** | La H264/AVC définit un filtre de « déblocage » adaptatif en boucle qui a pour but de minimiser la visibilité des artefacts dus aux codages par blocs. Ce filtre réduit le débit binaire et offre une meilleure qualité d'image qu'avant filtrage. |

Tableau 2-3 : Les caractéristique de la norme h.264/avc.

## 9 Conclusion

Dans ce chapitre, nous avons étudiés les notions de base des données visuelles (vidéo) comme la compression et le codage des vidéos numériques. Dans la suite de notre chapitre, nous présentons le principe de fonctionnement de notre algorithme de chiffrement symétrique AES.

# *CHAPITRE 3 : Le principe de base de fonctionnement de l'algorithme AES*

# 1 INTRODUCTION

L'Institut national des normes et de la technologie (NIST) a lancé un appel à propositions pour l'Advanced Encryption Standard (AES), une norme fédérale de traitement de l'information (FIPS) utilisée pour sécuriser les données électroniques. L'AES est un algorithme de chiffrement de bloc symétrique capable de crypter et décrypter des informations, transformant les données en texte chiffré lors du cryptage et les ramenant à leur forme originale, le texte en clair, lors du décryptage. Il peut utiliser des clés cryptographiques de 128, 192 et 256 bits pour traiter des blocs de données de 128 bits.

# 2 FONCTIONNEMENT D'AES

À l'origine, le chiffrement de Rijndael (vainqueur du concours AES) incluait des options de chiffrement avec des clés de 192 et 256 bits en plus de celles de 128 bits. Les différentes tailles de clés n'altèrent pas le fonctionnement de l'algorithme, la variation réside uniquement dans le nombre d'itérations des quatre opérations de la deuxième phase. Le Tableau 2.1 spécifie le nombre d'itérations (Nr) effectuées, dépendant du nombre de colonnes (Nk) et de lignes (Nb) de la matrice contenant la clé. Par exemple, dans AES 128 bits, le nombre de tours de boucle est égal à Nr – 1. L'algorithme se déroule en plusieurs étapes, communément appelées "rounds".**[16]**

| | Nk | Nb | Nr |
|---|---|---|---|
| 128 | 4 | 4 | 10 |
| 192 | 6 | 4 | 12 |
| 256 | 8 | 4 | 14 |

Tableau 3-1 : Tableau du nombre d'itérations par rapport à la clé

Le round initial permet de réaliser une opération initiale de clé. Ensuite, quatre opérations sont répétées neuf fois

## 2.1 SubBytes

Cette procédure effectue une substitution non linéaire sur la matrice state[8]. Chaque octet subit un remplacement par un autre octet sélectionné dans une table distincte appelée S-Box. La S-Box est un tableau bidimensionnel comprenant 16 cases en X et en Y, totalisant ainsi 256 valeurs uniques. Par exemple, prenons la lettre A avec une valeur ASCII de 65, soit 0100 0001 en binaire. En divisant ces 8 bits en deux groupes de 4 bits, les valeurs obtenues sont 4 et 1. Ces deux valeurs correspondent aux indices x et y de la matrice, pointant ainsi vers la nouvelle valeur de A.**[2]**

## 2.2 ShiftRows

Durant cette phase, chaque élément du tableau altéré par l'étape précédente subit un décalage. En considérant la matrice state comme une grille de 4 cases sur 4 (chaque case contenant 8 bits, totalisant toujours 128 bits), le décalage se fait de la manière suivante : la première ligne reste inchangée, la deuxième ligne est décalée d'une position, la troisième de deux positions, et la quatrième de trois positions, tel que présenté dans la Figure 3.1 ci-dessous.**[2]**

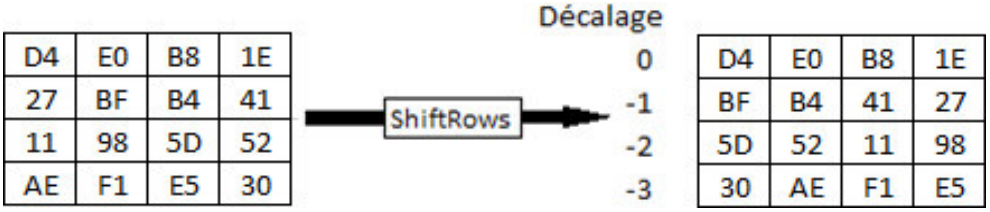



Figure3.1 : Opération ShiftRows

## 2.3 MixColumns

Cette phase implique la multiplication matricielle entre chaque colonne de la matrice state et une autre matrice. Mathématiquement, cette seconde matrice est calculée dans un corps fini à $2^8$ éléments. En prenant l'exemple de la Figure 3.2, la colonne de state utilisée représente un polynôme a(x) de degré 3, tandis que le polynôme $c(x) = 03x^3 + x^2 + x + 02$ est utilisé. Pour obtenir les nouvelles valeurs, le calcul suivant est effectué : $a(x) * c(x) \bmod (x^4 + 1)$. Il est crucial de noter que le modulo est délibérément appliqué pour garantir un résultat compris entre 0 et

255. De plus, c(x) et $(x^4 + 1)$ doivent être premiers entre eux pour éviter un résultat de modulo nul, ce qui empêcherait le déchiffrement par une opération inverse.

D'un point de vue informatique, ces calculs sont effectués sous forme de produits matriciels. En se référant à la Figure 9, le calcul pour obtenir la première valeur de la colonne serait : 2*D4 + 3*BF + 1*5D + 1*30. Ce processus est ensuite répété pour chaque ligne de la matrice pour obtenir les nouvelles valeurs.**[2]**

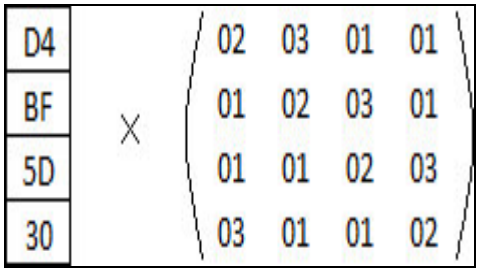



Figure 3.2 : Calcul de MixColumns

## 2.4 KeyExpansion

Avant de procéder à l'étape suivante, AddRoundKey, la clé subit une modification. La première colonne de la clé modifiée correspond à la dernière colonne, décalée de bas en haut. Ensuite, cette colonne est transformée à l'aide de la SBox, suivie d'une opération XOR entre le résultat obtenu, la première colonne de la clé initiale, et la colonne x (où x représente le numéro du tour) de la matrice Rcon (une table de constantes de tours). Les trois autres colonnes restantes sont obtenues par des opérations XOR entre la colonne de la clé initiale et la dernière colonne ajoutée à la nouvelle clé.**[2]**

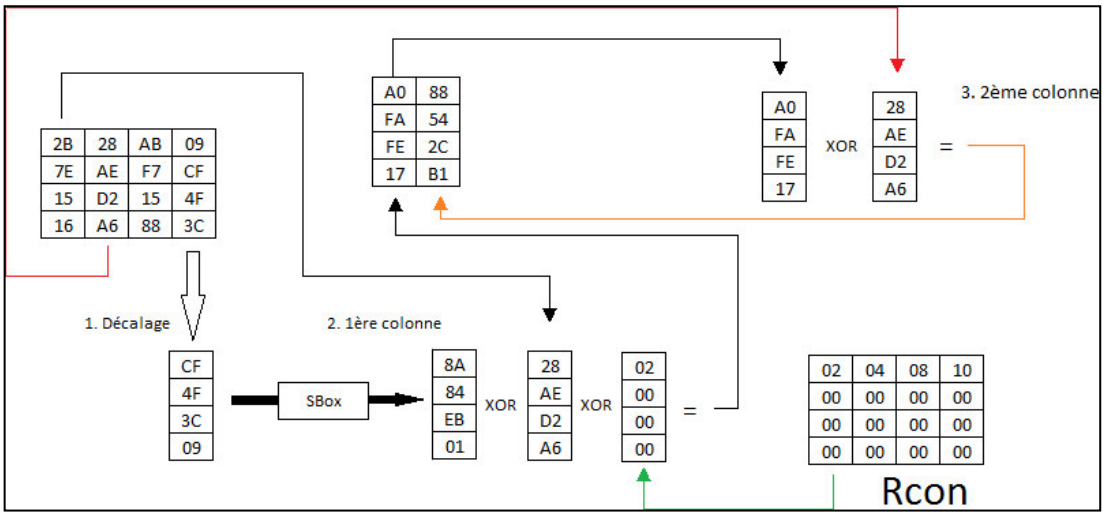



Figure 3.3: Schéma de fonctionnement de l'étape KeyExpansion

## 2.5 AddRoundKey

Cette étape va réaliser une simple opération xor entre la clé et le state.

Une fois les neuf rounds réalisés, il y a un round final. Ce round reprend les mêmes quatre étapes expliquées précédemment à l'exception de l'étape « MixColumns ».

Une fois le round final terminé, le message est chiffré.**[2]**

# 3 CHIFFREMENT ET DECHIFFREMENT

Avec l'AES Dans la suite, nous examinons l'AES-128, où les 128 bits de données sont répartis en 16 blocs de 8 bits (8 bits = 1 octet), qui sont ensuite disposés dans une matrice 4×4. Même les 128 bits de la clé sont structurés sous forme matricielle [BAB 12]. Pour des raisons de clarté, les valeurs binaires seront représentées en hexadécimal.

La première phase du chiffrement implique la combinaison de la matrice State (le bloc de texte clair) avec la clé, connue sous le nom d'AddRoundKey. À chaque tour, quatre transformations sont appliquées : SubBytes, ShiftRows, MixColumns et AddRoundKey, à l'exception du dernier tour où l'opération MixColumns n'est pas exécutée. Chaque tour utilise sa propre sous-clé générée par l'opération Key Expansion à partir de la clé principale.**[17]**

Le processus de déchiffrement est l'inverse du chiffrement, avec les transformations effectuées en sens inverse. La figure 3.4 illustre les diverses opérations réalisées à chaque tour lors du processus de chiffrement et de déchiffrement de l'AES

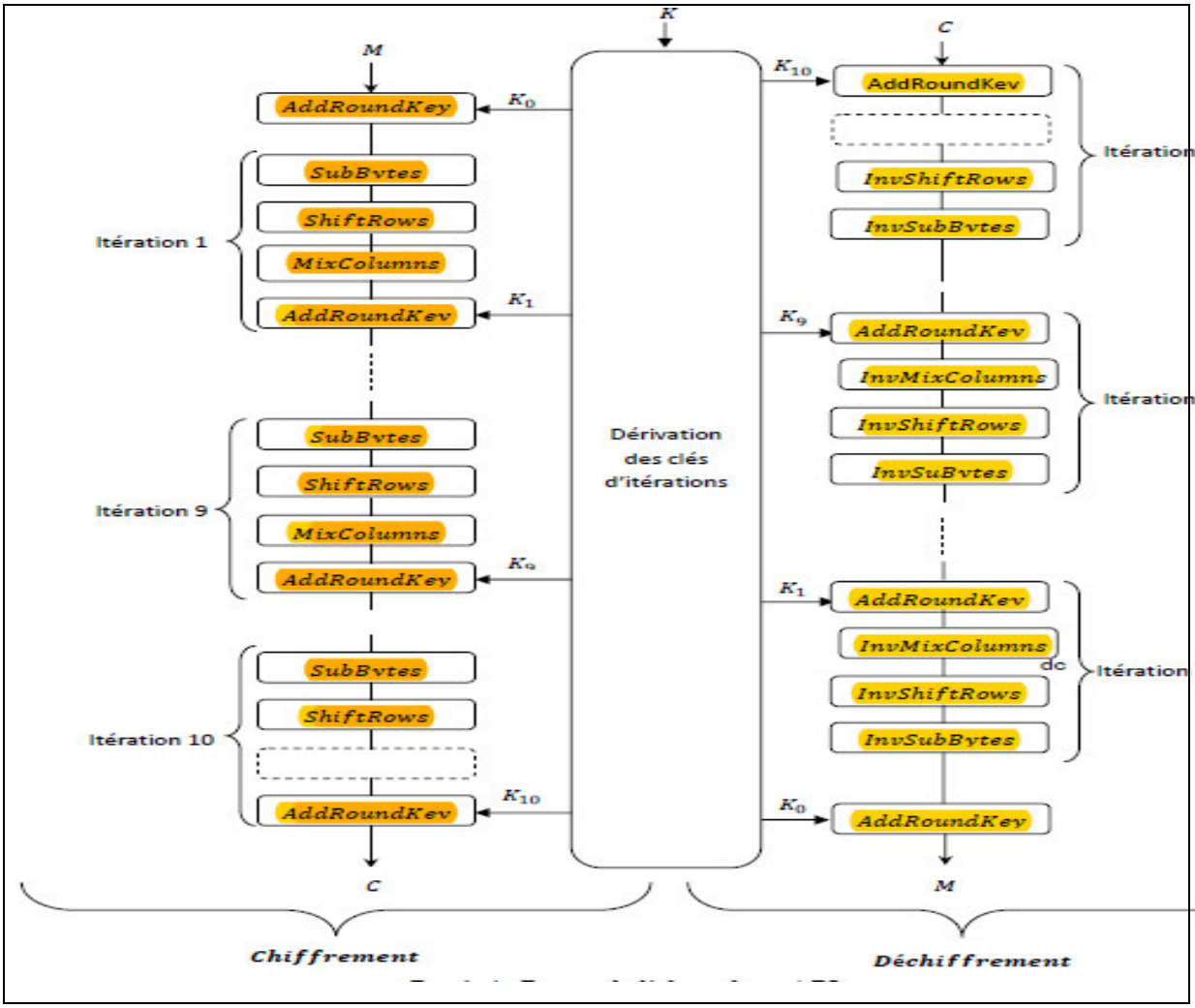



Figure 3.4: Etapes de l'algorithme AES

# 4 CHIFFRAGE

## 4.1 PROSESSUS DE CHIFFREMENT

Le schéma du processus de chiffrement de l'algorithme AES (Advanced Encryption Standard) est présenté dans la figure 3 .5. Les opérations de chiffrement reposent sur quatre fonctions prédéfinies : AddRoundKey, SubBytes, ShiftRows et MixColumns. Chacune de ces fonctions est appliquée au tableau d'états. Le processus de chiffrement comprend une étape initiale, des tours intermédiaires et un tour final. L'étape initiale implique l'application de la fonction AddRoundKey au tableau d'états. Les tours intermédiaires exécutent, dans l'ordre, les fonctions SubBytes, ShiftRows, MixColumns et AddRoundKey sur le tableau d'états. Le tour

final se distingue des tours intermédiaires par l'omission de la fonction MixColumns dans la séquence des transformations **[18][19]**.

Le nombre de tours dans l'AES dépend de la longueur de la clé. Par conséquent, avec une clé de 128 bits, le nombre de tours est de 10. Pour une clé de 192 bits, il y a 12 tours, et pour une clé de 256 bits, le nombre de tours est de 14.

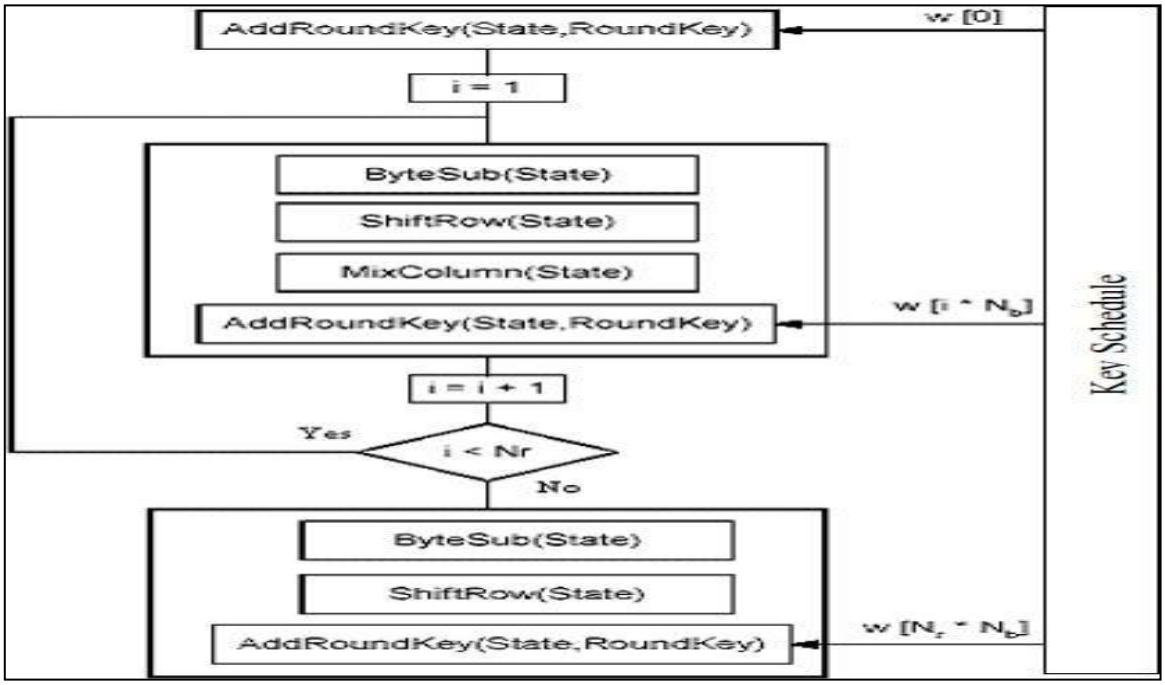



Figure 3.5: Le processus de chiffrement de l'AES

Ce diagramme est générique pour les spécifications AES. Il consiste en un nombre de transformations différentes appliquées consécutivement sur les bits du bloc de données, dans un nombre fixe d'itérations, appelées rounds. Le nombre de tours dépend de la longueur de la clé utilisée pour le processus de cryptage **[18][19]**.

## 4.2 LA TRONSFORMATION subBytes

La substitution des octets par la fonction subBytes (state) est une substitution non linéaire d'octets agissant individuellement sur chaque octet de l'état en se basant sur une table de substitution (S-box) illustrée dans la figure 3.8. Cette S-box, réversible, est créée en combinant deux transformations :

1_Calcul de l'inverse multiplicatif dans le champ fini GF(2^8), comme expliqué dans la section 1.3.2. L'élément {00} est associé à lui-même.

2_Application de la transformation affine suivante (sur GF(2)).

$b' = bi \oplus b(i+4) \bmod 8 \oplus b(i+5) \bmod 8 \oplus b(i+6) \bmod 8 \oplus b(i+7) \bmod 8 \oplus c\,i$

Pour $0 \leq i \leq 8$, où bi représente le ième bit de l'octet, et ci est le ième bit d'un octet c ayant la valeur {63} ou {01100011}. Lorsqu'un nombre premier est apposé sur une variable (par exemple, b'), cela signifie que la variable doit être mise à jour avec la valeur correspondante à sa droite. En notation matricielle, l'élément de transformation affine de la boîte S peut être exprimé comme

$$\begin{bmatrix} b'_0 \\ b'_1 \\ b'_2 \\ b'_3 \\ b'_4 \\ b_5 \\ b'_6 \\ b'_7 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} \pm \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

Figure 3.6: Notation matricielle de s-box [20]

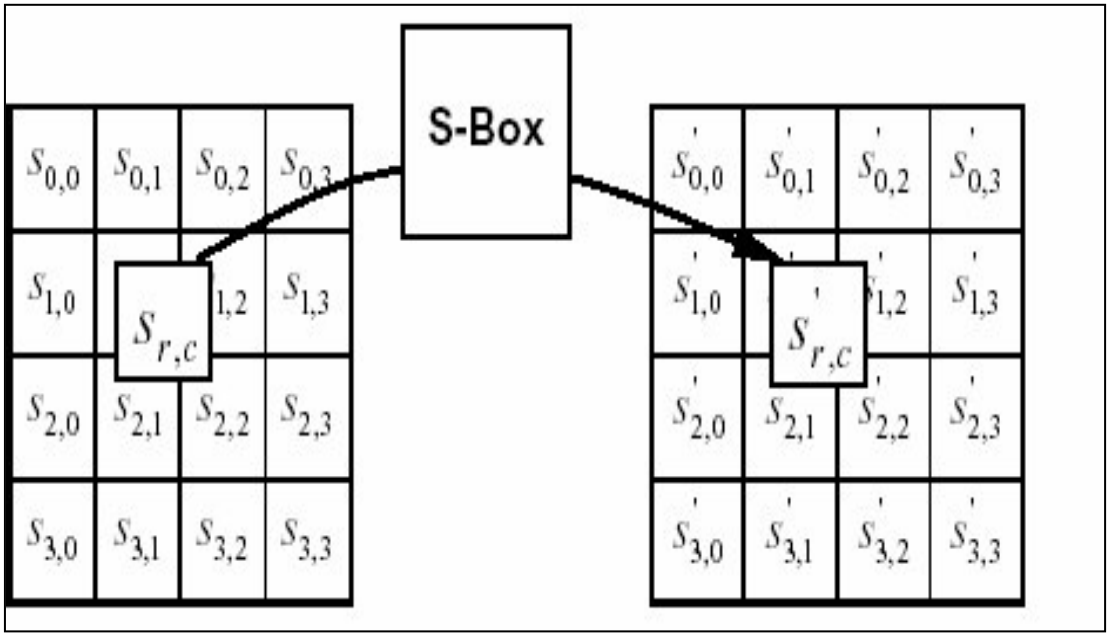



Figure 3.7: Application de S-box à chaque Byte de l'État [20]

La S-box employée dans la transformation SubBytes est représentée en format hexadécimal dans la figure 3.8. Par exemple, si f = S1,1 = {53}, alors la substitution correspondante serait obtenue en croisant la ligne marquée '5' avec la colonne marquée '3' sur la figure 3.8. Cela attribuerait à S'1, 1 une valeur de {ed}.

| | | y | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | a | b | c | d | e | f |
| x | 0 | 63 | 7c | 77 | 7b | f2 | 6b | 6f | c5 | 30 | 01 | 67 | 2b | fe | d7 | ab | 76 |
| | 1 | ca | 82 | c9 | 7d | fa | 59 | 47 | f0 | ad | d4 | a2 | af | 9c | a4 | 72 | c0 |
| | 2 | b7 | fd | 93 | 26 | 36 | 3f | f7 | cc | 34 | a5 | e5 | f1 | 71 | d8 | 31 | 15 |
| | 3 | 04 | c7 | 23 | c3 | 18 | 96 | 05 | 9a | 07 | 12 | 80 | e2 | eb | 27 | b2 | 75 |
| | 4 | 09 | 83 | 2c | 1a | 1b | 6e | 5a | a0 | 52 | 3b | d6 | b3 | 29 | e3 | 2f | 84 |
| | 5 | 53 | d1 | 00 | ed | 20 | fc | b1 | 5b | 6a | cb | be | 39 | 4a | 4c | 58 | cf |
| | 6 | d0 | ef | aa | fb | 43 | 4d | 33 | 85 | 45 | f9 | 02 | 7f | 50 | 3c | 9f | a8 |
| | 7 | 51 | a3 | 40 | 8f | 92 | 9d | 38 | f5 | bc | b6 | da | 21 | 10 | ff | f3 | d2 |
| | 8 | cd | 0c | 13 | ec | 5f | 97 | 44 | 17 | c4 | a7 | 7e | 3d | 64 | 5d | 19 | 73 |
| | 9 | 60 | 81 | 4f | dc | 22 | 2a | 90 | 88 | 46 | ee | b8 | 14 | de | 5e | 0b | db |
| | a | e0 | 32 | 3a | 0a | 49 | 06 | 24 | 5c | c2 | d3 | ac | 62 | 91 | 95 | e4 | 79 |
| | b | e7 | c8 | 37 | 6d | 8d | d5 | 4e | a9 | 6c | 56 | f4 | ea | 65 | 7a | ae | 08 |
| | c | ba | 78 | 25 | 2e | 1c | a6 | b4 | c6 | e8 | dd | 74 | 1f | 4b | bd | 8b | 8a |
| | d | 70 | 3e | b5 | 66 | 48 | 03 | f6 | 0e | 61 | 35 | 57 | b9 | 86 | c1 | 1d | 9e |
| | e | e1 | f8 | 98 | 11 | 69 | d9 | 8e | 94 | 9b | 1e | 87 | e9 | ce | 55 | 28 | df |
| | f | 8c | a1 | 89 | 0d | bf | e6 | 42 | 68 | 41 | 99 | 2d | 0f | b0 | 54 | bb | 16 |

Figure 3.8 : Valeurs S-box pour toutes les 256 combinaisons

en format hexadécimal [20]

## 4.3 LA TRANSFORMATION ShiftRows

Dans la transformation ShiftRows Shift Row (), les octets dans les trois dernières lignes de l'état sont décalés de manière cyclique sur différents nombres d'octets (offsets).

La première rangée, r = 0, n'est pas décalée. Plus précisément, la transformation ShiftRows () se déroule comme suit

$$s'_{r,c} = s_{r,(c+shift(r,Nb))} \bmod Nb \quad \text{for } 0< r < 4 \text{ and } 0 \le c \le Nb,$$

Lorsque le décalage de valeur de décalage (r, Nb) dépend du numéro de ligne, r, comme suit (Nb = 4)

$Shift(1,4) = 1$: $Shift(2,4) = 2$; $Shift(3,4) = 3$.

L'opération a pour effet de décaler les octets vers des positions "inférieures" dans la rangée (c'est-à-dire, des valeurs plus basses de c dans une rangée donnée), tandis que les octets "les plus bas" se déplacent vers le "haut" de la rangée (c'est-à-dire, des valeurs plus élevées de c dans une ligne donnée). La figure 3.8 présente la transformation ShiftRows().**[19]**

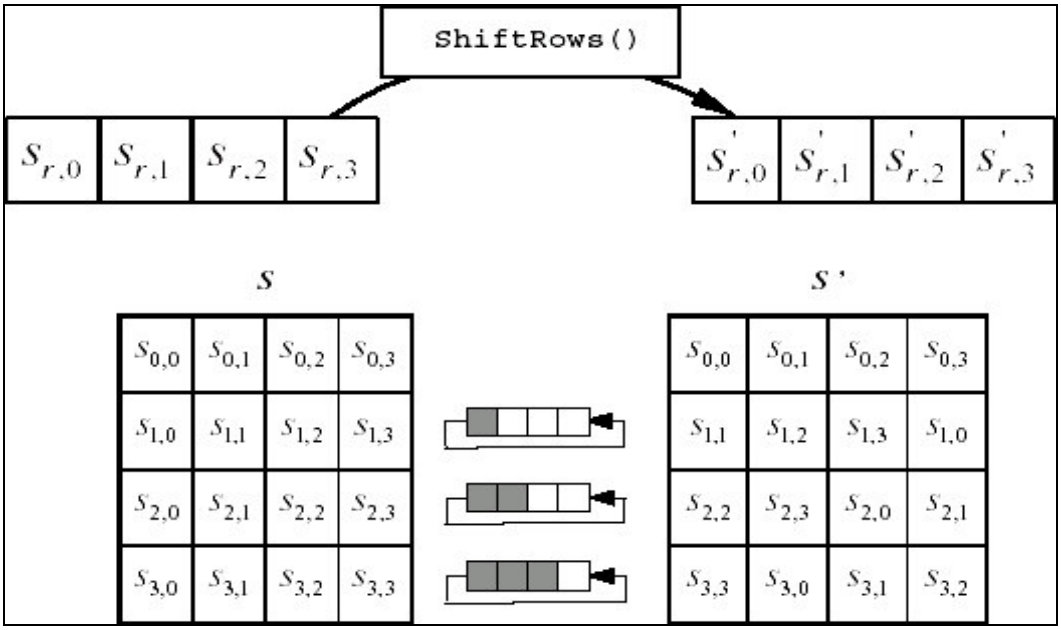



Figure 3.9 : Décalage cyclique des trois dernières rangées de l'État [20]

## 4.4 LA TRONSFORMATION MixColumns

Cette opération repose sur la multiplication dans le champ de Galois. Chaque octet d'une colonne est substitué par une valeur dépendant des quatre octets de la colonne en question. La transformation MixColumns() agit sur l'état colonne par colonne, traitant chaque colonne comme un polynôme à quatre termes, conformément à la section 1.3.4. Les colonnes sont considérées comme des polynômes sur $GF(2^8)$ et multipliées modulo x^4 + 1 par un polynôme fixe a(x), défini par l'équation suivante **[20]**.

$$a(x) = x^3 + \{01\}x^2 + \{01\}x + \{02\}.$$

Comme décrit dans la section. 1.3.4, ceci peut être écrit comme une multiplication matricielle. Laisser

$$S'(x) = a(x) \otimes S(x)$$

$$\begin{bmatrix} S'_{0,c} \\ S'_{0,c} \\ S'_{0,c} \\ S'_{0,c} \end{bmatrix} \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} S'_{0,c} \\ S'_{0,c} \\ S'_{0,c} \\ S'_{0,c} \end{bmatrix} \quad \text{Pour } 0 \leq c < Nb.$$

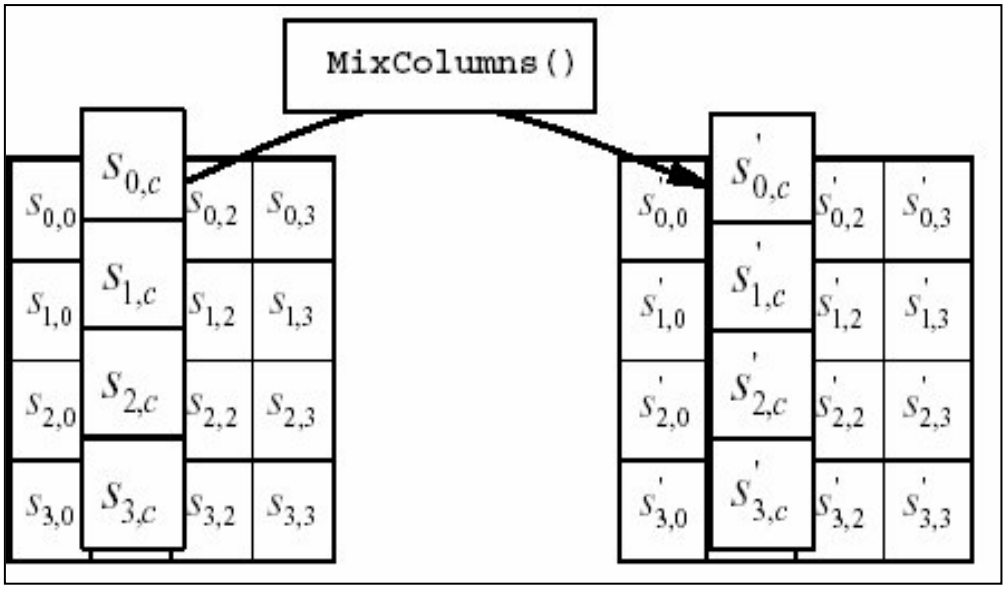
À la suite de cette multiplication, les quatre octets d'une colonne sont remplacés par les suivants

$$S'_{0,c} = (\{02\} \cdot S_{0,c}) \oplus (\{03\} \cdot S_{1,c}) \oplus S_{2,c} \oplus S_{3,c}$$

$$S'_{1,c} = S_{0,c} \oplus (\{02\} \cdot S_{1,c}) \oplus (\{03\} \cdot S_{2,c}) \oplus S_{3,c}$$

$$S'_{2,c} = S_{0,c} \oplus S_{1,c} \oplus (\{02\} \cdot S_{2,c}) \oplus (\{03\} \cdot S_{3,c})$$

$S'_{3,c}$ $S_{0,c}$ $S_{1,c}$ $S_{2,c}$ $S_{3,c}$




$$S'_{3,c} = (\{03\} \cdot S_{0,c}) \oplus S_{1,c} \oplus S_{2,c} \oplus (\{02\} \cdot S_{3,c})$$

Figure 3.10 : Mélange de colonnes de l'État [20]

## 4.5 Addition de la transformation de clé ronde

Lors de l'Addition de clé de ronde dans la transformation AddRoundKey(), une clé de ronde est combinée à l'état par une opération XOR élément par

élément. Chaque clé de ronde est formée de Nb mots issus de la génération des clés de calendrier (expliquée dans la section 2 .6 ci-dessous). Ces Nb mots sont ajoutés individuellement aux colonnes de l'état, de manière à

$$[s'_{0,c}, s'_{1,c}, s'_{2,c}, s'_{3,c},] = [s_{0,c}, s_{1,c}, s_{2,c}, s_{3,c}] \oplus [w_{round*Nb}] \qquad \text{pour } 0 \leq c \leq Nb$$

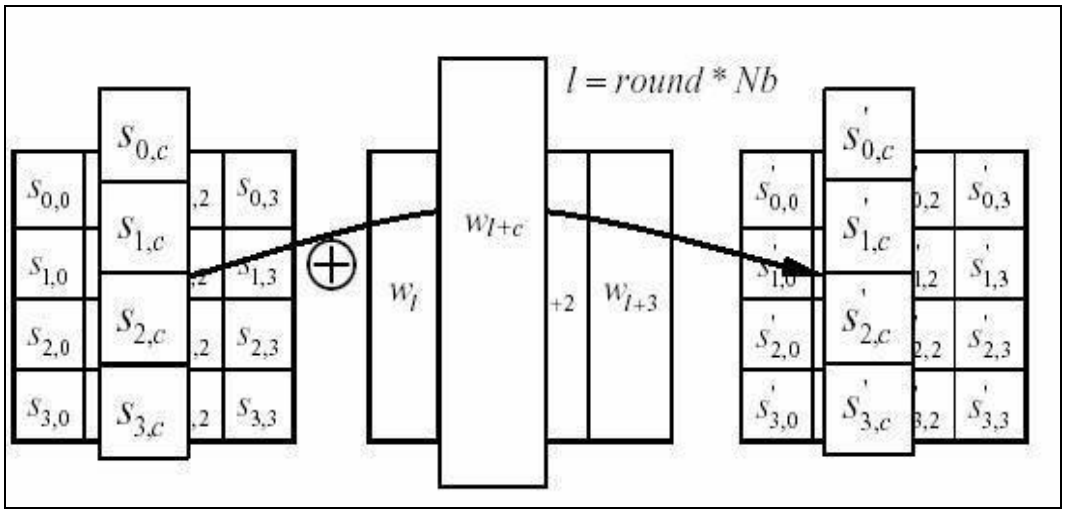



Figure 3.11 : Fonctionnement OU exclusif des mots

clés d'état et de chiffrement [20]

Pour le chiffrement, l'addition initiale de la clé ronde s'effectue lorsque round = 0, avant la première itération de la fonction round. La transformation AddRoundKey() est appliquée lors des Nr tours de chiffrement quand 1 ≤ round ≤ Nr. Cette opération est illustrée à la figure 3.11, où l = round * Nb, et l'adressage des octets dans les mots de la clé d'horaire est décrit à la section 1.2.2. Les mots [wi] correspondent aux mots de génération de clé.

## 4.6 Key schedule Expansion

Chaque clé de ronde est composée d'un tableau de 4 mots (128 bits) généré en combinant la clé de ronde précédente, une constante évolutive à chaque tour, et une série de recherches dans la S-Box (figure 3.7) pour chaque mot de 32 bits de la clé. Le premier mot de la clé reste identique à l'entrée initiale de l'utilisateur. Chaque octet (wo – w3) de la clé initiale subit une opération XOR avec une constante dépendant du tour en cours, et le résultat de la recherche dans la S-Box pour wi, pour former la

prochaine clé de ronde. Le nombre de tours requis pour trois longueurs de clé différentes est indiqué dans le tableau 3.2.

| | Longueur de clé (Nk mots) | Taille de bloc (Nb mots) | Nombre de Rondes (Nr) |
|---|---|---|---|
| AES-128 | 4 | 4 | 10 |
| AES-192 | 6 | 4 | 12 |
| AES-256 | 8 | 4 | 14 |

Tableau 3-2 : Bloc-clé- Combinaisons rondes [16]

Key Expansion un ensemble total de Nb(Nr + 1) mots : l'algorithme requiert un ensemble initial de Nb mots, et chaque tour parmi les Nr tours nécessite Nb mots de clé. Le programme de clés obtenu se compose d'un tableau linéaire de mots de 4 octets, noté [wi], où i est compris dans l'intervalle

$0 \leq i < Nb(Nr + 1)$.

# 5 DÉCRYPTAGE

## 5.1 Processus de décryptage

Le déchiffrement de l'algorithme AES (Advanced Encryption Standard) est décrit dans la figure 3.12, et implique l'exécution des opérations inverses des quatre fonctions de chiffrement, dans l'ordre inverse. Chaque fonction utilisée dans le processus de chiffrement possède une fonction inverse dédiée pour le déchiffrement : InvShiftRows, InvSubBytes et InvMixColumns. La fonction AddRoundKey reste inchangée. Comme pour le chiffrement, le processus de déchiffrement comprend une étape initiale, des tours intermédiaires et un tour final. L'étape initiale consiste à appliquer la fonction AddRoundKey à la matrice d'état. Les tours intermédiaires exécutent, dans l'ordre, les fonctions InvShiftRows, InvSubBytes, AddRoundKey et InvMixColumns sur la matrice d'état. Le tour final se distingue des tours intermédiaires

par l'omission de la fonction InvMixColumns dans la séquence des transformations.

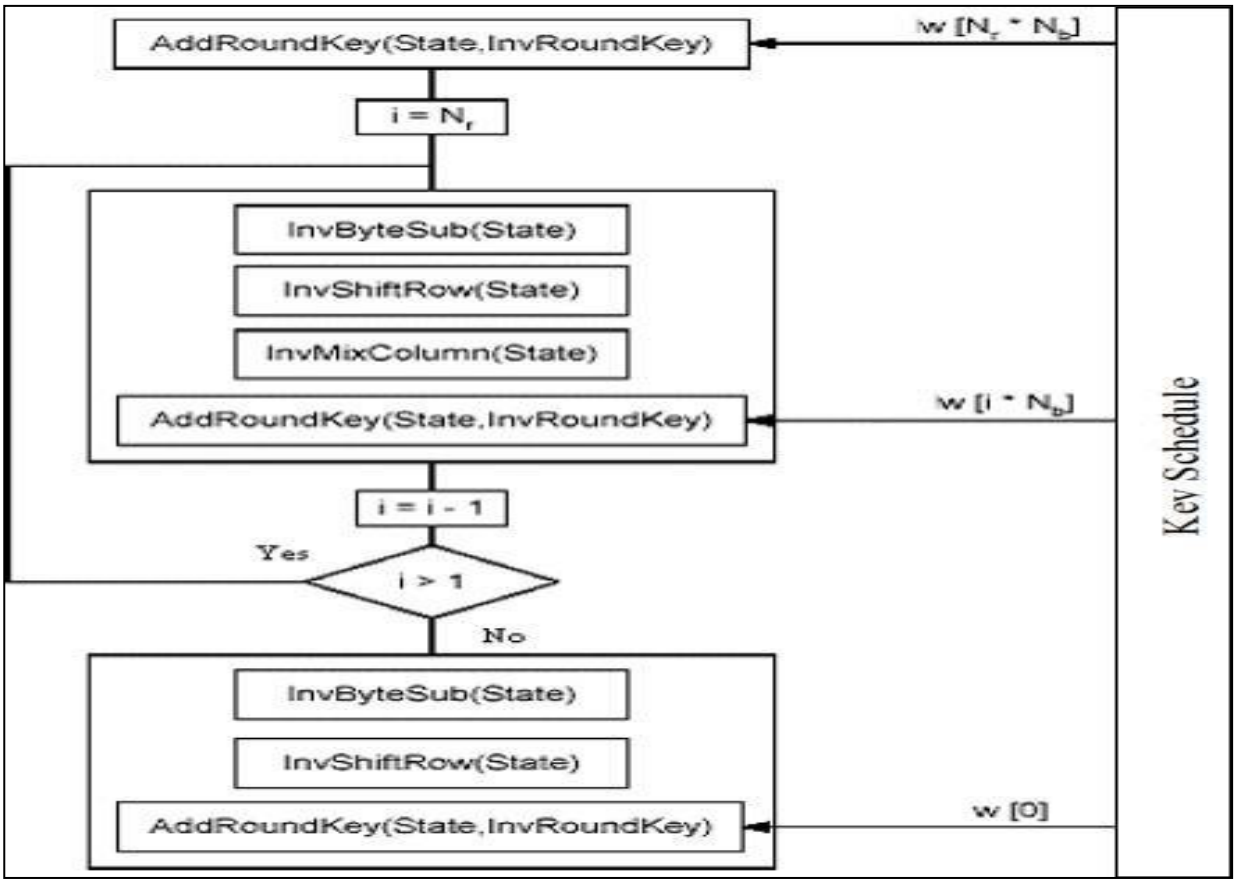



Figure 3.12: Processus de Décryptage [16]

Le processus de déchiffrement est le contraire direct du processus de chiffrement. Toutes les transformations effectuées lors du chiffrement sont inversées dans ce processus. Ainsi, les dernières valeurs obtenues lors du chiffrement, tant pour les données que pour la clé, deviennent les premières entrées du premier tour du déchiffrement et sont traitées dans l'ordre inverse.

## 5.2 Transformation InvSubBytes

La transformation InvSubBytes() est la réciproque de la substitution des octets, où la S-Box inverse (figure 3.13) est utilisée sur chaque octet de l'état. Cela est réalisé en appliquant l'inverse de la transformation affine de l'équation (16), suivi de l'inverse multiplicatif dans GF(2^8).

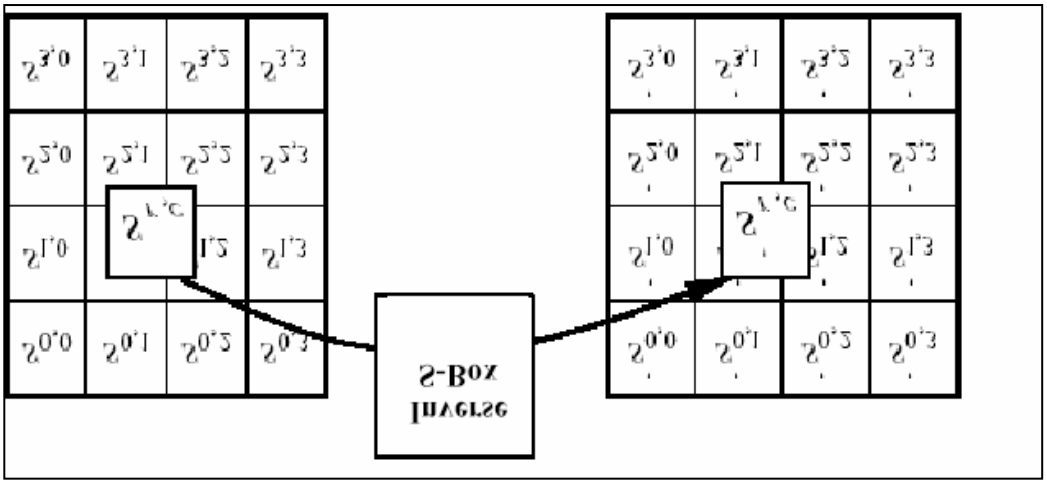


Figure 3.13: Application de la boîte S inverse à chaque octet de l'État [20]

| | | y | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | a | b | c | d | e | f |
| x | 0 | 52 | 09 | 6a | d5 | 30 | 36 | a5 | 38 | bf | 40 | a3 | 9e | 81 | f3 | d7 | fb |
| | 1 | 7c | e3 | 39 | 82 | 9b | 2f | ff | 87 | 34 | 8e | 43 | 44 | c4 | de | e9 | cb |
| | 2 | 54 | 7b | 94 | 32 | a6 | c2 | 23 | 3d | ee | 4c | 95 | 0b | 42 | fa | c3 | 4e |
| | 3 | 08 | 2e | a1 | 66 | 28 | d9 | 24 | b2 | 76 | 5b | a2 | 49 | 6d | 8b | d1 | 25 |
| | 4 | 72 | f8 | f6 | 64 | 86 | 68 | 98 | 16 | d4 | a4 | 5c | cc | 5d | 65 | b6 | 92 |
| | 5 | 6c | 70 | 48 | 50 | fd | ed | b9 | da | 5e | 15 | 46 | 57 | a7 | 8d | 9d | 84 |
| | 6 | 90 | d8 | ab | 00 | 8c | bc | d3 | 0a | f7 | e4 | 58 | 05 | b8 | b3 | 45 | 06 |
| | 7 | d0 | 2c | 1e | 8f | ca | 3f | 0f | 02 | c1 | af | bd | 03 | 01 | 13 | 8a | 6b |
| | 8 | 3a | 91 | 11 | 41 | 4f | 67 | dc | ea | 97 | f2 | cf | ce | f0 | b4 | e6 | 73 |
| | 9 | 96 | ac | 74 | 22 | e7 | ad | 35 | 85 | e2 | f9 | 37 | e8 | 1c | 75 | df | 6e |
| | a | 47 | f1 | 1a | 71 | 1d | 29 | c5 | 89 | 6f | b7 | 62 | 0e | aa | 18 | be | 1b |
| | b | fc | 56 | 3e | 4b | c6 | d2 | 79 | 20 | 9a | db | c0 | fe | 78 | cd | 5a | f4 |
| | c | 1f | dd | a8 | 33 | 88 | 07 | c7 | 31 | b1 | 12 | 10 | 59 | 27 | 80 | ec | 5f |
| | d | 60 | 51 | 7f | a9 | 19 | b5 | 4a | 0d | 2d | e5 | 7a | 9f | 93 | c9 | 9c | ef |
| | e | a0 | e0 | 3b | 4d | ae | 2a | f5 | b0 | c8 | eb | bb | 3c | 83 | 53 | 99 | 61 |
| | f | 17 | 2b | 04 | 7e | ba | 77 | d6 | 26 | e1 | 69 | 14 | 63 | 55 | 21 | 0c | 7d |

Figure 3.14 : Valeurs S-box inverses pour toutes les 256 combinaisons en format hexadécimal [20]

## 5.3 Transformation InvShiftRows

Les octets des trois dernières lignes de l'état subissent des décalages cycliques sur un nombre spécifique d'octets. La première ligne, r = 0, reste inchangée. Les trois lignes inférieures subissent des décalages cycliques d'un

certain nombre d'octets défini par Nb-shift(r, Nb), où la valeur de décalage (r, Nb) varie en fonction du numéro de la ligne et est détaillée dans la section 2.3. Concrètement, la transformation InvShiftRows() est implémentée de la manière suivante:

$$S'_{r,(c+shift(r,Nb)) \bmod Nb} = S_{r,c} \quad \text{pour } 0 \le r<4 \text{ and } 0 \le c<Nb$$

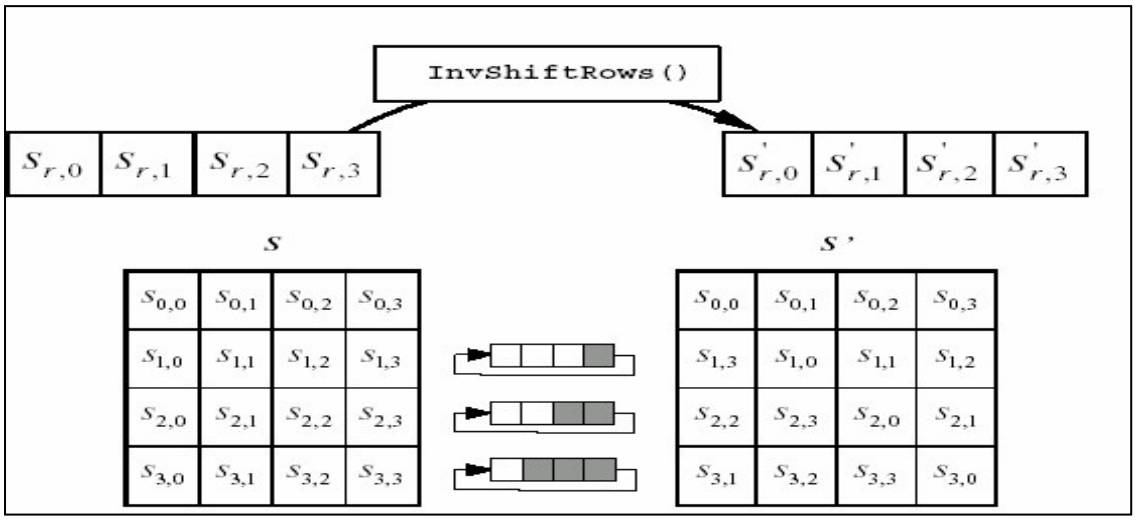



Figure 3.15 : Décalage cyclique inverse des trois dernières rangées de l'État **[20]**

## 5.4 Transformation InvMixColumns

InvMixColumns opère sur l'état colonne par colonne, en traitant chaque colonne comme un polynôme à quatre termes comme décrit dans la section.1.3.4. Les colonnes sont considérées comme des polynômes sur GF ($2^8$) et multipliées modulo $x^4 + 1$ avec un polynôme fixe $a^{-1}(x)$, donné par

$$a^{-1}(x) = \{0b\}x^3 + \{0d\}x^2 + \{09\}x + \{0e\}.$$

$$\begin{bmatrix} S'_{0,c} \\ S'_{0,c} \\ S'_{0,c} \\ S'_{0,c} \end{bmatrix} = \begin{bmatrix} 0e & 0b & 0d & 09 \\ 09 & 0e & 0b & 0d \\ 0d & 09 & 0e & 0b \\ 0b & 0d & 09 & 0e \end{bmatrix} \begin{bmatrix} S_{0,c} \\ S_{0,c} \\ S_{0,c} \\ S_{0,c} \end{bmatrix} \quad \text{Pour } 0 \le c \le Nb$$

Comme décrit dans la section 1.3.4, ceci peut être écrit comme une multiplication matricielle. Laisser $S'(x) = a^{-1}(x) \otimes S(x)$

À la suite de cette multiplication, les quatre octets d'une colonne sont remplacés par les équations suivantes.

$$S'_{0,c} = (\{0e\} \bullet S_{0,c}) \oplus (\{0b\} \bullet S_{1,c}) \oplus (\{0d\} \bullet S_{2,c}) \oplus (\{09\} \bullet S_{3,c})$$

$$S'_{1,c} = (\{09\} \bullet S_{0,c}) \oplus (\{0e\} \bullet S_{1,c}) \oplus (\{0b\} \bullet S_{2,c}) \oplus (\{0d\} \bullet S_{3,c})$$

$$S'_{2,c} = (\{0d\} \bullet S_{0,c}) \oplus (\{09\} \bullet S_{1,c}) \oplus (\{0e\} \bullet S_{2,c}) \oplus (\{0b\} \bullet S_{3,c})$$

$$S'_{3,c} = (\{0b\} \bullet S_{0,c}) \oplus (\{0d\} \bullet S_{1,c}) \oplus (\{09\} \bullet S_{2,c}) \oplus (\{0e\} \bullet S_{3,c})$$

Par conséquent, l'État peut être représenté

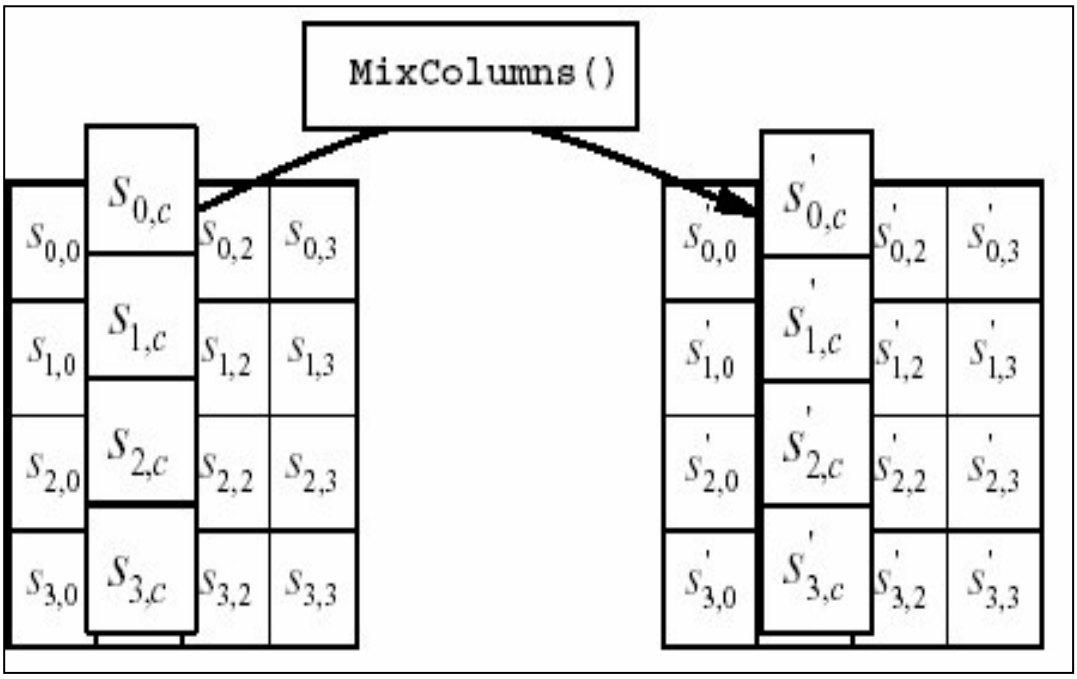



Figure 3.16 : Opération de Inverse Mix Column sur l'état [20]

## 6 Conclusion

Ce chapitre a décrit en détail le fonctionnement de l'algorithme AES (Advanced Encryption Standard) pour les modes de chiffrement et de déchiffrement. Nous avons identifié les étapes des algorithmes respectifs, notant que pour une clé de 128 bits, il y a 10 tours (rounds). Chaque round comprend une fonction non linéaire SubBytes() et des fonctions linéaires ShiftRows(), MixColumns(), suivies de AddRoundKey() qui effectue un OU exclusif sur les colonnes de la matrice d'état avec la clé ronde. La fonction KeyExpansion() permet de dériver les clés de round à partir de la clé de chiffrement initiale. Le

déchiffrement inverse les opérations citées. Dans le prochain chapitre, nous traitons la bibliothèque ffmpeg.

# Chapitre 4

# ETUDE LA BIBLIOTYEQUE FFmpeg

# 1 Introduction

Le traitement de la vidéo a pour but l'amélioration la qualité, l'édition, la compression/décompression pour stockage ou transmission, l'extraction d'information ... etc. Il s'agit d'une branche dérivée du traitement d'images, qui réutilise beaucoup de ses concepts. Bien qu'il soit courant de « traiter » une vidéo en traitant en réalité chacune de ses images (ou frames) indépendamment, le traitement vidéo consiste à priori à tirer également parti de la relation qui lie chacune des images à ses précédentes et ses suivantes, par évolution de mouvements ou autres modifications graduelles dans le temps. Ce traitement nécessite des outils performants, fiables et aussi rapides. Dans ce chapitre, nous allons présenter un de ces outils qui est la bibliothèque FFmpeg.

# 2 Présentation de l'outil FFmpeg

## 2.1 Définition

Fast Forward MPEG (FFmpeg) est une bibliothèque informatique gratuite qui permet le traitement des fichiers audio et vidéo. Le traitement est basé sur des lignes de commande et est particulièrement apprécié pour sa rapidité lors du codage et décodage des fichiers audio et vidéo. L'utilisation de cette bibliothèque peut être basée sur cmd (l'invite de commandes est un logiciel d'interprétation des commandes, affiche une interface utilisateur en ligne de commande, ou sur l'introduction manuelle des commandes dans l'invite de commande). Afin de comprendre les principes fondamentaux du logiciel et de se familiariser avec son utilisation sans l'interface graphique, l'utilisateur peut ajuster les commandes à ses préférences **[21]**

## 2.2 Installation du FFmpeg

FFmpeg est une bibliothèque très puissante qui peut être installée non seulement sous Windows, Linux mais aussi sous d'autres systèmes d'exploitation. Dans ce qui suit, nous allons décrire les méthodes d'installation sous Windows ainsi que sous linux **[22]**

### 2.2.1 Installation sous Windows

Pour l'installation de FFmpeg sous Windows il suffit de suivre les étapes suivantes :

1. Télécharger FFmpeg : Il suffit de consulter le lien suivant

http://ffmpeg.zeranoe.com/builds/ et de télécharger la version statique 32 ou 64 bits (selon votre système).

2. Décompresser le fichier : Pour rendre la taille du téléchargement agréable et petite, il est compressé dans un fichier .7z, qui ressemble à un fichier .zip mais plus petit.

3. Décompressez-le dans un dossier facile à trouver, comme directement sur votre lecteur C: \. Il devrait créer un dossier du type ffmpeg-20140530-git-98a6806-win64-static , mais renommez-le simplement en ffmpeg pour simplifier.

4. Ajouter au chemin : Enfin, nous devons ajouter le dossier bin , qui contient le fichier ffmpeg.exe , à notre chemin système pour nous permettre d'exécuter facilement les commandes.

5. Dans le menu Démarrer, cliquez avec le bouton droit de la souris sur Ordinateur et choisissez Propriétés. Ensuite, sélectionnez Paramètres système avancés. Ouvrez les variables d'environnement et éditez ensuite la variable Path.

6. Le chemin est juste une liste de dossiers contenant des commandes que vous êtes autorisés à utiliser sans taper le chemin complet des fichiers exe.

7. Ajoutez alors C:\ffmpeg\bin à la fin de la ligne, en vous assurant qu'il y a un pointvirgule ( ;) après le dossier précédent:

8. 5: Utilisez-le! C'est tout!

### 2.2.2 Installation Sous linux

1. Il faut installer FFmpeg via le PPA ou Personal Package Archives (proposer facilement et rapidement les versions récentes de leurs logiciels aux utilisateurs d'Ubuntu.) recommandé dans le blog officiel.

2. Ouvrez un nouveau terminal (CTRL + ALT + T).

3. Puis exécutez les commandes suivantes.

```
$ sudo add-apt-repository ppa: mc3man / trusty-media
$ sudo apt-get update
$ sudo apt-get install ffmpeg
$ ffmpeg –version
```

Lorsque vous exécutez la commande sudo apt update, votre système utilise **l'outil APT** ou Advanced Packaging Tool qui est un système complet et avancé de gestion de paquets pour effectuer une vérification par rapport au référentiel et stocke les

informations relatives au 24 logiciel et à leur version dans une mémoire cache. Lorsque vous utilisez la commande sudo apt install nom_package, elle utilise ces informations pour extraire ce package à partir de l'URL où le logiciel réel est stocké.

## 2.3 Principe de fonctionnement

ffmpeg repose sur un principe très simple. Chaque fichier d'entrée sera démultiplexé grâce à la bibliothèque libavformat. Les paquets ainsi encodés sont passés au décodeur qui va produire des frames non compressées. Une fois cette étape terminée, les paquets sont réencodés puis multiplexés pour devenir les fichiers de sortie (voir la figure 4.1). **[23]**

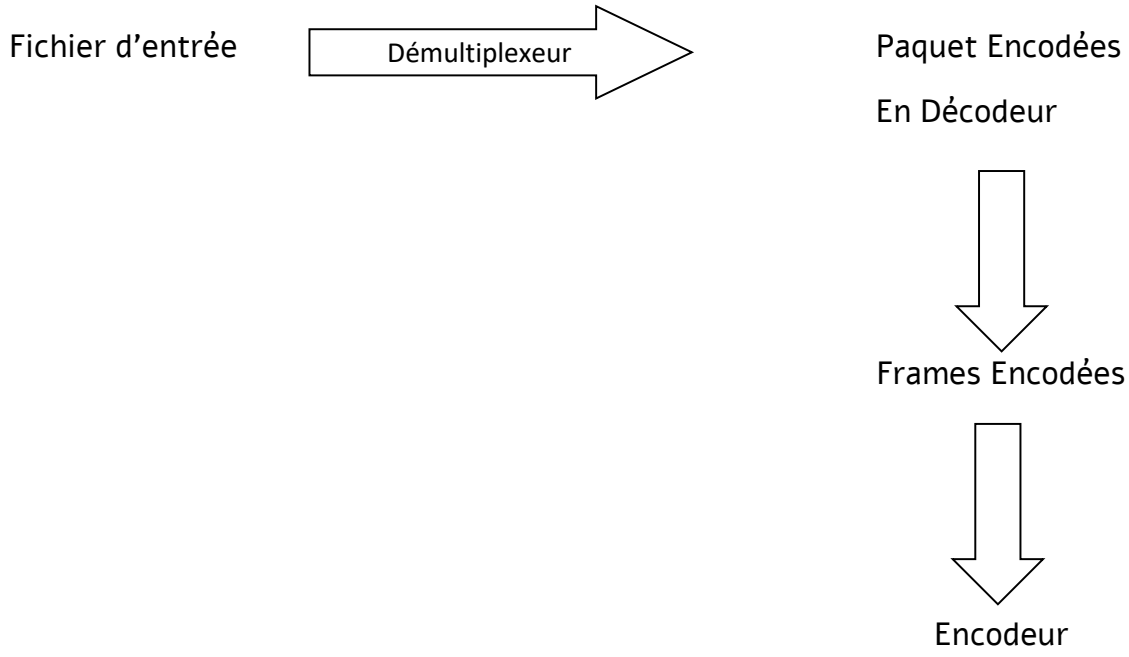



Figure 4.1: Principe de fonctionnement du FFmpeg

## 2.4 Les bibliothèques

### 2.4.1 Définition

Une bibliothèque logicielle est une collection de routines, qui peuvent être déjà compilées et prêtes à être utilisées par des programmes. Les bibliothèques sont enregistrées dans des fichiers semblables, voire identiques aux fichiers de programmes, sous la forme d'une collection de fichiers de code objet rassemblés accompagnés d'un index permettant de retrouver facilement chaque routine. **[24]**

### 2.4.2 Bibliothèques utilisées par FFmpeg

**Libavutil** : La bibliothèque libavutil est une bibliothèque d'utilitaires facilitant la programmation multimédia portable. Elle contient des fonctions, des générateurs de nombres aléatoires, des structures de données, des fonctions mathématiques supplémentaires, des fonctions de cryptographie et multimédias (telles que des énumérations pour les formats de pixels et d'échantillons).

**Libswscale** : La bibliothèque libswscale effectue une mise à l'échelle hautement optimisée des opérations ainsi que des opérations de conversion d'espaces colorimétriques et de formats de pixels.

**Libswresample** : La bibliothèque libswresample effectue des opérations de ré-échantillonnage audio, de rematriçage et de conversion de format d'échantillon hautement optimisées.

**Libavcodec** : La bibliothèque libavcodec fournit un cadre générique de codage / décodage et contient plusieurs codeurs et décodeurs pour les flux audio, vidéo ainsi que plusieurs filtres de flux de données.

**Libavformat** : La bibliothèque libavformat fournit un cadre générique pour le multiplexage et le démultiplexage (multiplexage et démultiplexage) des flux audio, vidéo et de sous-titres. Elle englobe plusieurs multiplexeurs et démultiplexeurs pour les formats de conteneur multimédia.

**Libavdevice** : La bibliothèque libavdevice fournit un cadre générique pour la saisie et le rendu sur de nombreux périphériques d'entrée/sortie multimédia courants, et prend en charge plusieurs périphériques d'entrée et de sortie.

**Libavfilter** : La bibliothèque libavfilter fournit un cadre de filtrage audio / vidéo générique contenant plusieurs filtres. **[25]**

## 2.5 Les formats

Le format est le container qui permet le transport de la vidéo, du son et des sous-titres soit sous forme de fichier (mkv, mov...) soit sous forme de flux (MPEG TS). A l'intérieur d'un container on peut insérer (muxer) ou extraire (demuxer):

- Un ou plusieurs flux vidéo (un film, ou des chaînes de télévision....)
- Un ou plusieurs flux audio (la version originale, la version française, ... )
- Un ou plusieurs flux de sous-titres. (français, sourd et malentendant, ...)

- Plus des métadonnées (titre, nom de l'artiste par exemple)

On parle de multiplexer les différentes pistes (flux ou stream) dans un format. FFmpeg fournit une liste des formats qu'il supporte : **ffmpeg -formats**

```
DE avi             AVI (Audio Video Interleaved)
DE ogg             Ogg
D  matroska,webm   Matroska / WebM
 E mov             QuickTime / MOV
D  mov,mp4,m4a,3gp,3g2,mj2 QuickTime / MOV
 E webm            WebM
```

Figure 22 : Résultat de la commande ffmpeg -formats

Le D signifie la capacité à le lire, et E la possibilité d'encapsuler dans le format. **[26]**

## 2.6 Les codecs

Un codec est un algorithme qui permet d'encoder la vidéo ou le son afin de l'adapter au protocole de transport (IP, DVB, fichier...) notamment en réduisant le débit (Kbits/s). Selon les codecs, la compression peut s'accompagner d'une perte de qualité dans l'image ou le son plus ou moins importante. De la même manière que pour les formats, ffmpeg liste les codecs qu'il est capable de gérer: **ffmpeg –codecs** .

```
Codecs:
 D..... = Decoding supported
 .E.... = Encoding supported
 ..V... = Video codec
 ..A... = Audio codec
 ..S... = Subtitle codec
 ...I.. = Intra frame-only codec
 ....L. = Lossy compression
 .....S = Lossless compression
```

Figure 4.3: Principaux codecs du ffmpeg

Puisqu'il y en a beaucoup, on va citer les plus connus :

DEV.L. h261 H.261

DEV.L. h263 H.263 / H.263-1996, H.263+ / H.263-1998 / H.263 version 2

D.V.L. h263i Intel H.263

DEV.L. h263p H.263+ / H.263-1998 / H.263 version 2

DEV.LS h264 H.264 / AVC / MPEG-4 AVC / MPEG-4 part 10 (decoders: h264 h264_qsv h264_cuvid ) (encoders: libx264 libx264rgb h264_amf h264_nvenc h264_qsv nvenc nvenc_h264 )

DEV.L. hev H.265 / HEVC (High Efficiency Video Coding) (decoders: hevc hevc_qsv hevc_cuvid ) (encoders: libx265 nvenc_hevc hevc_amf hevc_nvenc hevc_qsv )

D.V.L. hnm4video HNM 4 video **[27]**

## 2.7 Les filtres

ffmpeg dispose aussi d'une base importante de filtres qui permettent de modifier le contenu de chaque flux, comme changer la résolution, modifier le volume d'une piste, incruster un logo etc.... ffmpeg liste les filtres qu'il est capable de gérer: **ffmpeg -filters** Si on résume, ffmpeg permet de multiplexer ou de-multiplexer dans différents formats:

- Des flux vidéos compressés (ou pas),
- Des flux audio compressés (ou pas),
- Des sous-titres dans différents formats. **[28]**

## 2.8 Principales Commandes FFmpeg

-**L** : Afficher la licence.

-**h**, - **?**, -**help**, --**help**: Afficher l'aide.

-**formats** : **P**our obtenir une liste de tous les multiplexeurs et démultiplicateurs.

-**filters** : Pour obtenir une liste de tous les filtres.

-**version** : Montrer la version.

-**demuxers** : Afficher les démultiplicateurs disponibles.

-**muxers** : Afficher les multiplexeurs disponibles.

-**dispositifs** : Afficher les appareils disponibles.

-**codecs** : Affiche tous les codecs connus de libavcodec.

-**décodeurs** : Afficher les décodeurs disponibles.

-**codeurs** : Afficher tous les encodeurs disponibles.

-**bsfs** : Afficher les filtres de flux binaires disponibles.

-**protocoles** : Afficher les protocoles disponibles.

-**filtres** : Afficher les filtres libavfilter disponibles.

-**pix_fmts** : Afficher les formats de pixels disponibles.

-**sample_fmts** : Afficher les formats d'échantillons disponibles.

-**layouts** : Afficher les noms de canaux et les dispositions de canaux standards.

-**couleurs** : Afficher les noms de couleurs reconnues

**Ffplay** : visualisation de la vidéo.

**Ffprobe** : Avoir des informations sur la vidéo. **[29]**

## 2.9 Liste des traitements dont le FFmpeg est capable

- Obtenir des infos sur un fichier vidéo : **ffmpeg -i video.avi**.
- Transformer une série d'images en vidéo : **ffmpeg -f image2 -i image%d.jpg video.mpg**
- Extraire une série images d'une vidéo :**ffmpeg -i video.mpg image%d.jpg**
- Extraire le son d'une vidéo et l'enregistrer en mp3 : **ffmpeg -i video_origine.avi son_final.mp3**
- Convertir un fichier avi en mpeg :**ffmpeg -i video_origine.avi video_finale.mpg**
- Convertir un mpeg en avi :**ffmpeg -i video_origine.mpg video_finale.avi**
- Associer une vidéo et un son pour créer une vidéo sonorisée :**ffmpeg -i son.wav -i video_origine.avi video_finale.mpg**
- **Mixer un son et une vidéo :ffmpeg -i son.wav -i video_origine.avi video_finale.mpg**
- Convertir une vidéo mp4 en avi en conservant la même qualité : **ffmpeg -i video.mp4 -qscale 0 video.avi**
- Supprimer le son d'une vidéo : **ffmpeg -i video.mp4 -c copy -an video-sansson.mp4**
- Couper une vidéo : **ffmpeg -i sequenceacouper.avi -ss 00:39:45.00 -t 00:00:30.00 -c:v copy -c:a copy nouvellesequence.avi**
- Changer de Format : **ffmpeg -i input.mkv -c copy output.mov [30]**

| Commandes Actions | Commandes Actions |
|---|---|
| -i | spécifie le fichier d'entrée |
| -c copy | copie à l'identique la totalité des flux |
| -c :v copy | copie à l'identique les pistes vidéos |
| -c :a copy | copie à l'identique les pistes audio |
| -c :s copy | copie à l'identique les pistes sous-titres |

Tableau 4-1 : Signification des commandes

- **Encoder la vidéo**

| Option | Explication de l'option |
|---|---|
| -r | définit le nombre d'images par seconde |
| -s | configuration de la taille du cadre d'affichage |
| -aspect | configuration du format d'affichage |
| -vcodec ou -c:v | décision du choix du codec |
| -pass | nombre de passage à l'encodage |
| -crf | permet de définir un niveau de qualité entre 0 et 51 (petit nombre = meilleure qualité mais plus de temps de calcul) (défaut 23) |

Tableau 4-2 : Options d'encodage de la vidéo

- **Encoder le son**

| Option | explication de l'option |
|---|---|
| -acodec ou -c:a | détermine le choix du codec |
| -ar | configuration de la fréquence d'échantillonnage (44100 Hz) |
| -ab | configuration du débit binaire par défaut 64 kbps |
| -ac | configure le nombre de canaux (mono-stéréo) |

Tableau 4-3 : Options d'encodage de l'audio

## 2.10 CRF (Constant Rate Factor)

Logarithmique entre 0 et 51. Une différence de 6 points double ou divise par 2 environ la taille du fichier final. Un petit nombre égal une meilleure qualité mais plus de temps de calcul, la valeur par défaut est souvent 23. Le choix du CRF dépend du type d'image à encoder, de la résolution de l'image, de la qualité souhaitée ou encore de la taille du fichier désirée.

## 2.11 QP (quantization parameter)

La quantification dans un codeur est contrôlée par un paramètre de quantification, QP, compris entre 0 et 51. Lorsque le QP est définie directement, il reste constant tout au long du codage et chaque image est compressée en fonction de la valeur définie. **[31]**

# 3 Conversion de Formats Vidéo et Audio Utilisant FFmpeg

## 3.1 Commande de base

Pour convertir un fichier multimédia d'un format à un autre, vous pouvez utiliser une commande simple avec la structure suivante :

```
ffmpeg -i [input_file_name] [output_file_name]
```

L'argument -i spécifie le fichier d'entrée. Après le nom du fichier d'entrée, vient le nom du fichier de sortie qui peut être de n'importe quel type d'extension de fichier (par exemple .mkv, .mp4, .mp3). FFmpeg détectera automatiquement le format d'entrée et le convertira dans le format de sortie spécifié par l'extension de fichier.

Jetons un coup d'œil à quelques exemples...

### 3.1.1Conversion de MKV en MP4

MKV, également connu sous le nom de Matroska Video, est un format conteneur multimédia qui peut contenir un nombre illimité de pistes vidéo, audio, image ou sous-titres dans un seul fichier. Comme les fichiers MKV ont une grande capacité de stockage, ils sont souvent utilisés pour stocker du contenu haute définition avec des codecs sans perte ou à débit binaire élevé. Cependant, le MKV n'est pas largement pris en charge, il ne peut pas être lu sur les lecteurs web et est uniquement pris en charge par certains lecteurs locaux, comme VLC Player, KMPlayer et MPC-HC.

En raison de la prise en charge limitée, convertir les fichiers MKV en un format plus largement pris en charge comme le MP4 facilite la distribution de contenu. Pour convertir un fichier MKV en MP4 à l'aide de FFmpeg, utilisez la commande suivante:

```
ffmpeg -i input.mkv output.mp4
```

Vous pouvez ajouter l'option -c copy dans la commande pour indiquer à FFmpeg de copier les flux vidéo et audio du fichier d'entrée vers le fichier de sortie sans les réencoder.

```
ffmpeg -i input.mkv -c copy output.mp4
```

Cela rend la conversion plus rapide et garantit une conversion sans perte.

### 3.1.2 Conversion de MOV en MP4

MOV est l'extension de nom de fichier pour le format de fichier multimédia QuickTime. Bien que MOV ait été initialement développé pour les Mac d'Apple, il est désormais compatible avec Windows 10 ou ultérieur - il peut être lu nativement en utilisant Microsoft Photos. Cela dit, si vous souhaitez garantir la compatibilité sur d'autres plates-formes et appareils, MP4, le format de fichier vidéo le plus courant, reste un choix plus sûr.

En outre, le MP4 est également connu pour sa compression efficace, ce qui peut entraîner des tailles de fichier plus petites. Pour convertir un fichier MOV en MP4 en utilisant FFmpeg, utilisez la commande suivante:

```
ffmpeg -i input.mov output.mp4
```

De même, vous pouvez ajouter l'option -c copy dans votre commande pour convertir le fichier MOV en MP4 sans les réencoder.

### 3.1.3 Conversion de WebM en MP4

WebM est un autre format vidéo populaire. Sa structure de fichier est basée sur le conteneur Matroska et comprend des flux vidéo compressés avec les codecs vidéo VP8 ou VP9 et des flux audio compressés avec les codecs audio Vorbis ou Opus. Bien que WebM soit un sous-ensemble de Matroska, il est conçu pour une utilisation sur le web et le streaming. Par conséquent, il est couramment utilisé pour les vidéos HTML5.

Cependant, les vidéos WebM ne peuvent pas être lues nativement sur certains appareils Apple. Vous devez installer un lecteur multimédia tiers pour lire les vidéos WebM, ce qui le rend moins pratique par rapport au MP4. Pour convertir un fichier WebM en MP4 à l'aide de FFmpeg, utilisez la commande suivante:

```
ffmpeg -i input.webm output.mp4
```

Cela convertira la vidéo WebM en MP4 et augmentera sa compatibilité avec différents appareils et plates-formes.

### 3.1.4 Conversion de MP4 en MP3

En plus de convertir des fichiers vidéo d'un format à un autre, vous pouvez également les convertir en d'autres formats multimédias comme le MP3. Cela peut être utile pour réutiliser le contenu car seul l'audio sera extrait et enregistré dans le fichier de sortie. Par exemple, l'audio d'un fichier vidéo peut être extrait et utilisé comme contenu pour un podcast.

Pour convertir un fichier MP4 en MP3 ou extraire l'audio d'un fichier vidéo à l'aide de FFmpeg, utilisez la commande suivante :

```
ffmpeg -i input.mp4 output.mp3
```

Cela encodera l'audio avec le codec audio libmp3lame par défaut et l'enregistrera sous forme de fichier MP3. Cela encodera l'audio avec le codec audio libmp3lame par défaut et l'enregistrera sous forme de fichier MP3.

## 3.2 Commande avancée

Les commandes ci-dessus convertissent les fichiers d'entrée en formats de fichier spécifiés en utilisant les options par défaut. Vous pouvez ajouter d'autres options à la commande pour personnaliser davantage le fichier de sortie, comme spécifier le codec, le taux de trame, la résolution, et plus encore pour ajuster la qualité de sortie.

### 3.2.1 Changement de Codec Vidéo/Audio

Le codec d'un fichier multimédia peut influencer sa qualité, sa taille et sa compatibilité. FFmpeg prend en charge une large gamme de codecs vidéo et audio que vous pouvez spécifier dans la commande en utilisant l'option -c.

Pour spécifier le codec vidéo, vous pouvez utiliser l'option -vcodec ou -codec:v / -c:v. Par exemple, la commande ci-dessous utilise le codec libx264 pour encoder le fichier vidéo de sortie :

```
ffmpeg -i input.mov -vcodec libx264 output.mp4
```

De même, vous pouvez spécifier le codec audio pour un fichier audio en utilisant l'option -acodec ou -codec:a / -c:a :

```
ffmpeg -i input.mp4 -acodec libshine output.mp3
```

Vous pouvez également spécifier à la fois les codecs vidéo et audio dans la même commande pour les fichiers vidéo contenant un flux audio :

```
ffmpeg -i input.mov -vcodec libx264 -acodec libshine ouput.mp4
```

### 3.2.2 Ajustement de la Qualité Vidéo/Audio

De nombreux facteurs influent sur la qualité d'un fichier vidéo/audio et le débit binaire en est un. Le débit binaire fait référence à la quantité de données qui peuvent être transférées ou traitées dans un certain laps de temps. Lorsqu'un fichier multimédia a un débit binaire élevé, il a généralement une meilleure qualité (mais entraîne également une taille de fichier plus importante).

Vous pouvez contrôler le débit binaire d'un fichier vidéo en utilisant l'option -b:v. La commande ci-dessous définit le débit binaire vidéo à 64 kbit/s :

```
ffmpeg -i input.mov -b:v 64k output.mp4
```

For audio files, use the -b:a option:

```
ffmpeg -i input.mp4 -b:a 224k output.mp3
```

## 4 Conclusion

Dans ce chapitre, le principe de fonctionnement du ffmpeg, son installation et son utilisation ont été présentés. Néanmoins, la découverte de toute la puissance de cet outil se fera en l'utilisant de manière plus approfondie et fréquente.

Choisir le bon format de fichier multimédia est important pour garantir que le contenu puisse être lu sur les plates-formes et les appareils cibles. Il est important de garder à l'esprit que chaque format a ses avantages et ses inconvénients. Nous avons utilisé cette bibliothèque dans le cryptage des vidéos en utilisons l'algorithme AES.

# CHAPITTRE 05

# IMPLEMENTATION DE L'ALGORITHME AES

# 1 Introduction

Nous avons présenté dans ce chapitre, les résultats obtenus de l'implémentation de différents algorithmes de chiffrement Appliqués aux vidéos.

On a choisi quatre algorithmes de chiffrement : ou-exclusif, mask, blowfish et l'algorithme AES.

Dans notre étude on a effectué une étude comparative entre les différents algorithmes selon deux critères importants : c'est la vitesse de chiffrement de chaque algorithme (le temps de chiffrement) et le deuxième critère c'est la qualité visuelle de l'opération de chiffrement appliqué sur les vidéos.

# 2 Les Algorithmes comparés

Cette section a l'intention de donner l'arrière-plan nécessaire sur les différents algorithmes comparés.

## 2.1 AES

Est un algorithme de chiffrement symétrique largement utilisé qui assure la confidentialité et l'intégrité des données. Il a été sélectionné par l'Institut National des Normes et de la Technologie en 2001 comme successeur du Standard de chiffrement des données (DES). L'AES fonctionne sur des blocs de données de taille fixe et prend en charge des clés de 128, 192 et 256 bits.

## 2.2 BLOWFISH

Est un algorithme de chiffrement symétrique par blocs ,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.

## 2.3 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.

### 2.4 MASK

Est un algorithme de chiffrement symétrique qui utilise une clé secrète aléatoire de même longueur que le texte en clair. Il fonctionne en effectuant un OU exclusif (XOR) entre chaque bit du texte en clair et de la clé pour produire le texte chiffré.

## 3 Présentation de l'Environnement et du Matériel et Outils Utilisés

Nous avons développé notre application sur un environnement Linux en utilisant la distribution Ubuntu 22.04. Pour l'implémentation, nous avons utilisé le compilateur GCC, reconnu pour sa puissance et sa performance en langage C.

Tout le code source de notre application est écrit en C. De plus, nous avons intégré les bibliothèques :

FFmpeg, qui offre une riche collection de fonctions pour la manipulation et le traitement de données multimédias, incluant les images, les vidéos et le son.

### 3.1 GTK

#### 3.1.1 Définition

Est un ensemble de bibliothèques logicielles, c'est-à-dire un ensemble de fonctions permettant de réaliser des interfaces graphiques. Cette bibliothèque a été développée originellement pour les besoins du logiciel de traitement d'images GIMP. GTK+ est maintenant utilisé dans de nombreux projets, dont les environnements de bureau GNOME, Xfce, Lxde et ROX.

#### 3.1.2 l'iInstallation

GTK+-3.0 (version )

sudo apt-get install libgtk-3-dev

### 3.2 GStreamer

#### 3.2.1 Definition

GStreamer est un framework multimédia basé sur des pipelines de segmentation écrits en langage de programmation C utilisant le type de

système basé sur GObject. GStreamer permet au programmeur de créer divers composants qui gèrent les médias, notamment la lecture audio, la lecture vidéo, l'enregistrement, le streaming et l'édition. La conception tubulaire de GStreamer constitue la base de la création de nombreux types d'applications multimédias telles que des éditeurs vidéo, des systèmes de streaming multimédia et des lecteurs multimédias.

### 3.2.2 Installation

sudo apt-get update

sudo apt-get install gstreamer1.0-libav

## 3.3 OpenSSL

### 3.3.1 Définition

est une boîte à outils de chiffrement comportant deux bibliothèques, libcrypto et libssl, fournissant respectivement une implémentation des algorithmes cryptographiques et du protocole de communication SSL/TLS, ainsi qu'une interface en ligne de commande, openssl.

### 3.3.2 Installation

sudo apt-get install openssl

## 3.4 FFMPEG

### 3.4.1 Définition

Fast Forward MPEG (FFmpeg) est une application informatique gratuite qui permet le traitement des fichiers audio et vidéo. Le traitement est basé sur des lignes de commande et est particulièrement apprécié pour sa rapidité lors du codage et décodage des fichiers audio et vidéo.

### 3.4.2 Installation

$ sudo add-apt-repository ppa: mc3man / trusty-media

$ sudo apt-get update

$ sudo apt-get install ffmpeg

$ ffmpeg –version

## 4 Les paramètres du système

| | |
|---|---|
| **Processeur** | **Intel(R) Core(TM) i5-2430M CPU @ 2.40GHz**<br>**2.40 GHz** |
| **Fabricant** | Intel |
| **Vitesse** | 2.40 GHz |
| **Nombre de cœurs** | 1 |
| **Mémoire** | 6 GB |
| **Système d'exploitation** | Windows 10 Professionnel |
| **Taille** | 64 Bit |

Tableau 5-1 : Les paramètres du système

## 5 Etude comparative entre les différents algorithmes de cryptage

- **Vidéo 01**

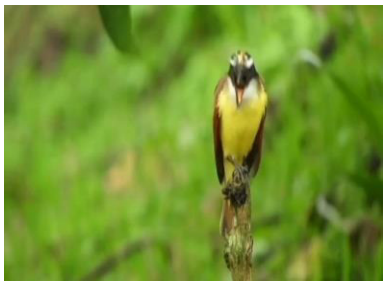
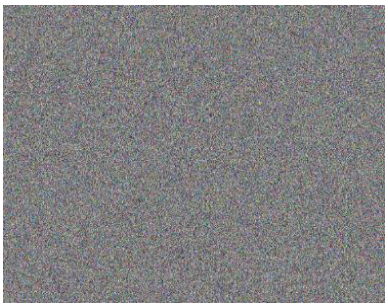
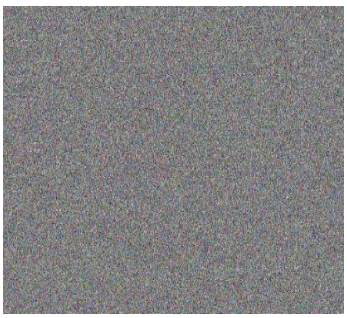
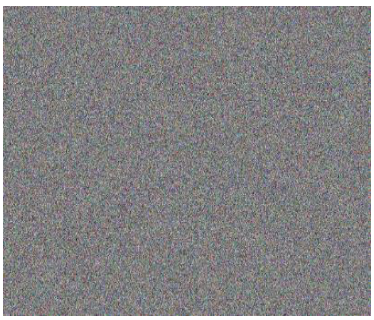
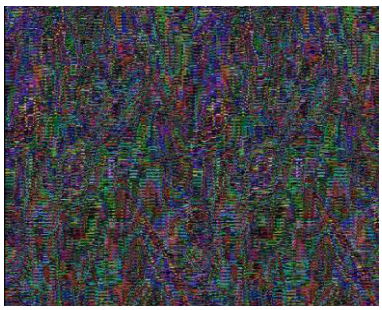
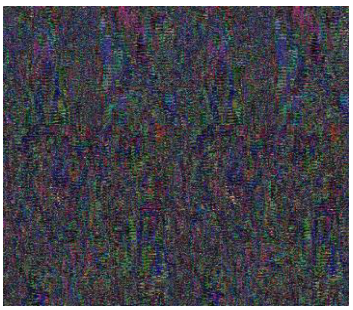
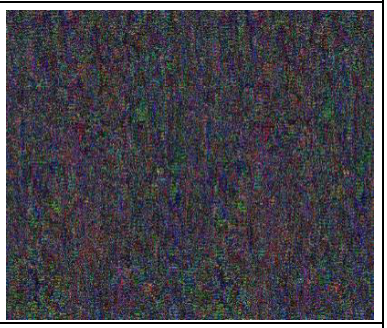
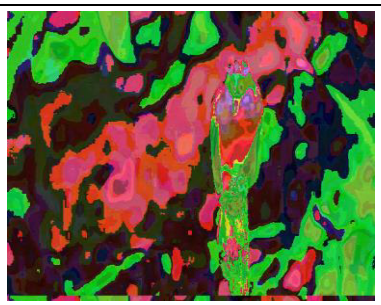
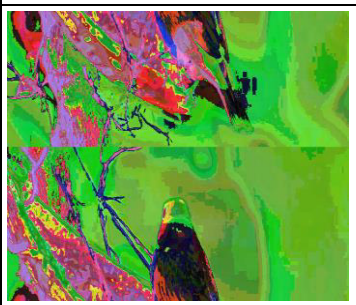
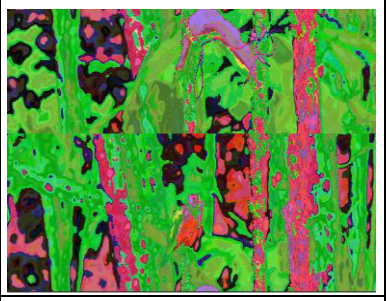
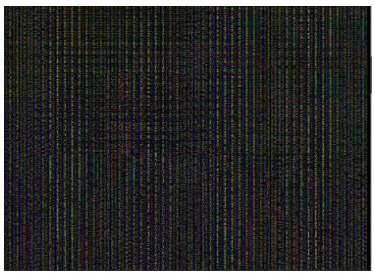
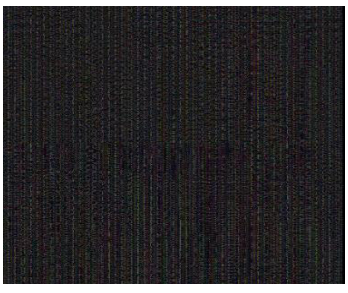
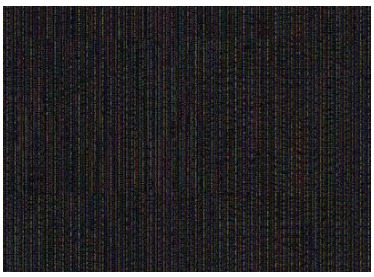
| Temps Vidéos | Type De Vidéos | Le Début | Le milieu | La FIN |
|---|---|---|---|---|
| Original | |  |  |  |
| Chiffrement Vidéo :01 | Résultat AES |  |  |  |
| | Résultat BLOWFISH |  |  |  |
| | Résultat XOR |  |  |  |
| | Résultat MASK |  |  |  |

Tableau 5-2 : Résultats Comparatifs du Chiffrement par AES,BLOWFISH ,XOR et MASK- Vidéo 01-

- **Vidéo 02**

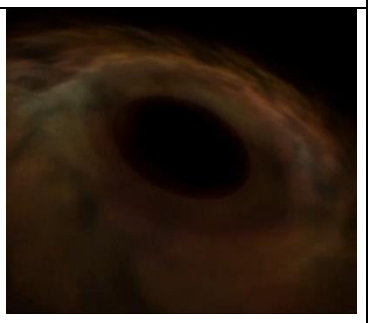
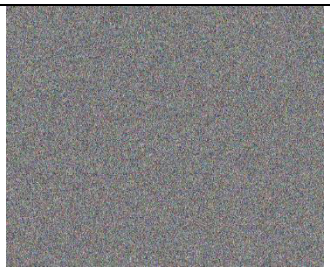
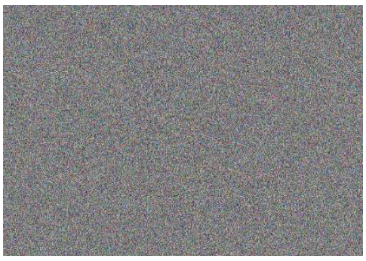
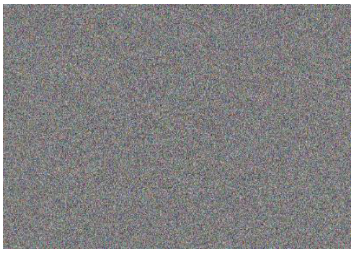
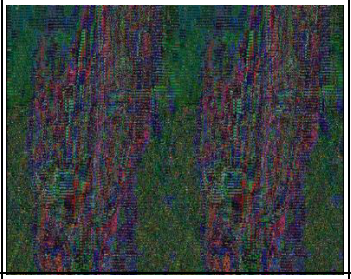
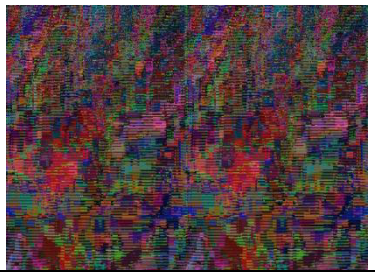
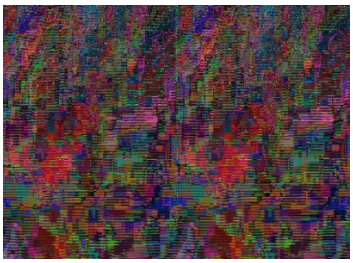
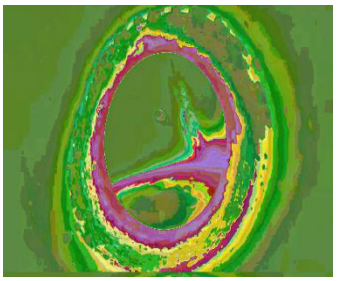
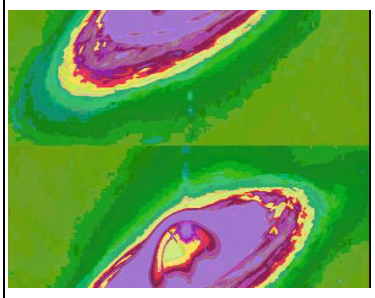
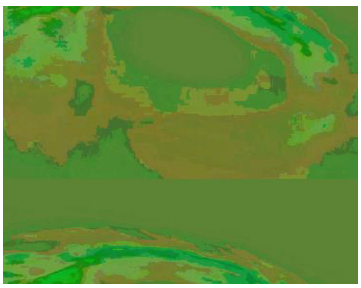
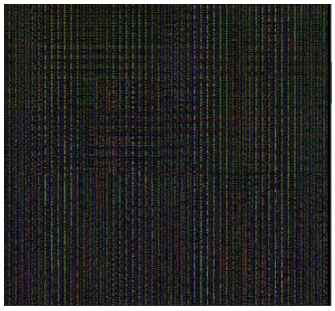
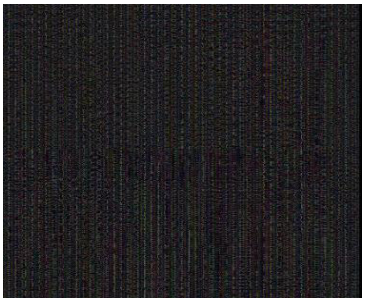
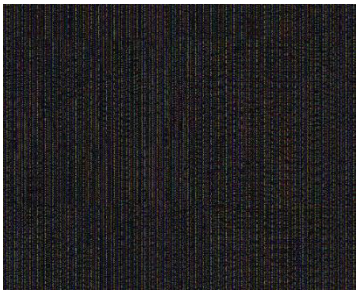
| Temps Vidéos | Type De Vidéos | Le Début | Le milieu | La FIN |
|---|---|---|---|---|
| Original | |  |  |  |
| Chiffrement Vidéo :02 | Résultat AES |  |  |  |
| | Résultat BLOWFISH |  |  |  |
| | Résultat XOR |  |  |  |
| | Résultat MASK |  |  |  |

Tableau 5-3 : Résultats Comparatifs du Chiffrement par AES,BLOWFISH ,XOR et MASK- Vidéo 02-

- **Vidéo 03**

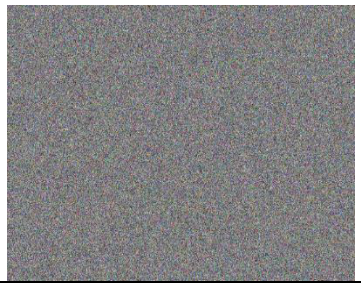
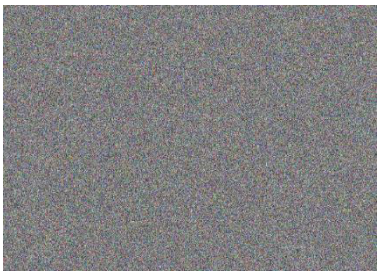
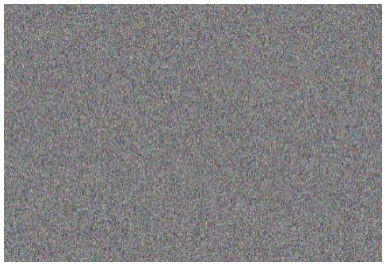
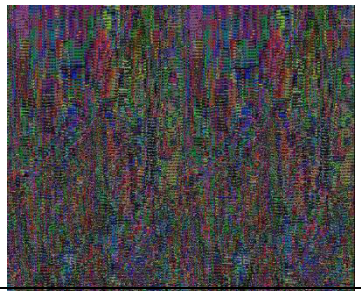
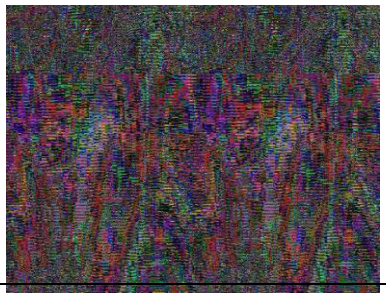
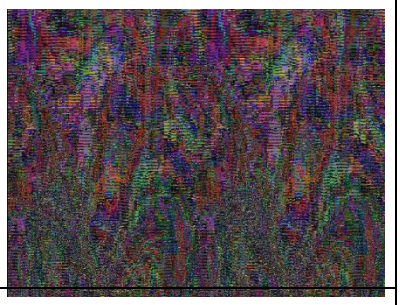
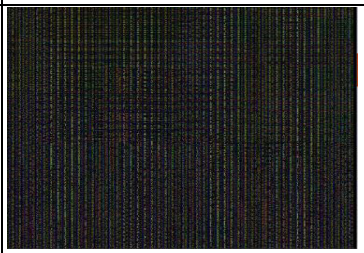
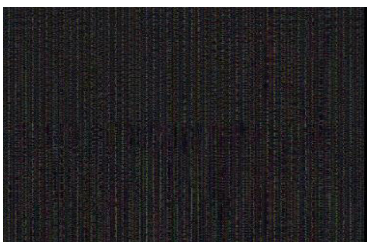
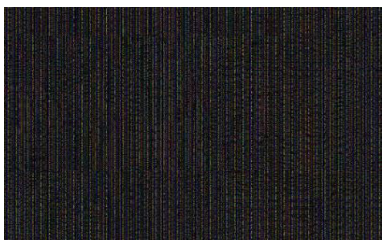
| Temps Vidéos | Type De Vidéos | Le Début | Le milieu | La FIN |
|---|---|---|---|---|
| Original | |  |  |  |
| Chiffrement Vidéo :03 | Résultat AES |  |  |  |
| | Résultat BLOWFISH |  |  |  |
| | Résultat XOR |  |  |  |
| | Résultat MASK |  |  |  |

Tableau 5-4 : Résultats Comparatifs du Chiffrement par AES,BLOWFISH ,XOR et MASK- Vidéo 03-

## 5.1 Critère sur la vitesse et le temps de chiffrement de chaque algorithme

- **Temps de chiffrement :**

| | Le temps de chiffrement Aes | Le temps de chiffrement Mask | Le temps de chiffrement Blowfish | Le temps de chiffrement Xor |
|---|---|---|---|---|
| **Vidéo 01** | 2 h 28 min | 3 min | 28 min | 2 min |
| **Vidéo 02** | 25 min | 1 min 25 s | 15 min | 1 min |
| **Vidéo 03** | 16 min | 50 s | 9 min | 37 s |

Tableau 5-5 : Temps de cryptage

- **Histogramme de temps de chiffrement**

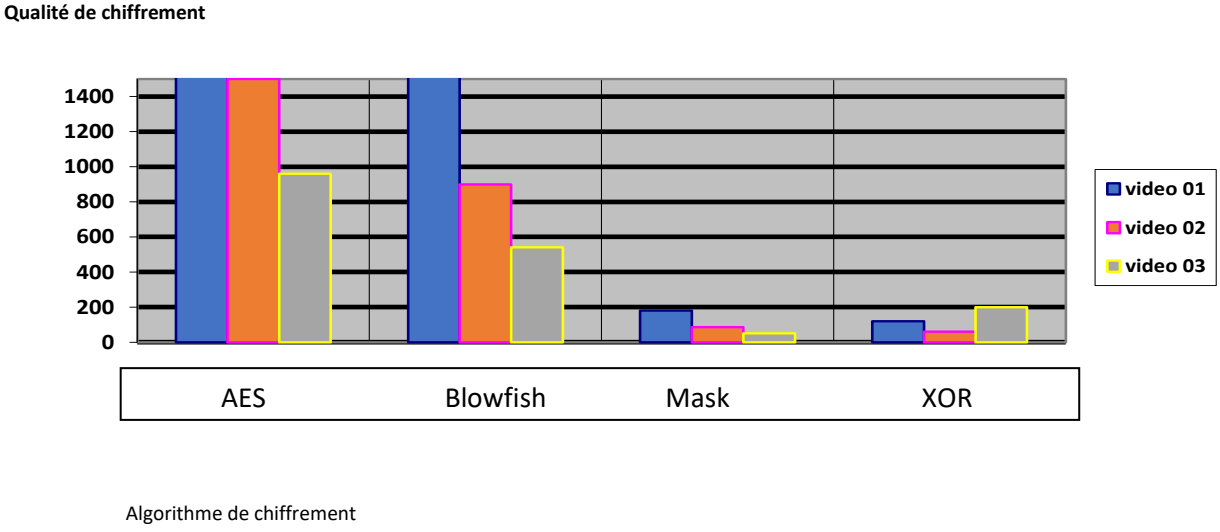



Figure 5.1 : Comparaison temps de chiffrement des 03 Vidéos

**Remarque**

- D'après les résultats obtenus, nous remarquons que les deux algorithmes ou-exclusif et MASK sont très rapide dans la vitesse de l'opération de chiffrement des vidéos par rapport aux deux algorithmes BLOWFISH et AES.
- Les deux algorithmes AES et BLOWFISH prennent beaucoup temps dans l'opération de chiffrement.

## 5.2 Critère sur la qualité de l'opération de chiffrement de chaque algorithme

- Entre 80 et 100 meilleures qualités
- Entre 40 et 80 bonnes qualités
- Entre20 et 40 moyennes qualité
- Entre 10 et 20 mauvaises qualités

- **Qualité de chiffrement**

| | Ou exclusif | MASK | BLOWFISH | AES |
|---|---|---|---|---|
| **Video1** | 10 | 70 | 50 | 70 |
| **Vidéo 02** | 20 | 70 | 70 | 80 |
| **Vidéo 03** | 30 | 80 | 70 | 90 |

Tableau 5-6 : la qualité de chiffrement des vidéos

- **Histogramme de qualité de l'opération de chiffrement**

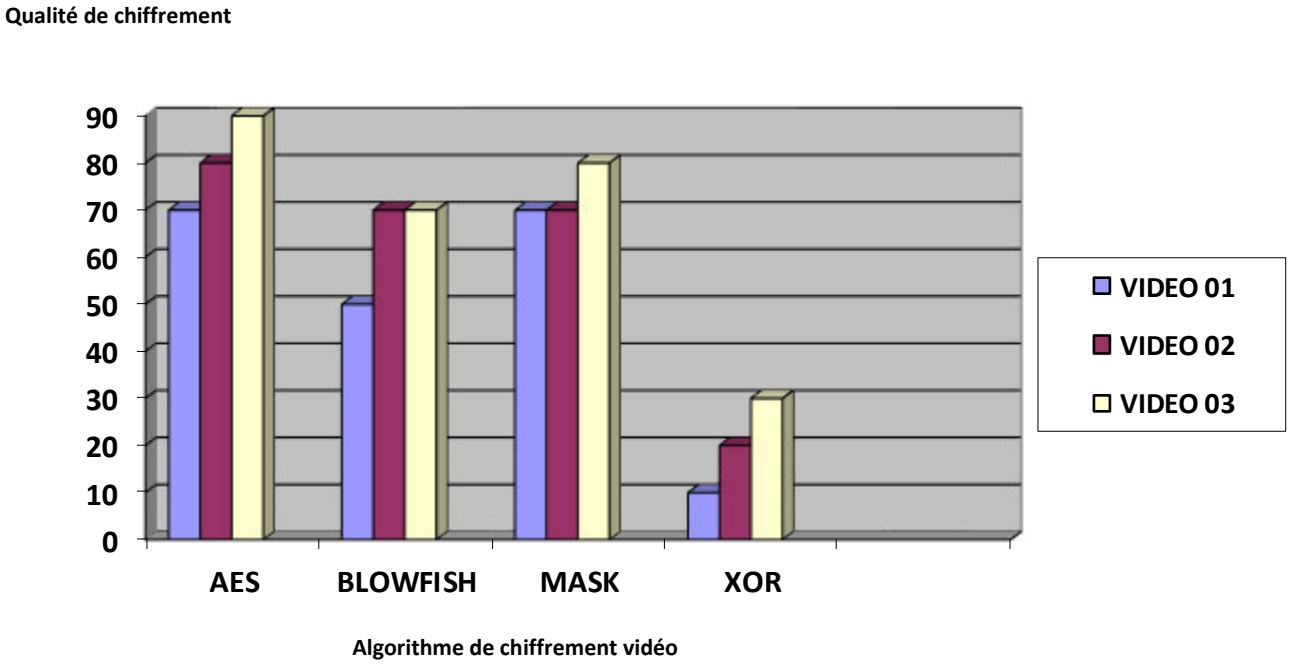



Figure 5.2: la qualité de chiffrement des vidéos

- **Remarque**

Nous remarquons que l'algorithme AES offre une meilleure qualité de chiffrement par rapport aux autres algorithmes.

L'algorithme Ou-exclusif produit une qualité de chiffrement inférieure en comparaison avec les autres algorithmes.

Le MASK présente une bonne qualité de chiffrement, supérieure à celle de l'algorithme BLOWFISH.

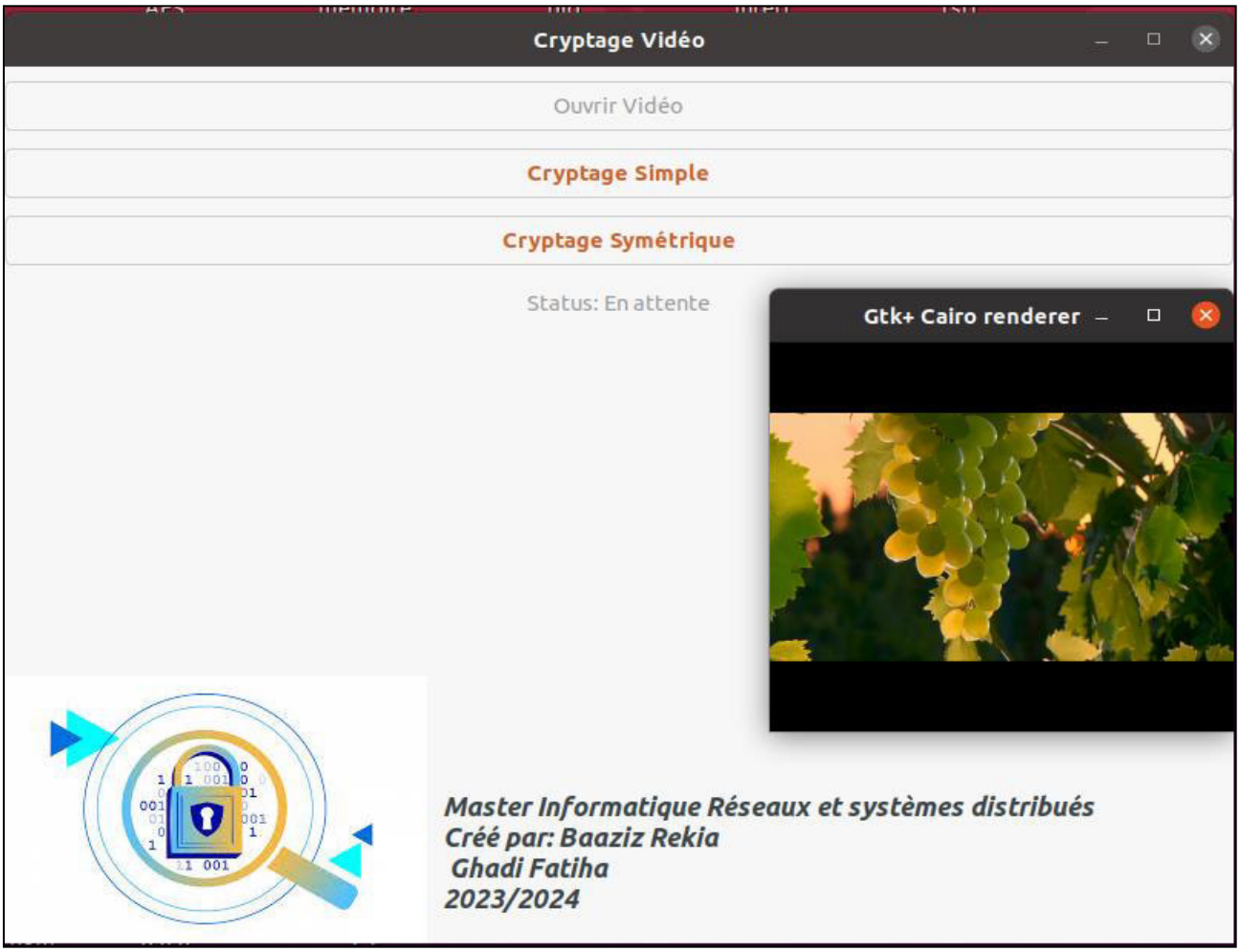
## 6 F Présentation de l'application

pour exécuter l'interface d'application il faut accéder au répertoire qui contient le code source en c et compiler la commande suivante :

```
user00@user00-Inspiron-N5050:~$ cd Bureau
user00@user00-Inspiron-N5050:~/Bureau$ cd interf
user00@user00-Inspiron-N5050:~/Bureau/interf$ gcc -o P1 v.c `pkg-config --cflags
 --libs gtk+-3.0 openssl gstreamer-1.0`
user00@user00-Inspiron-N5050:~/Bureau/interf$ ./P1
user00@user00-Inspiron-N5050:~/Bureau/interf$
```

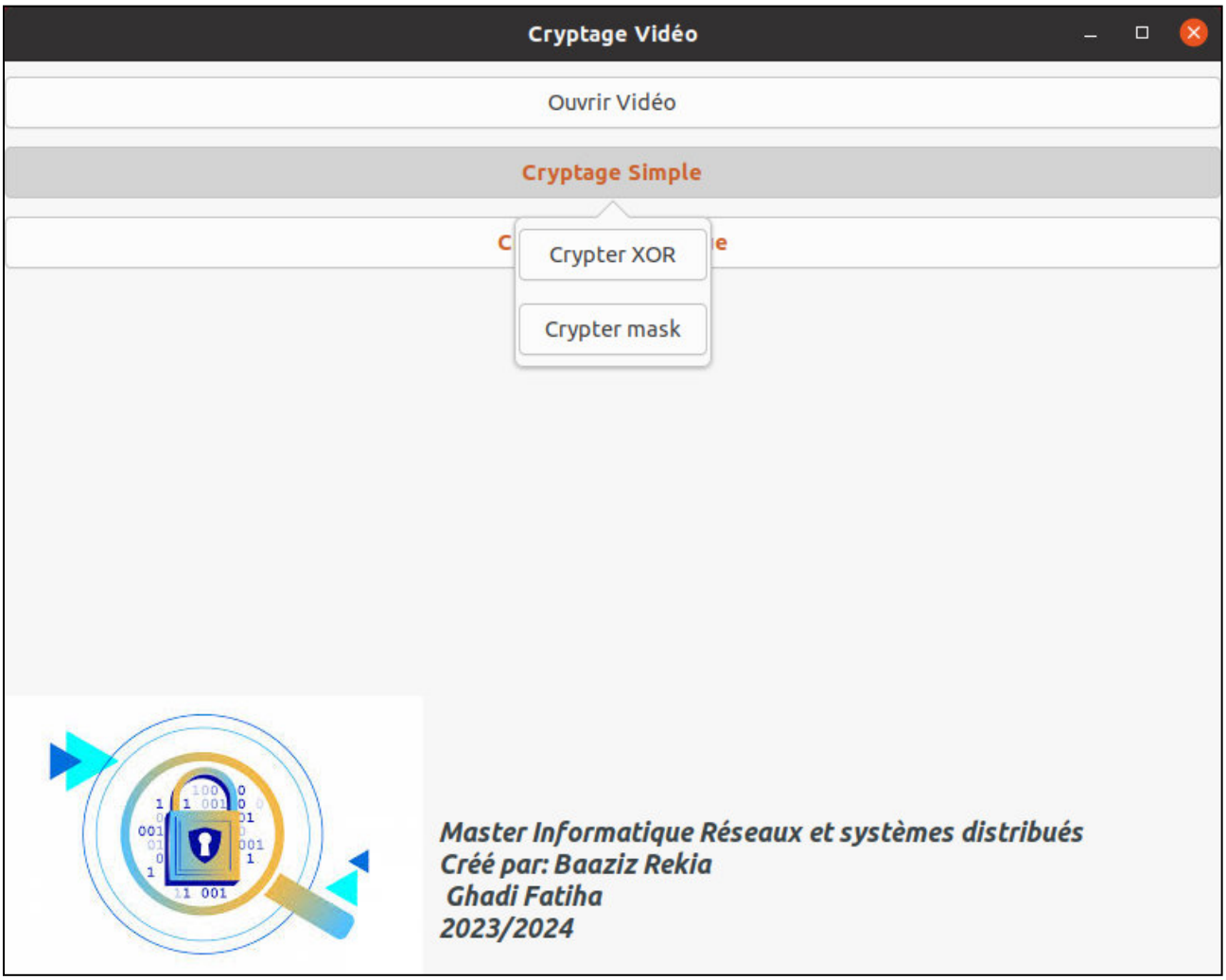
L'interface comporte 03 boutons principaux (Ouvrir Vidéo, Cryptage Simple, Cryptage Symétrique) pour afficher les différentes images cryptées.

Au démarrage de l'application la fenêtre ci-dessous s'ouvre



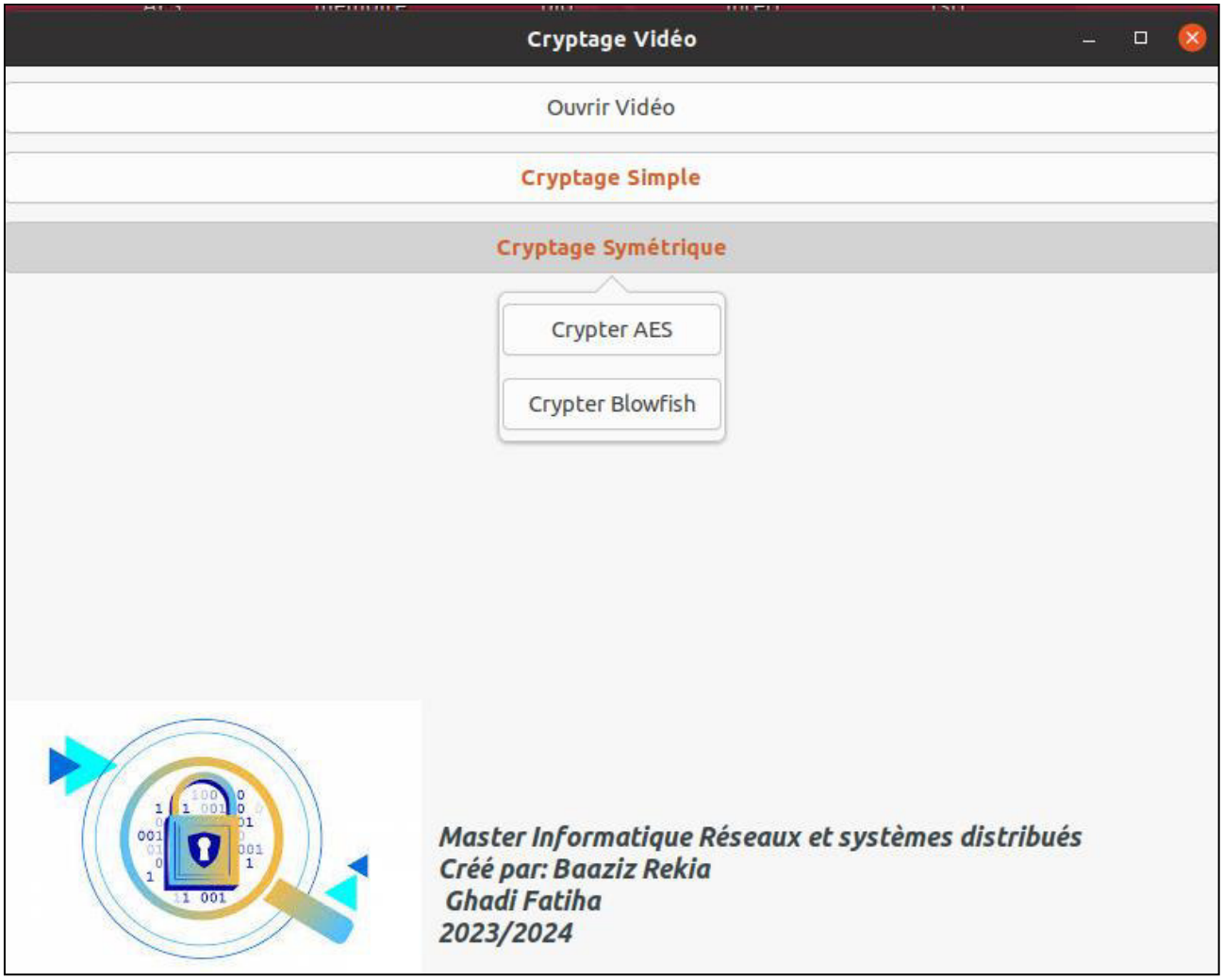

| | |
|---|---|
| 01 | le bouton Ouvrir Vidéo permet d'afficher la vidéo souhaite à crypter avant et après le cryptage. |

Ainsi, lorsque nous appuyons sur le bouton Cryptage Simple et choisi Crypter XOR ou Crypter mask, la vidéo sera cryptée avec le ouexclusif ou mask, et lorsque le cryptage est terminé, elle sera enregistrée dans le même répertoire de code source .




| | |
|---|---|
| 02 | le bouton Cryptage Simple contient deux boutons secondaire ( Crypter XOR , Crypter Mask) |

Ainsi, lorsque nous appuyons sur le bouton Cryptage Symétrique et choisi Crypter AES ou bien Crypter Blowfish, la vidéo sera cryptée avec AES ou blowfish, et lorsque le cryptage est terminé, elle sera enregistrée dans le même répertoire de code source .




| | |
|---|---|
| 03 | le bouton Cryptage Symétrique contient deux button secondaire ( Crypter AES, Crypter Blowfish). |

## 7 Conclusion

D'après les différents tests, le chiffrement ou-exclusif s'est révélé constamment plus rapide, bien que produisant une qualité visuelle mauvaise, notamment lors du traitement de vidéos à fort similarité.

En revanche, le chiffrement mask offre une qualité visuelle élevée mais souffre d'une vitesse d'exécution très lente. Étant donné que la rapidité est cruciale, le chiffrement ou-exclusif est souvent privilégié dans les applications pratiques, malgré sa qualité visuelle inférieure.

AES est le meilleur choix pour des applications nécessitant un chiffrement fort, rapide et standardisé pour protéger des données sensibles.

BLOWFICH est plus adapté pour des scénarios où il est crucial de cacher la présence d'information plutôt que de la chiffrer, comme dans la stéganographie ou le tatouage numérique.

En résumé, pour la sécurité des données en général, AES est largement préféré en raison de sa robustesse et de ses performances. Pour des applications spécifiques de masquage d'information, BLOWFICH offre des avantages uniques.

Pour conclure, l'algorithme AES est idéal pour ses garanties de sécurité et ses bonnes performances globales en matière de chiffrement. Cependant, il présente des limitations en termes de vitesse lorsqu'il s'agit de chiffrer des vidéos. L'algorithme AES devient optimal lorsqu'il est utilisé pour le cryptage sélectif des vidéos. Le nombre de frames (images) dans une vidéo affecte à la fois la qualité et la vitesse du processus de chiffrement.

## Conclusion générale

De nos jours, de plus en plus les vidéos numériques sont transférées ou stockées sur les réseaux informatiques. Avec le temps, la confidentialité de vidéo numérique est devenue indispensable.

Dans notre étude, nous avons examiné le cryptage des vidéos à l'aide de divers algorithmes de chiffrement. Cela nous a permis de réaliser une étude comparative des performances de chaque type d'algorithme en fonction du temps et de la qualité de l'opération de chiffrement.

Les résultats obtenus montrent que les algorithmes AES et MASK offrent une meilleure qualité de cryptage des vidéos comparativement aux algorithmes BLOWFISH et XOR mais soufre de la vitesse et le temps de chiffrement. Cependant, nous avons également constaté que lorsque le nombre de frames augmente, la vitesse du chiffrement des vidéos augments et quand le facteur de similarité est faible la qualité de chiffrement est meilleure.

Pour les futures recherches, nous prévoyons de nous orienter vers le cryptage sélectif des vidéos dans le cadre d'un projet de fin d'études de master, afin de comparer ces résultats avec ceux obtenus dans cette étude.

## Bibliographie

**[1]**. Kebir Bahia, Développement d'une application pour l'échange des messages sécurisés, Université Abou Bakr Belkaid.Tlemcen, 2014/2015.

**[2]**. Daniel LAMAS, La cryptographie,École de Gestion de Genève (HEG-GE), 05/06/2015.

**[4]**. National Institute of Standards and Technology: Data Encryption Standard (DES). http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf, (2001).

**[7]**. Simon Guillem-Lessard « Tutoriel de la cryptographie », Projet de fin d'étude 2001-2002 Département des mathématiques et de l'informatique Université du Québec à Trois-Rivières.

**[10]**. Chenene Boubakeur, Chiffrement des vidéos numériques, Université Mohamed Boudiaf.M'sila, 2018 /2019.

**[11]**. R. Westwater, B. Furht, Real-Time Video Compression Techniques and Algorithms, 1997.

**[12]**. I. E. G. Richardson, H.264 and MPEG-4 Video Compression, Aberdeen, UK: The Robert Gordon University, 2003.

**[14]**. F. D. Rango, Digital Vidéo,InTech, 2010.

**[15]**. Y. Tan, D. D. Saur, S. R. Kulkami, P. J. Ramadge, Rapid estimation of camera motion from compressed video with application to video annotation, IEEE Transactions on Circuits and Systems for Video Technology, n°110, p. 1333146, 2000.

**[16]**. Betka Ibrahim, Etude et Implémentation Pipeline sur FPGA de L'algorithme de Chiffrement AES, Université Mohamed Boudiaf.M'sila, 2017 /2018.

**[17]**. O. Azzouzi, Système embarqué flexible pour un chiffrement hybride symétrique / asymétrique, Mémoire de Magistère en Informatique, Ecole Nationale Supérieure, Oued Smar, Algérie, 2014.

**[18]**. Joan Daemen and Vincent Rijmen, AES submission document on Rijndael, June 1998.

**[19]**. The Design of RijndaelJ Daemen, V Rijmen – 2001/ http://jda.noekeon.org/JDA_VRI_Rijndael_2002.pdf.

**[20]**. FIPS PUB 197, Advanced Encryption Standard (AES), National Institute of Standards and Technology, U.S. Department of Commerce, November 2001.http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf.

## Biblioweb

**[5]**. http://fr.wikipedia.org/.

**[6]**. http://sevat.fr/damien/M1S1/Theorie_Info/Rapport/.consulté le 11 03 2024.

**[3]**. https://academy.binance.com/fr/articles/symmetric-vs-asymmetric-encryption .consulté le 11 03 2024.

**[8]**. https://elearning-facsci.univ-annaba.dz/pluginfile.php/13925/mod_resource/content/0/CHAP2.4_%20Diffie%20et%20Hellman.pdf.consulté le 17 04 2024.

**[9]**. https://apprendre-le-cinema.fr/standards-de-codage-couleur-et-modes-de-balayage/.consulté le 22 04 2024.

**[13]**. wikiversity, https://fr.wikiversity.org/wiki/Formats_vidéo. Consulté le 07 05 2024.

**[21]**. https://fr.wikipedia.org/wiki/FFmpeg

**[22]**. https://fr.wikihow.com/installer-FFmpeg-sur-Windows

**[23]**. https://tel.archives-ouvertes.fr/tel-01943877/document

**[24]**. https://fr.wikipedia.org/wiki/Biblioth%C3%A8que_logicielle

**[25]**. https://www.projet-plume.org/fiche/ffmpeg

**[26]**. https://debian-facile.org/doc:media:ffmpeg

**[27]**. https://wiki.debian.org/fr/ffmpeg

**[28]**. https://debian-facile.org/doc:media:ffmpeg

**[29]**. https://ffmpeg.org/ffmpeg.html

**[30]**. https://doc.ubuntu-fr.org/ffmpeg

**[31]**. https://www.vcodex.com/news/h264-quantization-parameter/

**Résumé**

Dans le contexte actuel, l'augmentation exponentielle des informations transitant par les réseaux internationaux impose une nécessité accrue de cryptage pour garantir la confidentialité et la sécurité des données. C'est peut-être l'un des défis les plus cruciaux d'Internet. En termes simples, le cryptage empêche l'interception, la lecture ou la modification des messages en clair, ainsi que la création de messages falsifiés.
La sécurisation des données visuelles, telles que les vidéos, lors de leur transfert via des systèmes de communication comme Internet, est essentielle pour empêcher les modifications non autorisées de leur contenu. C'est dans ce contexte que s'inscrit notre travail, qui vise à implémenter divers algorithmes de cryptographie pour chiffrer des vidéos. Notre étude se conclut par une analyse comparative du niveau et de la qualité de cryptage des données visuelles pour chaque type d'algorithme de chiffrement.

**Mots clés** : cryptage symétrique, cryptage asymétrique, algorithme AES, algorithme XOR, algorithme MASK, algorithme BLOWFISH.

## Abstract

In the current context, the exponential increase in information transiting through international networks imposes an increased need for encryption to guarantee data confidentiality and security. This is one of the most crucial challenges of the Internet. Simply put, encryption prevents the interception, reading, or modification of clear messages, as well as the creation of falsified messages.
Securing visual data, such as videos, when transferred through communication systems such as the Internet, is essential to preventing unauthorized modification of their content. This is the context of our work, which aims to implement various cryptographic algorithms to encrypt videos. Our study concludes with a comparative analysis of the level and quality of encryption of visual data for each type of encrypting algorithm.

**Keywords** : symmetric encryption, asymmetric encryption, AES algorithm, XOR algorithm, MASK algorithm, BLOWFISH algorithm.

## ملخص

وفي السياق الحالي، فإن الزيادة الهائلة في المعلومات التي تنتقل عبر الشبكات الدولية تفرض حاجة متزايدة إلى التشفير لضمان سرية البيانات وأمنها. وهذا أحد أهم التحديات التي تواجه الإنترنت. ببساطة، يمنع التشفير اعتراض الرسائل الواضحة أو قراءتها .أو تعديلها، فضلاً عن إنشاء رسائل مزورة

يعد تأمين البيانات المرئية، مثل مقاطع الفيديو، عند نقلها عبر أنظمة الاتصالات مثل الإنترنت، أمرًا ضروريًا لمنع التعديل غير المصرح به لمحتواها. هذا هو سياق عملنا الذي يهدف إلى تنفيذ خوارزميات التشفير المختلفة لتشفير مقاطع الفيديو. وتختتم .دراستنا بتحليل مقارن لمستوى وجودة تشفير البيانات المرئية لكل نوع من خوارزميات التشفير

**الكلمات المفتاحية** : التشفير المتماثل، التشفير غير المتماثل، BLOWFISH، خوارزمية MASK، خوارزمية XOR، خوارزمية AES