*A mon père et ma mère,*

*A mes frères Faouzi, Issam et Omar,*

*A mes amis Issam, Hichem, Hafedh et Taher*

*A 62635,*

*A mes yeux,*

*A tous ceux que j'aime et qui m'aiment,*

*Je dédie ce travail.*

# *Remerciements*

*Je tiens à remercier tout d'abord mes deux encadrants, Monsieur Toufik Ahmed et Madame Nawel Zangar pour leurs soutiens et leurs conseils durant ces quatre mois de stage de projet de fin d'études.*

*J'aimerais aussi remercier mon ami et frère Mossaab Hariz, de l'INT d'Evry - France, pour ses conseils importants tout au long de ce stage.*

*Je remercie aussi Mr. Ismail Djama pour son soutien et sa collaboration le long de ce stage.*

*Pour terminer, j'aimerais remercier les membres de jury pour leur présences, leurs implications ainsi que pour le temps consacré à ce rapport de PFE.*

# Table des matières

# Liste des figures

# Liste des tableaux

# I. Introduction Générale

## 1. Contexte et objectif du Projet

Ces dernières années ont vu un développement sans précédent des communications sans fils, ouvrant la porte à plusieurs applications concrètes. Par ailleurs, le protocole IP est devenu, de fait, le standard pour la plupart des réseaux sans fils existants et en cours de développement (par exemple : WiFi, WiMax, UMTS, Bluetooth, et autres).

Un intérêt industriel particulier est particulièrement accordé aux communications dans les réseaux locaux sans fils (WiFi networks –*WLAN*) vu leur grande disponibilité et large adoption pour l'usage particulier. Aujourd'hui, pratiquement tous les services IP (Web, VoIP, streaming vidéo, etc.) peuvent être déployés pour des terminaux sans fil.

Par ailleurs, avec le développement de multiples outils *Open Source* pour le streaming vidéo (MPEG4IP, VLC, VAT, Darwin, etc.), la conception de system (client/serveur) de streaming vidéo est devenu assez facile. Par exemple, VLC (VideoLAN Client, http://www.videolan.org/vlc/) offre le code source d'un système très complet (client/serveur) pour la capture, transcodage, adaptation, transmission, réception et affichage de flux vidéo. Ce projet a été initié et continue à être maintenu par les élèves ingénieurs de l'Ecole Centrale Paris. Les librairies VLC offrent un environnement idéal pour l'intégration de mécanismes pour l'automatisation de l'accès aux sessions vidéo à partir des clients où encore l'adaptation de la qualité de service (QoS) des flux vidéo. Ces mécanismes peuvent par exemple offrir à l'utilisateur les choix de sessions vidéo actives

à partir d'une interface graphique. Les informations sur les format et résolution de la session vidéo peuvent être aussi rafraîchi dynamiquement.

Les réseaux de nouvelle génération sont supposés être très complexes, interconnectant différentes technologies et architectures de réseaux fixes et mobiles à travers plusieurs plates-formes matérielles et logicielles et offrant une multitude de services (données, téléphonie, vidéo, etc.). La garantie de la qualité de service (QoS) et de la sécurité des données de bout en bout sur ces réseaux hétérogènes, pour ces types de service, est actuellement un axe de recherche important. Quelques solutions ont été proposées mais qui restent partielles puisqu'elles traitent des cas bien particuliers pour chaque type de réseaux et services et n'intègrent qu'une composante de la gestion de réseau, à savoir la QoS ou la sécurité.

L'objectif principal de ce projet est de proposer, développer et évaluer une architecture ouverte, complète et évolutive, intégrant à la fois la QoS et la sécurité, pour des utilisateurs fixes et mobiles demandant des niveaux de service spécifiques sur les réseaux sans-fil (WiFi). Le scénario de déploiement ciblé est la vidéosurveillance accessible (et contrôlable) via des terminaux WiFi.

Il s'agit de couvrir l'aspect d'adaptation et de contrôle de la QoS et de la sécurité dans la chaîne de distribution des services, afin de supporter une QoS et une sécurité de bout en bout. Le but est double, d'une part, définir des stratégies de QoS et de sécurité sur chaque entité individuelle de la chaîne, et d'autre part, harmoniser leurs fonctionnalités sur les réseaux hétérogènes.

Offrir la sécurité de la communication et la qualité de service exigée par l'application dans les réseaux de nouvelle génération, font parti des défis majeurs que la gestion de réseaux et services doit relever pour permettre à l'utilisateur d'être connecté en continu de partout et à haut débit.

## 2. Cadre du projet et organisme d'accueil

Notre projet s'inscrit dans le cadre du projet de la région d'Aquitaine dont l'objectif principal est de proposer, développer et évaluer une architecture ouverte, complète et évolutive, intégrant à la fois la QoS et la sécurité, pour des utilisateurs fixes et mobiles demandant des niveaux de service spécifiques sur les réseaux de nouvelle génération.

Nous effectuons ce stage au sein du Laboratoire Bordelais de Recherche en Informatique, le LaBRI[1] qui est une unité de recherche associée au CNRS, à l'Université Bordeaux 1[2] et à l'ENSEIRB[3], il est partenaire de l'INRIA[4] à travers l'Unité de Recherche Futurs. Les missions du LaBRI s'articulent autour de trois axes principaux : recherche (théorique, appliquée), transfert de technologie et formation. Le laboratoire s'est construit

[1] http://www.labri.fr
[2] http://www.u-bordeaux1.fr
[3] http://www.enseirb.fr
[4] http://www.inria.fr

initialement autour de recherches en informatique théorique puis a étendu le spectre de ses activités vers des domaines plus appliqués. Parmi ces domaines se distingue celui des réseaux et de la sécurité, un thème qui relève des considérations de l'équipe COMET (COntext-aware Management and nETworking) à laquelle nous appartenons et qui travaille depuis plusieurs années sur la négociation, le contrôle et la configuration dynamique de services dans les réseaux IP offrant des garanties de QoS.

# II. LA DIFFUSION MULTIMEDIA OU STREAMING

## 3. La Diffusion Multimédia

### 3.1 Définition

Le streaming est une technique de transfert de données sous forme d'un flux (*stream d'où l'appellation streaming*) régulier et continu. Il permet donc de diffuser des contenus multimédia, à la demande et en temps réel, et ce sans solliciter le disque dur de l'utilisateur.

Il existe deux grandes familles de protocoles pour faire du streaming :

- *Streaming sur TCP* : en utilisant le protocole HTTP (Hypertext Transfer Protocol) qui est un protocole de transfert de fichier. On peut néanmoins parler de streaming dans la mesure où les vidéos peuvent être affichées au fur et à mesure du téléchargement. L'utilisation de TCP comme protocole de transport présente plusieurs inconvénients :
    - Avec une entête de 20 octets (8 pour l'entête UDP), TCP entraîne une augmentation de l'overhead de transmission pour des flux bas débit.
    - TCP ne supporte pas le multicast.
    - TCP implémente les mécanismes de contrôle d'erreurs qui occasionnent une perte de débit et qui ne sont pas nécessaires aux applications de streaming. En effet, l'audio et la vidéo ont un mécanisme de contrôle naturel d'erreurs dans le système audiovisuel humain.

- *Streaming sur UDP* : en utilisant le protocole RTP (Real Time Protocol) qui permet de faire à proprement parler du streaming, c'est à dire de la diffusion de contenu en temps réel. Ce protocole sera présenté en détail dans le paragraphe 2.1.

## 3.2 Domaines d'application

Les domaines d'application du streaming sont nombreux, on peut en citer :

- diffusion de clips vidéo, bandes-annonces de films ou de musiques
- éducation : e-learning
- communication commerciale : vente - publicité – promotion
- marketing direct : vidéo par e-mails
- communication publicitaire : diversification des bannières en Rich Media
- communication événementielle : conférences, salons, défilés
- chaînes de TV Interactives, télévisions d'entreprises
- diffusion de musique, retransmissions de soirées musicales live
- émissions de radio interactives
- la vidéosurveillance et la vidéoconférence

## 3.3 Principe et mode de diffusion

### 3.3.1 Principe

Le fichier vidéo est envoyé par paquets vers l'utilisateur dont l'ordinateur décompresse et recompile.
Toutefois, la bande passante présente des fluctuations énormes ce qui constitue un handicap de taille pouvant affecter la qualité de la vidéo.
En effet, la qualité de la vidéo est déterminée par la taille de l'image, sa résolution et son rafraîchissement. La bande passante exigée en diffusion multimédia augmente proportionnellement à ces facteurs.

Pour pallier aux problèmes de fluctuation de bande passante, une mémoire tampon est créée par le lecteur de média.
Si celle-ci trouve qu'elle possède suffisamment d'informations, elle retransmet les images en vitesse constante. L'ordinateur client envoie en permanence des informations de contrôle pour informer le serveur de sa vitesse de transfert. Ce dernier est à même d'adapter le débit, la qualité peut en pâtir mais la vidéo n'est pas interrompue.
Tout se passe comme si on possédait virtuellement l'entièreté de la vidéo.

### 3.3.2 Mode de diffusion

La diffusion de vidéo peut être Unicast (un seul destinataire) ou Multicast (plusieurs destinataires) à partir du serveur. Dans une configuration Unicast, le serveur répète la transmission pour chaque destinataire final. Dans une configuration Multicast, le même signal est envoyé sur le réseau, en une seule transmission, à plusieurs destinataires, ou simplement, à un groupe d'utilisateurs.

#### 3.3.2.1 Unicast

Cette technique consiste à associer un flux à chaque utilisateur (Figure 1). Le serveur est évidemment plus sollicité mais cela permet une plus grande souplesse vis-à-vis des clients. En effet, ces derniers peuvent choisir le débit qui convient à leur infrastructure.

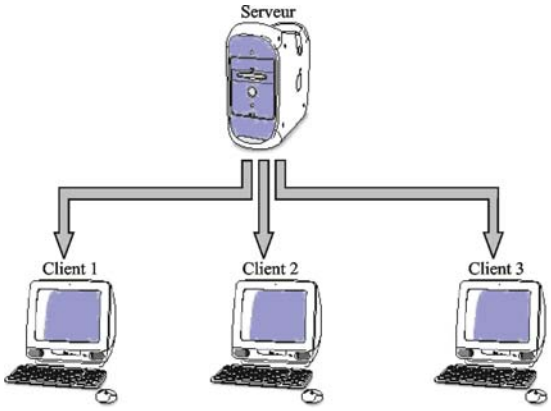


***Figure 1 : Schéma du système de streaming Unicast***

#### 3.3.2.2 Multicast [1]

Dans cette technique un seul flux est émis à partir du serveur (Figure 2). Ce flux est reçu par tous les clients. Cependant, les clients doivent s'abonner au groupe pour recevoir le flux. Cette technique est utilisée dans le streaming sur Internet mais pose quelques problèmes d'implémentation au niveau des routeurs (les routeurs doivent pouvoir gérer le multicast).

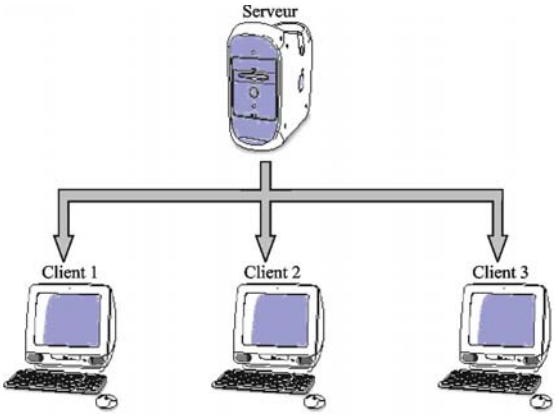


***Figure 2 : Schéma du système de Streaming Multicast***

Le premier qui a lancé les travaux sur l'IP Multicast est Steve Deering dans son PhD en 1988. La première utilisation de cette technique à grande échelle, eut lieu quant à elle pendant l'Audioconférence de la réunion de l'IETF en mars 1992 à San Diego.

La mise en œuvre du Multicast nécessite :

- La spécification et la gestion du groupe Multicast. Exemple de protocoles développés et implémentés : Management Protocol (IGMP) pour IPV4 et MLD pour IPV6
- La construction d'un arbre de routage. Exemple de protocoles de routage : DVMRP, PIM, CBT…

## 3.4 Les avantages du streaming vidéo

Contrairement au simple téléchargement d'une vidéo, utiliser un serveur de streaming ou de flux vidéo apporte de nombreux avantages.

- **Accès quasi-instantané** : La lecture de la vidéo commence quasiment immédiatement.
- **Durée illimitée** : Il n'y a plus aucune limite de taille de fichier puisqu'il est lu "au fur et à mesure" et la durée de la vidéo peut ainsi être illimitée. Ceci est utile pour la diffusion en direct.
- **Possibilité de retransmission en direct** : Le flux vidéo peut être capturé, encodé et compressé, puis diffusé directement, seul un léger décalage sera perceptible.
- **Evènements « Live »** : Le streaming est le seul moyen de distribution des évènements en temps réel sur le net.
- **Adaptabilité à l'état du réseau** : Quel que soit l'état du réseau, la bande passante utilisable par le client, le streaming permet de lire la vidéo.
- **Le multicast** : plusieurs lecteurs peuvent prendre pour source un seul flux circulant sur le réseau, ce qui réduit considérablement le nombre de flux occupant le réseau.

## 3.5 Marché et perspectives

Le marché du streaming constitue l'un des secteurs de l'Internet possédant la plus forte croissance prévisionnelle, principalement liée à deux facteurs :

- L'évolution perpétuelle des technologies qui permettent de disposer d'une bande passante de plus en plus importante,
- L'extension de la communication des secteurs traditionnels de l'audiovisuel.

On ne peut pas passer sans donner des chiffres qui confirmeraient le bel avenir qui se présente devant cette technique. En effet, en 1998 en France, seulement 34% des sites d'actualité, de loisirs et de sports offraient du son et de la vidéo contre 95% en 2001. D'un autre coté, les "streamers" sont plus consommateurs d'Internet que les autres : ils passent 50% de temps en plus sur les sites qui offrent du son ou de l'image [2].

## 3.6 Architecture du système de streaming

La diffusion de médias sur Internet fonctionne selon un modèle client-serveur. La communication audio et vidéo va du serveur vers le poste client comme la montre la figure suivante (Figure 3).

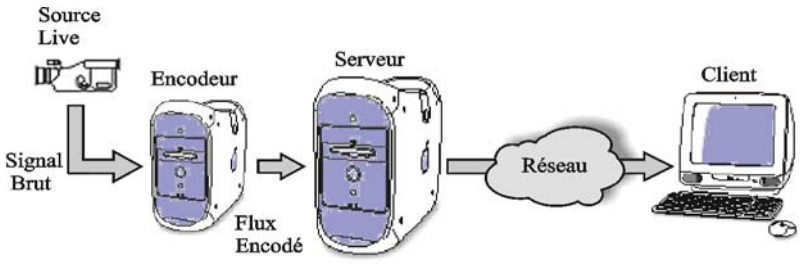


***Figure 3 : Architecture du Système de Streaming***

Les éléments constituant cette architecture sont :

### 3.6.1 L'encodeur

Les opérations dont est chargé cet élément du système de streaming se résument à la production de contenu. Le signal sera encodé dans un format

numérique spécifique, qui réduit la taille du fichier en substituant des compressions aux images originales, avant d'être mis en ligne.

### 3.6.2 Le serveur de médias

C'est la partie la plus intéressante dans le système. Un serveur doit être équipé d'un logiciel spécialisé pour l'émission des données encodées sous forme de paquets. Le serveur transforme le fichier vidéo demandé sous une version adaptée à la connexion spécifiée du lecteur au travers d'un dialogue permanent pour réguler le flux.
Donc, Le but d'un serveur de streaming est de diffuser un flux avec la meilleure qualité possible. Il doit, donc, trouver un compromis entre qualité et débit.

### 3.6.3 Le réseau

Les médias à la demande peuvent circuler sur tous les réseaux IP, intranet d'entreprise, Internet. Le réseau sur lequel circulent les données du streaming pose quelques problèmes lors de l'émission d'un flux média : limitation de la bande passante, perte des paquets, types de protocoles de transport sur le réseau, présence de certains composants tels que les FireWalls qui empêchent le passage des paquets UDP, …

### 3.6.4 Le client

Comme pour le serveur, un logiciel spécialisé doit être installé sur le terminal client pour recevoir les paquets du flux multimédia : un clic sur la vidéo désirée, et elle commence à jouer après quelques secondes d'attente.

# 4. Protocoles

Il existe actuellement trois protocoles qui permettent de faire du streaming. Les deux premiers, HTTP et FTP, sont des protocoles de transfert de fichier. On peut néanmoins parler de streaming dans la mesure où le flux peut être affiché au fur et à mesure du téléchargement. Ceci dit, du fait de leurs mécanismes de contrôle d'erreurs qui ont tendance à ralentir la transmission ou à décaler le son et la vidéo, on considère que c'est RTP ou Real Time Protocol qui permet de faire du vrai streaming c'est-à-dire de la diffusion de contenu en temps réel.

## 4.1 Le Real Time Transport Protocol, RTP [3]

RTP (RFC 3350) est un protocole basé sur IP fournissant un support pour le transport des données en temps réel comme la vidéo. Les services fournis par RTP consistent en la reconstruction temporelle, la détection de pertes, l'identification du contenu et la synchronisation entre médias.

RTP a été conçu à l'origine pour des données temps réel multicast, mais il peut aussi être utilisé pour faire de l'unicast. Il peut également être utilisé pour le transport dans une seule direction comme la VoD (Video On Demand) mais aussi pour des applications interactives telles que la téléphonie à travers Internet. RTP est conçu pour travailler en conjonction avec le protocole RTCP afin de permettre la mesure des performances et le contrôle de la session en cours.

Typiquement, RTP est exécuté au dessus de UDP (Figure 4). Les blocs de données, générés par la sortie de l'application multimédia, sont encapsulés dans des paquets RTP, puis chaque paquet RTP est à son tour encapsulé dans un segment UDP. Considérons l'exemple d'une application téléphonique sur RTP. Supposons que la source soit encodée au format PCM à 64 kb/s et que les échantillons soient disponibles toutes les 20 ms. L'application serveur ajoute à chaque bloc un entête RTP qui inclut le type de codage, un numéro de séquence et une estampille de temps. L'en-tête et le bloc de données forment le paquet RTP. Ce paquet est ensuite envoyé sur un socket UDP où il sera empaqueté dans un paquet UDP. Du coté client, c'est un paquet RTP qui est reçu via le socket. L'application cliente peut extraire le bloc de données et utiliser l'en-tête pour le décoder puis le jouer correctement.

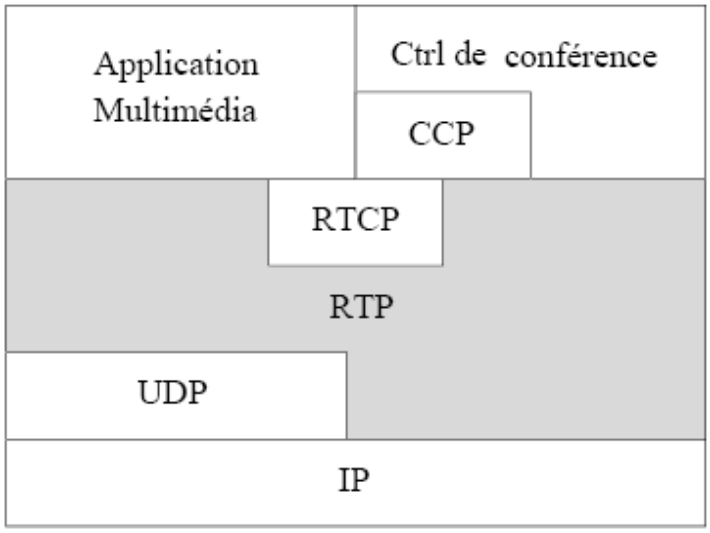


***Figure 4 : La place de RTP dans les couches réseaux***

RTP permet d'associer un flot de paquets à chaque type de source, par exemple un flot pour un micro et un autre pour une caméra. Dans ce cas, pour une vidéoconférence à deux, 4 flots seront créés, 2 flots audio et 2 flots vidéo dans chaque sens. Cependant dans de nombreux formats de compression dont MPEG1 et MPEG2, les images et le son sont mélangés. Pour de tels cas, seul un flot est créé dans chaque sens. RTP est notablement compatible avec les réseaux multicast.

### 4.1.1 Les entêtes RTP

L'entête d'un paquet RTP (Figure 5) est obligatoirement constituée de 12 octets, éventuellement suivie d'une liste d'identificateurs de sources contributrices CSRCs dans le cas d'un mixeur. Cette entête précède le "payload" qui représente les données utiles.

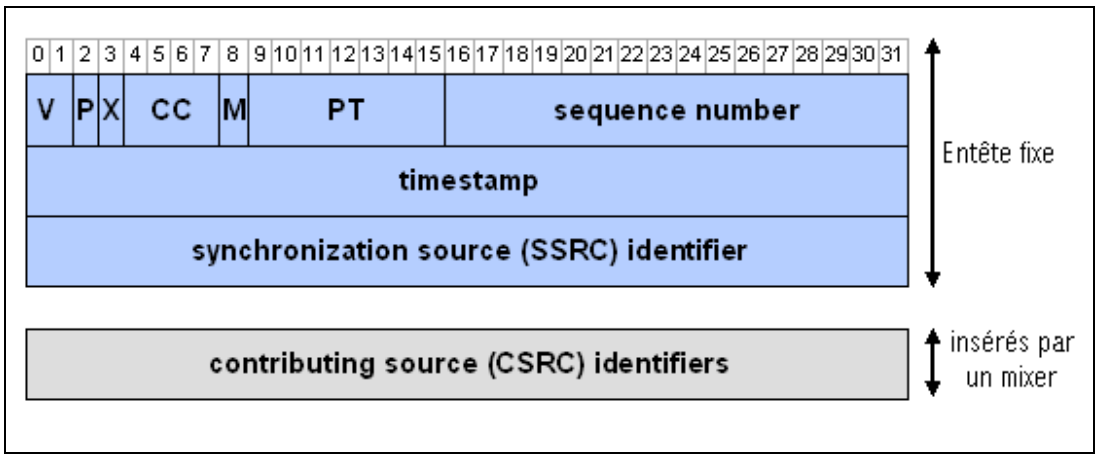


***Figure 5 : Entête RTP***

L'entête RTP comporte les champs suivants:

- version (V) : 2 bits, indique la version du protocole (V=2).
- padding (P) : 1 bit, si P est égal à 1, le paquet contient des octets additionnels de bourrage.
- extension (X) : 1 bit, si X=1 l'entête fixe est suivie par une extension.
- CSRC count (CC) : 4 bits, contient le nombre de CSRC identifiers qui suivent l'entête.
- marker (M) : 1 bit, son interprétation est définie par un profil d'application.
- payload type (PT) : 7 bits, identifie le format des données contenues dans le paquet RTP. Un profil définit de façon statique la correspondance entre un type de données et le format des données (voir RFC 1890).
- sequence number : 16 bits, sa valeur initiale est aléatoire et il est incrémenté de 1 pour chaque paquet envoyé.
- timestamp : 32 bits, reflète l'instant où le premier octet du paquet RTP a été échantillonné. Cet instant doit être dérivé d'une horloge qui augmente de façon monotone et linéaire dans le temps pour permettre la synchronisation et le calcul de la gigue à la destination.
- SSRC: 32 bits, identifie de manière unique la source.
- Le champ CSRC: 32 bits, identifie les sources (SSRC) qui ont contribué à l'obtention des données contenues dans le paquet. Le nombre d'identificateurs est donné dans le champ CC.

RTP est conçu pour travailler en conjonction avec le protocole RTCP afin de permettre d'obtenir des feed-back concernant la qualité de la transmission et des informations au sujet des participants pour la session à réaliser.

## 4.2 Le Real Time Control Protocol, RTCP [4]

Le protocole RTCP permet de contrôler en temps réel l'état d'une session RTP typiquement représentée par la figure 6. Les paquets RTCP ne comportent pas de données multimédia mais transportent des rapports qui contiennent des statistiques utiles à l'application. Parmi ces statistiques, on trouve le nombre de paquets envoyés, le nombre de paquets perdus et la variation des délais (jitter). Les spécifications de RTP ne définissent pas la manière d'utiliser ces informations, leur utilisation est laissée au libre choix des développeurs. Elles peuvent servir par exemple à modifier les taux de compression en cours de session. Les données RTCP sont transportées en utilisant le même mécanisme de distribution que celui des données. Le protocole utilisé doit donc permettre le multiplexage des paquets de données et de contrôle, par exemple, avec RTP, en utilisant deux ports UDP distincts.

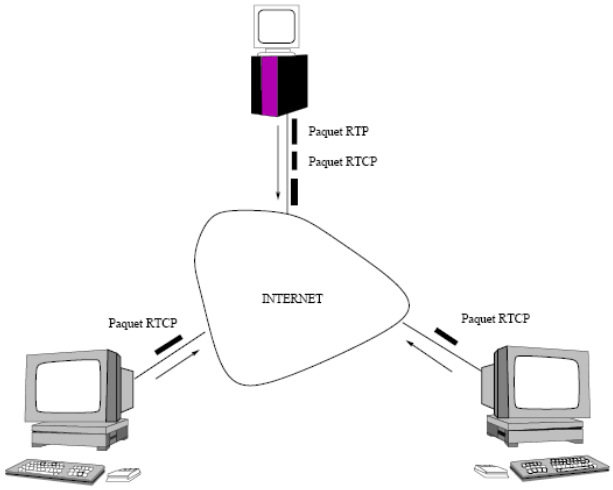


***Figure 6 : Une session RTP typique***

RTCP fournit principalement trois fonctions à savoir :

- Contrôler la congestion et fournir un monitoring de la QoS : Ceci constitue la fonction primordiale de RTCP. Il fournit un feed-back à l'application au sujet de la qualité de la distribution des données. L'information de contrôle est utile aux émetteurs et aux récepteurs. L'émetteur peut ajuster sa transmission en se basant sur le rapport du récepteur. Les récepteurs peuvent déterminer si une congestion est locale, régionale ou globale.

- Permettre l'identification de la source : les paquets RTCP admettent des « canonical names » comme des identificateurs uniques pour les participants de la session.
- Permettre la synchronisation inter média : Les rapports des émetteurs RTCP contiennent une indication de temps réel et l'horloge de temps RTP correspondante. Ceci peut être utilisé pour la synchronisation inter média (vidéo).

Il est à noter que le trafic RTCP devrait être gardé en dessous de 5% du trafic de la session.

## 4.3 Real Time Streaming Protocol (RTSP) [5]

RTSP est un protocole de niveau applicatif, conçu pour diffuser efficacement des données audio-visuelles pour un grand nombre de personnes.
Comme une télécommande de magnétoscope [6], le protocole RTSP fournit un mécanisme qui permet aux utilisateurs de demander spécifiquement des données d'un ou de plusieurs serveurs, ainsi qu'un type de transfert spécifique et une destination de transmission des données. Le demandeur de flux peut ainsi lancer, arrêter et mettre en pause la transmission des données. Il est de plus possible d'obtenir l'accès direct à différents paquets de données (c'est impossible dans le cas d'une transmission en direct).
Le protocole RTSP a été conjointement développé par Real Networks, Netscape et Columbia University au sein du groupe MMUSIC de l'IETF (Internet Engineering Task Force).

De même que H.323 (standard promu par l'UIT-T pour les applications de visioconférence), RTSP utilise RTP pour former les paquets contenant les données multimédia. Cependant, alors que H.323 est conçu pour la vidéo conférence, et donc pour un nombre modéré de personnes, RTSP est lui conçu pour diffuser efficacement des données audiovisuelles à un grand nombre de personnes.

### 4.3.1 Les principes de RTSP

i. **Un protocole temps réel selon un modèle client/serveur :** RTSP est un protocole temps réel qui a été conçu pour transférer et contrôler un ou plusieurs flux synchronisés d'objets média continus comme la vidéo et l'audio. Le protocole est fondé sur le modèle client/serveur. Les commandes de contrôle de la présentation d'un objet média (lecture, pause, ...) sont traduites sous forme de requêtes d'accès RTSP envoyées par le client et exécutées par le serveur.
ii. **RTSP dépassant les limites de HTTP :** RTSP a été créé car on ne pouvait se satisfaire de HTTP, celui-ci étant fait pour le transfert de page Web n'intégrant

pas des contraintes de temps. Cependant RTSP hérite de toute la syntaxe de HTTP, de ses mécanismes de sécurité et de ses procédures d'extension.

iii. **Un protocole à états :** RTSP est un protocole avec états qui utilise la notion de session. Cette caractéristique est importante pour effectuer le transfert de médias continus. En effet, la présentation d'objets de type média continu est caractérisée par une évolution entre différents états de présentation, comme les états Unmapped, Mapped, Active, Terminated et Suspended. L'évolution des transferts de données effectués au moyen du protocole RTSP est décrite par une machine d'états associée au client et au serveur. Le serveur change d'état quand il reçoit une requête du client, et le client change d'état quand il envoie une requête au serveur.

Les principales méthodes sont :

- **DESCRIBE :** Elle est utilisée par le client pour récupérer la description d'une présentation ou d'un objet média, identifié par un URI, sur un serveur RTSP.

- **SETUP :** Elle est utilisée par le client pour spécifier au serveur les paramètres de transport du flot média, comme par exemple le type de protocole de transport (RTP), le mode de transport (point à point ou multipoint) et le numéro du port de communication.

- **PLAY :** Elle signale au serveur qu'il peut commencer à envoyer les données via le mécanisme spécifié dans la méthode SETUP. Elle permet également de jouer un ou plusieurs sous-intervalles de la durée d'un objet média, par exemple jouer une vidéo à partir de la 10ème seconde jusqu'à la 20ème seconde.

- **PAUSE :** Elle est utilisée par le client pour demander au serveur d'interrompre temporairement le transfert du flux média. Le client peut reprendre la transmission du flux en envoyant la requête PLAY au serveur.

- **TEARDOWN :** Elle est utilisée par le client pour demander au serveur d'arrêter définitivement le transfert du flux média.

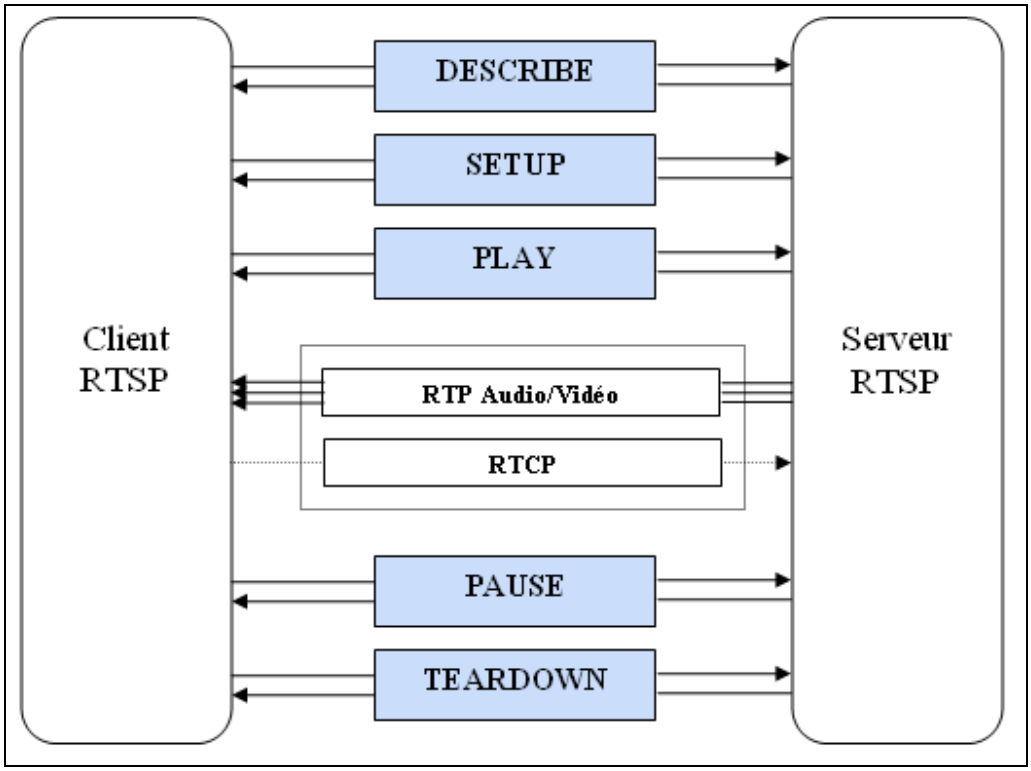


***Figure 7 : Echanges entre un serveur et un client RTSP***

### 4.3.2 Le streaming unicast avec RTSP

Le client contacte le serveur de streaming grâce au protocole RTSP. En réponse à cette requête, le serveur retourne via RTSP une description de la session de streaming qu'il va ouvrir. Une session de streaming est composée d'un ou de plusieurs flux (stream), par exemple audio ou vidéo. Le serveur informe le client du nombre de flux. Il donne aussi des informations décrivant les flux comme le type du média et le codec de compression. Les flux sont quant à eux diffusés séparément via le protocole RTP comme illustré dans la figure ci-dessous (figure 8).

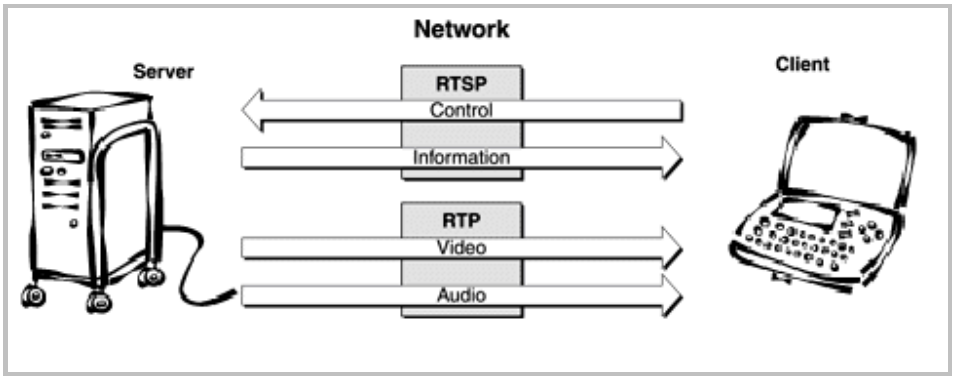


***Figure 8 : Echange de flux en unicast***

### 4.3.3 Le streaming multicast [1]

Dans le cas du streaming multicast, une seule copie de chaque flux est envoyée sur chaque branche du réseau, ce qui perme de réduire considérablement le trafic lors d'une diffusion pour de nombreux clients.
Une diffusion multicast est annoncée par un ficher SDP (Session Description Protocol) qui est téléchargé à partir d'un serveur web classique. Ce fichier contient les informations nécessaires pour recevoir le flux multicast, adresse IP du serveur, numéro du port et les informations de description des flux (même informations que celle envoyées par RTSP dans le cas d'une diffustion unicast). La figure 9 illustre cet échange multicast.

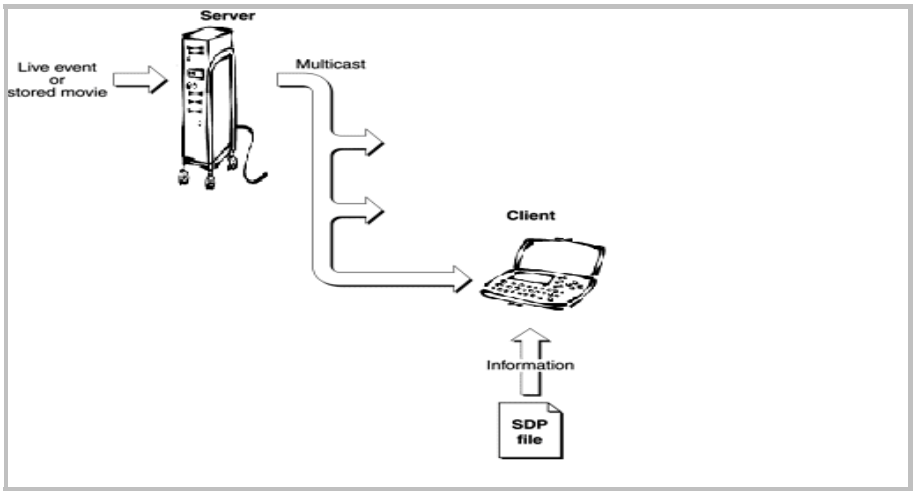


***Figure 9 : Echange de flux en multicast***

Actuellement, tous les routeurs ne supportent pas le mutlicast. Afin de permettre aux clients situés derrière ces routeurs d'accéder aux données multicast, il est possible d'installer un serveur de streaming qui va agir comme une passerelle entre multicast et unicast. Ce serveur est connecté aux flux multicast et sert aux clients qui se connectent à lui ces flux sous la forme de flux unicast en utilisant RTP et RTSP. Cette opération s'effectue en temps réel, ce qui permet de retransmettre aussi bien des vidéos préenregistrées que des images en direct. Le schéma qui suit décrit ce mécanisme (Figure 10).

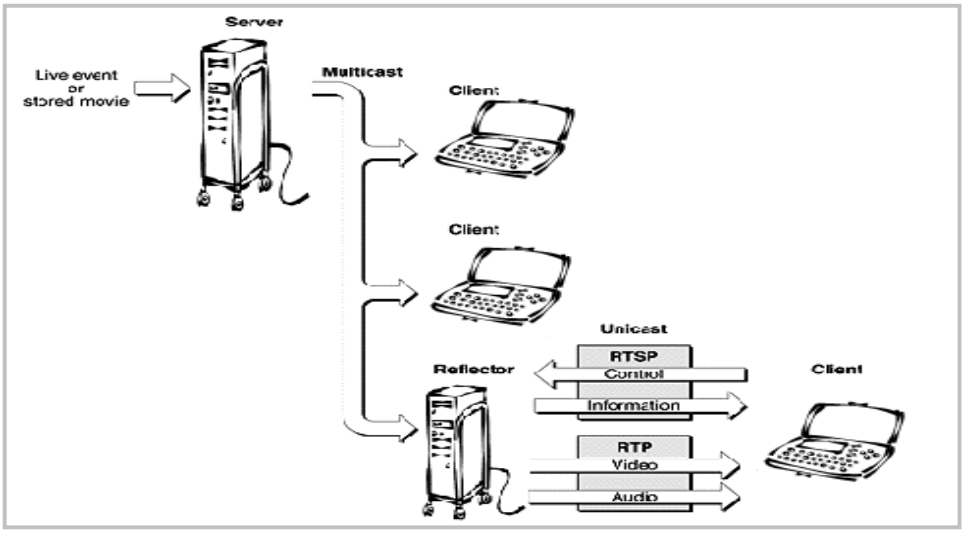

*Figure 10 : Multicast à travers les routeurs*

# 5. La Compression

Les fichiers multimédia ont une taille énorme, qui poserait problème, non pas au niveau du stockage - compte tenu des progrès réalisés dans ce domaine - mais lors de leur acheminement sur des supports de transmission dont la bande passante est limitée.
C'est là où la nécessité de la compression devient palpable. La compression vise à réaliser un compromis entre la qualité et la taille. Il s'agirait de supprimer les redondances par exemple ou les périodes de silence.
Un codec est un algorithme de compression utilisé pour réduire la taille d'un flux. Il y a des codecs audio et des codecs vidéo (MPEG-1, MPEG-2, MPEG-4, Vorbis, DivX..)
Un conteneur contient plusieurs flux déjà encodés à l'aide de codecs. Généralement, il y a un flux vidéo et un flux audio. On pourrait citer AVI, Ogg, MOV, ASF, MP4 comme exemple de conteneurs. Les flux contenus peuvent être encodés à l'aide de différents codecs. Il faut toutefois faire attention aux risques d'incompatibilités.
Ainsi, avant de transmettre un fichier, il faut l'encoder puis le multiplexer avec d'autres flux séparés. Pour le lire, il suffit de réaliser l'opération inverse.

## 5.1 La compression audio

Pour coder les fichiers audio, on peut utiliser :

- les propriétés de l'oreille humaine, (l'effet de masque).
- les propriétés du signal lui-même (propriétés énergétiques du signal) qui permettent d'optimiser en attribuant un nombre réduit de bits de quantification aux sous bandes de hautes fréquences ne contenant pas trop d'énergie.

Les trois principales normes de compression utilisées pour l'audio sont :

- MACE : La compression/décompression est très rapide, mais la qualité audio est grandement réduite. Par ailleurs MACE présente l'inconvénient de ne supporter que les fichiers audio 8 bits. Enfin sa compatibilité est quasi-limitée aux plateformes MAC.
- MPEG audio : La norme MPEG audio (MP2, MP3) fournit une compression de très bonne qualité audio et des fichiers de taille réduite, mais a longtemps nécessité une machine puissante (PowerMac ou Pentium) tant le codage/décodage est gourmand en puissance. Le développement impressionnant de la norme MPEG en fait aujourd'hui une norme incontournable.
- IMA : La norme IMA est une solution intermédiaire entre MACE et MPEG : elle permet un codage/décodage relativement rapide pour une altération réduite de la qualité. IMA 4:1 est la norme de compression audio standard de QuickTime.

## 5.2 La compression vidéo

La compression vidéo utilise des algorithmes se basant sur l'élimination des redondances qu'elles soient spatiales (entre pixels ou blocs voisins dans une même image) ou temporelles (entre images successives dans une séquence vidéo).
Cette compression peut se faire avec ou sans perte d'information.
Une compression avec perte d'information est qualifiée d'irréversible.
Une compression préservant toutes les informations est dite réversible.

Parmi les normes de compression d'images, on peut citer H261 et H263 utilisées dans la visiophonie et la visioconférence, ainsi que le MPEG.

Ces normes sont détaillées ci-après :

- **H-261** est à la base des codeurs vidéo actuels. Développé par C.C.I.T.T. (Commission Consultative Internationale de la Télégraphie et de la Téléphonie). C'est un codage hybride permettant d'éliminer à la fois la redondance spatiale et temporelle. Il code deux sortes d'images, les images I (Intra) et les images P (Predicted). Une image I ou I-frame est identique à une image codée par JPEG. Dans une image P ou Predicted-Frame, on ne code que la différence entre l'image courante et une référence qui est la dernière image I. La complexité de recherche du mouvement entre deux images est très grande. De plus, ce type de codage est peu tolérant aux pertes. Intra-H261 est une variante de H-261 présenant l'avantage d'être plus tolérant aux pertes. Il utilise la reconstitution conditionnelle.

- **MPEG-1 ou ISO/IEC 11172 :** [7] ce standard permet de compresser les données vidéo et les canaux audio associés. En effet, il comprend une partie système, une partie vidéo, une partie audio, une partie tests de conformité et une partie simulation logicielle. Il permet un débit 1.5Mb/s ce qui a fait de lui la norme de stockage de vidéo sur CD-ROM au format CD-I ou CD-vidéo.

- **MPEG-2 ou ISO/IEC 13818 :** [7] ce standard comprend en plus des parties du MPEG-1 quatre autres parties qui sont des extensions. Ayant un débit de 4 à 5 Mb/s il permet donc une compression vidéo qualité télévision. Son évolution lui a permis d'atteindre un débit de 40 Mb/s ce qui en fait un standard pour la télévision haute définition.

- **MPEG-4 ou ISO/IEC 14496 :** [8] un standard qui permet de faire coexister informatique, télécommunications et la télévision. MPEG-4 englobe toutes les nouvelles applications multimédias comme le téléchargement et le streaming sur Internet, le multimédia sur mobile, la radio numérique, les jeux vidéo, la télévision et les supports haute définition. Il porte sur la vidéo une vision orientée objet. Ce qui permet une plus grande interactivité. Cette approche orientée objet

décrit une scène comme une arborescence hiérarchisée d'images fixes, de vidéos en mouvement sans arrière plan et des objets audios et ce selon un script relatant leurs relations spatiales et temporelles. Il est associé au codec div-X.

# III. QUALITE DE SERVICE (QoS) ET SECURITE DANS LES RESEUAX Wi-Fi

## 1. Introduction

Aujourd'hui, la fiabilité de la communication reste le dernier grand facteur qui empêche le développement de services vidéo sur les réseaux sans-fil (TV, Vidéosurveillance, visioconférence, etc.). Le lien sans-fil est, en effet, caractérisé par des fluctuations aléatoires qui provoquent des pertes de paquets et, par la même, des dégradations importantes dans la qualité de la vidéo. Il existe plusieurs techniques pour palier ce problème (ex., retransmission, FEC) avec différents niveau de consommation de la bande passante, et différents délais de boute en bout.

La sécurité des systèmes d'information vise à garantir la confidentialité, l'intégrité et la disponibilité des services. C'est une tâche difficile, tout particulièrement dans un contexte de connectivité croissante et mobile.

Pour améliorer la sécurité, il faut mettre en place des mécanismes, d'une part pour assurer que seules les personnes autorisées puissent consulter ou modifier des données et d'autre part pour assurer que les services puissent être offerts correctement.

Dans ce chapitre, nous commencerons dans une première partie par décrire l'architecture adoptée par les réseaux locaux sans fils WLAN 802.11. Par la suite nous

passons à l'étude de la qualité de service (QoS). Finalement, nous présentant les mécanismes de sécurité intégrés dans la norme Wi-Fi et leurs défaillances.

# 2. La norme Wi-Fi [9]

La comité 802.11 a été crée par l'IEEE (Institute of Electrical and Electronics Engineers) dans le but de standardiser les réseaux locaux. Ce groupe a contribué à l'initiation de la norme des réseaux locaux sans fils IEEE 802.11 en 1990 puis elle a été publié en 1997. Elle couvre les deux couches basses du modèle OSI à savoir la couche MAC et la couche physique et fonctionne suivant des topologies s'apparentant davantage au monde de la téléphonie mobile qu'à celui des réseaux filaires.

Le terme Wi-Fi est une abréviation communément utilisée pour désigner l'ensemble des produits et des technologies dits de Wireless Fidelity en référence au norme IEEE 802.11b des réseaux sans fil à haut débit. Les réseaux concernés sont dits réseaux Ethernet à accès sans fil ou RLAN ou bien encore WLAN.

## 2.1 Architecture

La norme 802.11 définit deux modes : un mode infrastructure et un mode ad hoc.

### 2.1.1 Mode Infrastructure

En mode infrastructure (comme le montre la figure 11), le réseau sans fil consiste au minimum en un point d'accès connecté à l'infrastructure du réseau filaire et un ensemble de postes réseaux sans fil. Cette configuration est appelée BSS, ou ensemble de services de base. Un ESS, ou ensemble de services étendu est un ensemble d'au moins deux BSS formant un seul sous-réseau. En entreprise, la plupart des WLAN devront pouvoir accéder aux services pris en charge par le LAN filaire (serveurs de fichiers, imprimantes, accès Internet). Aussi fonctionneront-ils en mode infrastructure.[10]

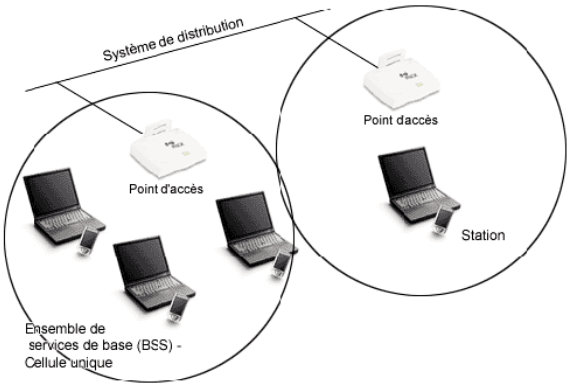

*Figure 11 : Mode infrastructure*

### 2.1.2 Mode ad hoc

Le mode ad hoc (également appelé point à point, ou ensemble de services de base indépendants), représente simplement un ensemble de stations sans fil 802.11 qui communiquent directement entre elles sans point d'accès ni connexion à un réseau filaire. Ce mode permet de créer rapidement et simplement un réseau sans fil là où il n'existe pas d'infrastructure filaire ou encore là où une telle infrastructure n'est pas nécessaire pour les services attendus, ou enfin lorsque l'accès au réseau filaire est interdit (cas du consultant sur le site du client).

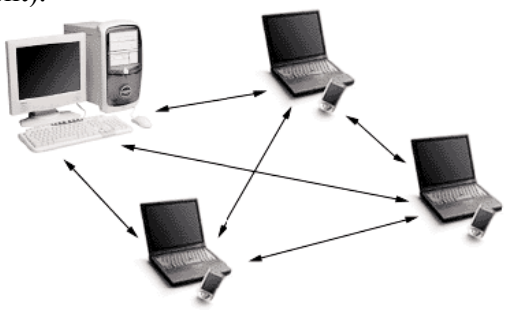

*Figure 12 : Mode ad hoc*

## 2.2 Structure des couches

Comme tous les normes IEEE 802, la norme 802.11 se concentre sur les deux couches inférieures du modèle ISO, la couche physique et la couche des liaisons données. [11]

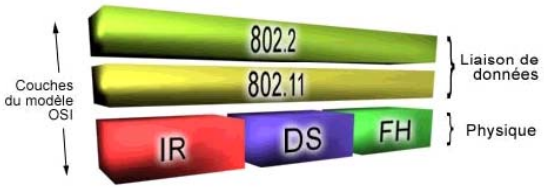


*Figure 13 : description des couches 802.11*

### 2.2.1 La couche physique

Selon la définition du modèle OSI, «La couche physique fournit les moyens mécaniques, électriques, fonctionnels et procéduraux nécessaires à l'activation, au maintien et à la désactivation des connexions physiques destinées à la transmission des éléments binaires entre entités de liaison». La couche physique 802.11, comme le montre la figure 1.3 joue le rôle d'interface entre la couche MAC et le support de transmission. Trois fonctions sont attribuées à cette couche:

- Servir d'interface entre la couche MAC et le support.

- Transmettre le signal grâce à une porteuse et une modulation d'étalement de spectre, qui sera détaillé dans la partie suivante.
- Vérifier l'activité du canal.

### 2.2.2 La couche liaison de données

La couche de liaison de données de 802.11 se compose de deux sous couches : le contrôle de la liaison logique, ou LLC et le contrôle d'accès au support, ou MAC. La norme 802.11 utilise la LLC 802.2 et l'adressage sur 48 bits, tout comme les autres LAN 802, simplifiant ainsi le pontage entre les réseaux sans fil et filaires. Le contrôle d'accès au support est en revanche propre aux WLAN.

# 3. La QoS dans les réseaux Wi-Fi

## 3.1 Définition

D'après la recommandation I.350 du CCITT, la qualité de service est :

- L'effet collectif d'une performance de service
- Détermine le degré de satisfaction d'un utilisateur de service
- Un ensemble de caractéristiques de performance de service, perçus et exprimés par l'utilisateur.

## 3.2 Paramètres de qualité de service

Les paramètres de qualité de service dans un réseau sans fil sont :

- Le délai de propagation : c'est le temps de transmission auquel s'ajoute le temps passé au sein des éléments.
- La gigue d'inter-arrivée : variation du délai de propagation des paquets de bout en bout.
- La bande passante : correspond au débit de transfert maximum entre deux points terminaux.
- Le taux de perte : taux de pertes moyen des paquets.

Ces paramètres peuvent être classés en :

- Des paramètres qualitatifs qui ne peuvent pas être mesurés directement mais perceptibles par l'utilisateur (qualité de son, de l'image …).
- Des paramètres quantitatifs qui peuvent être directement mesurés aux points d'accès au service (délai, débit,…).

## 3.3 Les mécanismes classiques de contrôle d'erreur

Pour assurer la fiabilité des transmissions par paquet, on distingue deux grandes classes de mécanismes :

- Les mécanismes réactifs (ARQ : Automatic Repeat reQuest) : l'émetteur réagit `a la signalisation d'une perte de paquet en retransmettant ce paquet. Cette signalisation peut être effectuée par l'émission d'acquittements positifs ou négatifs.
- Les mécanismes proactifs (FEC : Forward Error Correction) : l'émetteur rajoute a priori des paquets de redondance permettant au récepteur de récupérer certains paquets perdus. Ces paquets de redondance sont calculés en utilisant des codes correcteurs d'erreurs.

### 3.3.1 Les mécanismes FEC

Les codes média indépendants sont des codes qui ne tiennent pas compte du type des données contenues dans les paquets lors de la génération des paquets de redondance. Les codes de parité, les codes de type Reed-Solomon et les codes Tornado représentent de tels exemples de codes.

Ces codes prennent un ensemble de k paquets d'information (un bloc d'information) à partir desquels sont générés h paquets de redondance. On obtient ainsi un bloc FEC de taille n = k+h paquets. Si moins d'une certaine quantité de paquets (allant jusqu'à h) est perdue, le bloc d'information peut être reconstitué intégralement. Sur la Figure 14, nous pouvons voir un exemple de code FEC où un paquet de redondance est généré à partir de quatre paquets d'information. Si un paquet d'information est perdu, on se servira alors de l'information redondante pour reconstituer le paquet perdu.

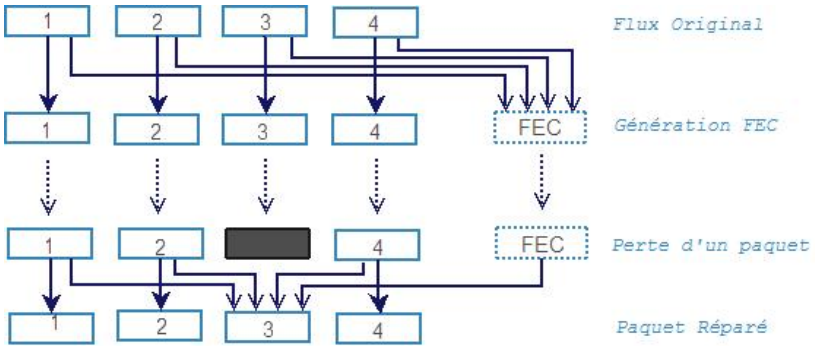


***Figure 14 : Principe du codage FEC***

#### 3.3.1.1 Code de parité

On considère un bloc d'information constitué de k paquets d'information auquel on ajoute un unique paquet de redondance p afin de former un bloc FEC de taille n = k + 1 paquets.

Le code de parité est très simple à mettre en œuvre puisqu'il consiste en une simple addition bit à bit à l'aide de l'opérateur logique " ou exclusif ".

Par ailleurs, la correction d'un paquet perdu est assez rapide car il suffit de faire également la somme bit à bit des (n – 1) paquets non perdus pour connaître le contenu du paquet perdu. Néanmoins, malgré la simplicité de sa mise en oeuvre, le code de parité n'est pas capable de corriger plus d'un paquet appartenant à un bloc FEC. Si deux paquets (ou plus) appartenant à un même bloc sont perdus, alors aucune correction ne sera possible. Il est donc nécessaire de ne pas prendre des blocs de donnée de trop grande taille.

#### 3.3.1.2 Code de type Reed-Solomon

Un code de type Reed-Solomon [12] est utilise pour générer des paquets de redondance, permettant ainsi la reconstruction d'éventuels paquets d'information perdus.

Pour cela, on considère un bloc d'information constitue de k paquets d'information $d_1$, $d_2$,… $d_k$, chaque paquet étant de taille m bits, et on génère à l'aide d'un codeur RSE (Reed-Solomon Erasure code) h paquets de redondance $p_1$, $p_2$,…. $p_h$ afin de former un bloc FEC de taille n = k + h. On désigne ce code par le couple (n, k). En général, les codes RSE sont utilisés pour des tailles de bloc FEC assez petites car autrement les temps de codage et décodage sont trop importants.

Si les k paquets d'information transmis à destination ne subissent aucune perte, alors il n'est pas utile de réparer les éventuels paquets de redondance perdus car ces derniers ne contiennent aucune information utile. C'est seulement dans le cas où des paquets d'information sont perdus que leur réparation est nécessaire.

Si le nombre total de paquets perdus (les paquets d'information et les paquets de redondance) est inférieur ou égal à h = n - k, le décodeur RSE est capable de réparer tous les paquets perdus. Ainsi, tous les paquets d'information seront sauvés. Au contraire, si le nombre de paquets perdus est supérieur (strictement) à h, il est impossible pour le décodeur RSE de réparer les paquets perdus.

#### 3.3.1.3 Code de type Tornado

Nous avons vu dans la section précédente que les codes de type Reed-Solomon étaient utilisés de préférence pour des blocs FEC de petite taille du fait de la grande complexité de décodage des paquets perdus. Pour des blocs FEC de grande taille, il est préférable d'utiliser des codes de type Tornado car la complexité de décodage est bien moins importante. Un code Tornado (n, k) nécessite la réception de k + ε paquets parmi les n paquets constituant le bloc FEC, où ε est un paramètre du codage Tornado ayant une valeur non nulle proche de 0.

La génération des paquets de redondance est à base d'opérations ou-exclusif et un paquet perdu peut être reconstitué facilement à l'aide de ces mêmes opérations. La Figure 15 illustre la génération des h = 4 paquets de redondance à partir de k = 8 paquets d'information.

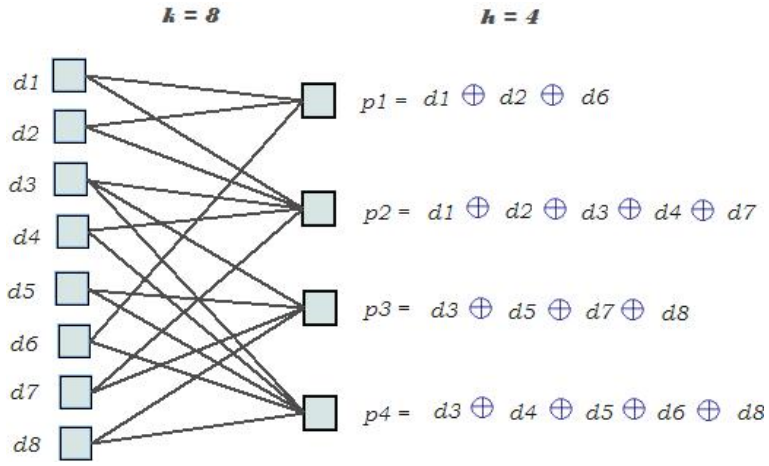


***Figure 15 : Génération des paquets de redondance a l'aide d'un code Tornado***

# 4. La sécurité des réseaux Wi-Fi

## 4.1 Les mécanismes de sécurité

La sécurité vise à garantir la confidentialité, l'intégrité et la disponibilité des services. Il faut mettre en place des mécanismes pour s'assurer que seules les personnes autorisées ont accès à l'information et que le service est rendu correctement.

Les réseaux locaux sans fils sont plus vulnérables que les réseaux filaires dans la mesure où la transmission se fait par voie radio et que la mobilité est inhérente à ce choix. Les données sont par conséquent diffusées et non sécurisées. Afin de pallier les problèmes de sécurité, les standards IEEE 802.11 incluent des mécanismes de sécurité de niveau couche MAC pour protéger les trames de données. Dans la première génération de WLAN, deux mécanismes ont été introduits :

- SSID (Service Set Identifier)
- WEP (Wired Equivalent Privacy)

### 4.1.1 Le Service Set Identifier

L'une des caractéristiques les plus courantes des WLAN est l'usage d'un mot de passe, le SSID, qui permet un niveau de sécurité rudimentaire. Il est analogue au nom de réseau pour les stations sans fil et les points d'accès dans un sous-système WLAN donné et sert à isoler logiquement les usagers et les points d'accès qui forment ce sous-système. Il est soit déclaré en broadcast dans les trames de gestion, soit pré-configuré dans les stations. Ainsi, il peut être demandé à une station dans une trame de "challenge" (Probe-Request frame) lorsqu'une station tente de joindre un WLAN ou diffusé par des beacons périodiques par le point d'accès.

Dans tous les cas, l'utilisation des SSID pour contrôler l'accès est dangereuse dans la mesure où il n'est pas typiquement sécurisé. En effet, il est généralement diffusé dans les beacons pour des raisons de compatibilité.

### 4.1.2 IEEE 802.11 Wired Equivalent Privacy (WEP) [13]

Le standard IEEE 802.11b tente de fournir une confidentialité équivalente à celle des réseaux filaires via un schéma de chiffrement optionnel appelé WEP (Wired Equivalent Privacy) qui sécurise le flux de données dans les réseaux WLAN. Les membres de l'alliance WECA recommandent des clés de chiffrement minimales de 40 bits. Le but de WEP est de:

- Dénier l'accès aux usagers non autorisés qui ne possèdent pas la clé adéquate.
- Empêcher le décodage du trafic capturé sans possession de la clé WEP.

WEP est un mécanisme de chiffrement symétrique. Dès qu'il est activé, l'émetteur isole le contenu des trames de données, les données utiles (payload), et leur applique l'algorithme de chiffrement. Par la suite, il remplace le payload initial par le résultat de l'algorithme. Les trames de données ainsi cryptées sont transmises (le bit spécifiant WEP dans l'entête de la trame 802.11 prend alors la valeur 1) vers le récepteur qui applique le même algorithme. Le résultat est le corps initial de la trame, déchiffré, qui sera transmis aux couches supérieures.

WEP utilise l'algorithme RC4 inventé par Ron Rivest de RSA Data Security pour le chiffrement. Il s'agit d'un algorithme de chiffrement de flux symétrique qui supporte des clés de chiffrement de longueur variable. La clé est la seule information partagée par les entités en chiffrement et déchiffrement, sa longueur pouvant atteindre 256 bits. Les concepteurs du standard IEEE 802.11b préconisent l'utilisation de clés de 40 bits. Toutefois, les constructeurs fournissent des équipements qui supportent des clés allant jusqu'à 128 bits. Le standard IEEE 802.11 décrit l'utilisation de l'algorithme RC4 et de la clé dans WEP mais a omis la distribution et la négociation des clés. De plus, plusieurs fabricants ont choisi d'implémenter des solutions propriétaires pour la gestion et la configuration des clés WEP. Cette fâcheuse omission ralentit l'interopérabilité des différentes solutions et relègue la normalisation de ce point aux standards en cours.

Le standard IEEE 802.11 précise deux mécanismes pour la sélection des clés. Le premier mécanisme est un ensemble de quatre clés par défaut. Elles sont partagées par toutes les stations d'un sous-système sans fils et peuvent alors être utilisées par un terminal pour communiquer avec tous les autres terminaux. Néanmoins, ces clés perdent leur efficacité si elles ne sont pas périodiquement changées. [14]

Le deuxième mécanisme permet à une station de mettre en place un schéma de clés (Key mapping) avec une autre station. Ce choix est nettement plus sûr vu que le choix est moins statique et spécifique à une paire de stations. Ce mécanisme présente clairement une problématique de distribution à mesure que le nombre des stations augmente.

L'entête et l'enqueue WEP sont accolées au corps chiffré de la trame, la clé par défaut utilisée pour le chiffrement est indiquée dans le KeyID de l'entête avec le vecteur d'initialisation (IV), la valeur de contrôle d'intégrité (Integrity Check Value = ICV) est indiqué dans l'enqueue. La longueur de la clé est alors égale à la somme des longueurs de la clé WEP et du vecteur d'initialisation.

## 4.2 Les failles des premiers mécanismes de sécurité [15]

Cette partie détaille les différentes menaces de sécurité non résolues par les premiers mécanismes de sécurité utilisés dans les réseaux Wi-Fi et met en évidence les besoins de sécurité à implémenter, en particulier l'authentification.

### 4.2.1 Limites physiques (matérielles)

Un premier mécanisme de sécurité commun pour les WLAN est l'utilisation de clés WEP statiques préprogrammées dans les adaptateurs de réseau (NIC) et les points d'accès comme sécurité de base. Cette technique nécessite la programmation de milliers de clés et engendre un déni de service périodique. Elle est impossible à gérer sauf pour des architectures peu denses (quelques dizaines d'usagers). D'autre part, il est impossible de détecter les violations de sécurité sans gestion centralisée des clés et une politique d'authentification par usage. En effet, ce schéma ne prend pas compte les situations où plusieurs utilisateurs partagent la même machine ou lorsqu'un visiteur et un administrateur sont perçus comme une même personne dans un système où seule l'adresse MAC identifie l'entité cliente.

La seule solution efficace est par conséquent d'implémenter une authentification centralisée associée à une distribution dynamique des clés WEP basée sur des droits d'accès par opposition aux implémentations spécifiques aux machines.

### 4.2.2 L'usurpation d'identité

L'utilisation des adresses MAC des stations clientes comme mécanisme de contrôle d'accès aux WLAN est un des mécanismes de sécurité rudimentaires les plus communs. Vu que 802.11 n'identifie pas les usagers, les solutions à base d'adresses MAC sont mal appropriées pour le déploiement de WLAN d'entreprises à grande échelle. Par ailleurs, il est facile pour le commun des mortels de modifier l'adresse MAC d'un adaptateur NIC.

### 4.2.3 Les points d'accès factices (Rogue Access Points)

L'un des principaux inconvénients du schéma d'authentification à clé partagée est qu'il n'y a pas d'authentification mutuelle entre station et point d'accès. La station s'authentifie auprès du point d'accès mais l'authentification réciproque n'a pas lieu. Ceci ouvre la voie aux attaques de déni de service via des points d'accès factices dans le WLAN. Ces attaques redirigent des usagers légitimes vers des points d'accès qui simulent leur appartenance au WLAN. Ce genre d'attaque ne peut être évité que par une authentification mutuelle par laquelle les deux entités, station et point d'accès, prouvent leur légitimité et ce l'espace d'un temps raisonnable.

### 4.2.4 Les failles d'intégrité

Dans la norme 802.11, WEP supporte l'intégrité du chiffrement par paquet mais pas l'authentification par paquet. Cela peut compromettre la sécurité ou mener à une modification des données.

Avec un schéma de sécurité basé sur WEP et des réponses données à un paquet connu (tel qu'ARP, DHCP, TCP ou un accusé de réception), il est possible de déchiffrer un flux de données RC4. Cela permet à un intrus d'altérer les paquets jusqu'au changement du Vecteur d'Initialisation (IV). Bien qu'une telle attaque soit relativement difficile à accomplir par le biais d'une connexion déjà établie, les pirates sont capables de tout faire.

Les approches possibles pour atténuer cette faille de la sécurité incluent le changement de l'IV pour chaque paquet dynamiquement, l'augmentation de la longueur de l'IV et le changement périodique de la clé WEP. Par ailleurs, une partie de notre projet porte sur l'amélioration du WEP, en particulier l'utilisation d'algorithmes de chiffrement plus robustes tels que AES (Advanced Encryption Standard).

### 4.2.5 La révélation de données (Exposition Involontaire de Données)

#### 4.2.5.1 L'écoute passive

En écoutant les canaux de contrôle 802.11, un intrus peut obtenir des informations concernant le point d'accès et les stations. L'information pourrait inclure les adresses MAC du point d'accès, du client ou de stations internes ainsi que les temps d'association/désassociation. Ce type d'information peut être utilisé par les pirates pour une analyse à long terme du trafic qui pourrait fournir des détails sur les utilisateurs et les machines.

#### 4.2.5.2 La découverte des clés

L'usage de clefs WEP statiques est d'autant plus dangereux que les clés restent inchangées longtemps. Une gestion centralisée et dynamique des clés atténue fortement cette menace.

#### 4.2.5.3 Les attaques par dictionnaires

Pour certaines implémentations, les clefs WEP sont dérivées de mots de passe, expressions ou SSIDs partagés ce qui les rend plus vulnérables aux attaques. Dans ce cas, l'attaquant pourrait utiliser une grande liste de mots (ou encore dictionnaires), deviner un mot de passe et dériver la clef. En améliorant les algorithmes de génération des clés, ce genre d'attaque peut être repoussé

#### 4.2.5.4 Le déni de service

Les messages d'association/désassociation 802.11 ne sont ni chiffrés ni authentifiés. Ceci permet de diriger de faux messages de désassociation ver des stations clientes. L'emploi de MIC (Message Integrity Check) est une solution envisageable.

## 4.3 Le cryptage [16]

La cryptographie permet de chiffrer un message intelligible en un texte chiffré incompréhensible. Le déchiffrement est l'opération inverse. Un système cryptographique est dit à usage restreint si sa sécurité repose sur les opérations de chiffrement et de déchiffrement. Ce type de système offre peu de valeur dans le contexte contemporain de communication car il ne représente pas vraiment un rempart contre les cryptanalystes.

Un système est dit à usage général si sa sécurité ne repose pas sur le secret des opérations de chiffrement et de déchiffrement mais plutôt sur la clé. Cette approche est connue sous le nom du «principe de kerckhoff », ainsi nommé en l'honneur du cryptologue néerlandais du 19$^{ième}$ siècle qui fut le premier à l'énoncer.

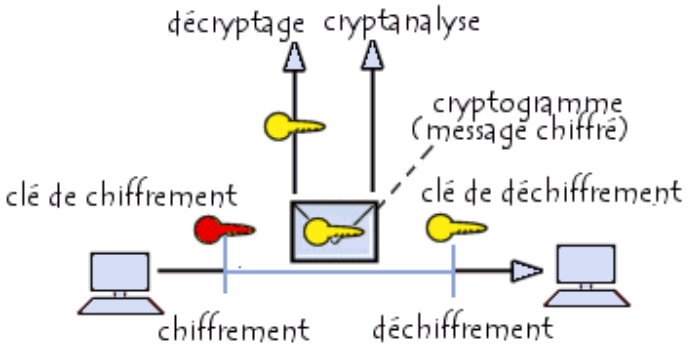


***Figure 16: Le mécanisme de cryptographie***

Il existe à l'heure actuelle deux grands principes de cryptage : le cryptage symétrique basé sur l'utilisation d'une clé privée et le cryptage asymétrique qui, repose sur un codage à deux clés, une privée et l'autre publique.

### 4.3.1 Le cryptage symétrique

Le cryptage à clé privée ou symétrique est basé sur une clé (ou algorithme) partagée entre les deux parties communicantes. Dans la plupart des systèmes symétriques cette même clé sert à crypter et décrypter les messages.

Le chiffrement peut se faire soit par bloc soit par flux :

#### 4.3.1.1 Le chiffrement par bloc

Il s'agit de transformer le texte en une suite binaire puis de découper cette chaîne en plusieurs blocs d'une longueur donnée. Ensuite, un algorithme de cryptage est appliqué à chacun de ces blocs. A noter que la taille de ces blocs ne doit pas se faire d'une façon aléatoire car elle a un impact direct sur la complexité et la sécurité. Un compromis est donc requis.

Le chiffrement par bloc se base sur la transformation. En effet, il est possible de faire un chiffrement par substitution (remplacer des symboles par d'autres), par transposition (mélanger les symboles) ou alors la combinaison des deux et c'est ce qui est le plus utilisé.

Différents modes d'opération sont utilisés pour implanter un chiffrement par bloc notamment :

- Electronic Code Book (ECB)
- Cipher Block Chaining (CBC)
- Cipher FeedBack (CFB)
- Output FeedBack (OFB)

Ils sont définis dans FIPS 81 et aussi dans ANSI X3.106-1983

- Les algorithmes utilisés

Plusieurs algorithmes sont utilisés pour réaliser le chiffrement par bloc, en particulier :

i. ***Blowfish*** : Il fut écrit par Bruce Shneier en 1993. il est gratuit et non breveté. Il permet de chiffrer des blocs de 64 bits de texte clair à l'aide de clés dont la taille peut aller de 32 à 448bits.
Son avantage est qu'il est rapide et simple. En outre, il a été finement analysé et sa solidité a été prouvée.

ii. ***DES :*** Le standard de chiffrement de données ou Data Encryption Standard a été développé par les laboratoires de la firme IBM Corp. Il est devenu le standard du NIST(National Institute of Standards and Technology) en 1976. Il a été mis en avant par le Bureau National des Standards américan (NBS) en 1977. il est public puisqu'il a été publié dans FIPS 46-3.
Cet algorithme chiffre un bloc de 64 bits à l'aide d'une clé secrète de 56 bits. Il transforme cette clé en 16 clés partielles de 48 bits au moyen d'un plan de génération des clés qui réutilise chacun des bits de la clé plusieurs fois. Après une présentation initiale fixe, le bloc de 64 bits de texte clair fait l'objet de 16 rondes suivies de l'inverse de la permutation initiale. Suivant le conseil de Shannon, chaque ronde effectue une étape de confusion suivie d'une étape de diffusion. Il est remarquable que l'étape de diffusion ne dépend pas de la clé ; la sécurité du système est augmentée en combinant confusion et diffusion même si l'une des deux transformation est fixe et publiquement connue.
L'algorithme est conçu de manière à ce que le déchiffrement s'effectue de la même façon que le chiffrement sauf bien entendu l'ordre des clés partielles qui est inversé au moment de leur génération.
Le principal problème est le partage de la clé et plus précisément de la façon dont elle pourrait être transmise d'une façon sécurisée. Ainsi la principale limite à l'utilisation du DES sur Internet réside dans le management des clés.

iii. ***RC2 RC5 RC6 :*** Tous furent créés par Ronald Rivest.
Le RC2 fut conçu en 1989 pour remplacer le DES car plus efficace avec les processeurs 16 bits. Son avantage était qu'il permettait de choisir la taille de la clé.
Le RC5, établi en 1995, permettait quant à lui une liberté totale que cela soit au niveau de la taille des blocs qu 'au niveau de celle des clés ainsi que du nombre de rondes.
Le RC6 créé en 1998, propose des améliorations au RC5.
Pour les trois, le déchiffrement se passe de la même manière que le chiffrement.

iv. ***Rijndael :*** Conçu par Joan Daemen et Vincent Rijmen. Le NIST a choisi cet algorithme pour remplacer le DES en tant que AES (Advanced Encryption Standard).

v. ***AES :*** (Advanced Encrytpion Standard) est fondé sur le Rijndael.
Il existe trois variantes de L'AES en fonction de la taille de la clé : "AES-128", "AES-192", "AES-256".

Le "AES-128" procède par blocs de 128 bits, avec une clé de 128 bits également. Chaque bloc subit une séquence de 5 transformations répétées 10 fois :

- Addition de la clé secrète (par un ou exclusif).
- Transformation non linéaire d'octets : les 128 bits sont répartis en 16 blocs de 8 bits eux même dispatchés dans un tableau 4×4. Chaque octet est transformé par une fonction non linéaire S.
- Décalage de lignes : les 3 dernières lignes sont décalées cycliquement vers la gauche : la 2ème ligne est décalée d'une colonne, la 3ème ligne de 2 colonnes, et la 4ème ligne de 3 colonnes.
- Brouillage des colonnes : Chaque colonne est transformée par combinaisons linéaires des différents éléments de la colonne.
- Addition de la clé de tour : A chaque tour, une clé de tour est générée à partir de la clé secrète par un sous-algorithme (dit de cadencement). Cette clé de tour est ajoutée par un ou exclusif au dernier bloc obtenu.

La figure suivante (Figure 17) présente les grandes lignes de l'AES-128 :

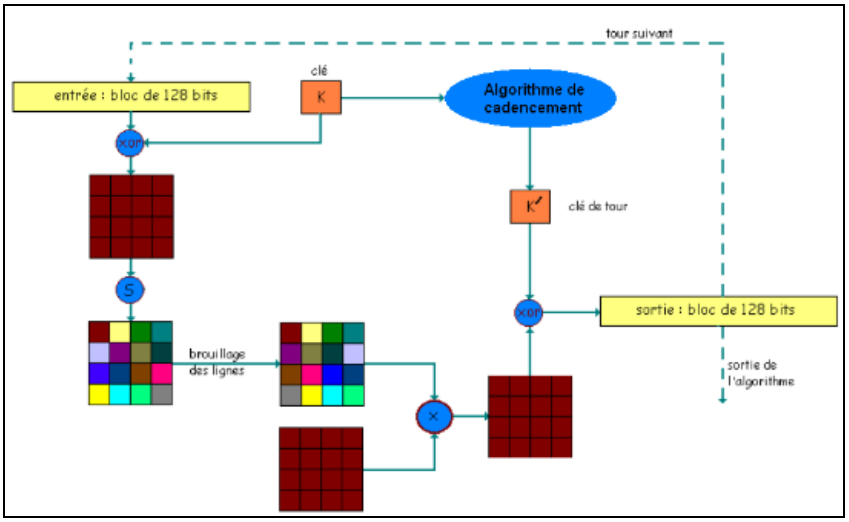


***Figure 17 : Principe de l'AES***

#### 4.3.1.2 Le chiffrement par flux

Le flux est un bloc très réduit de bits. Le chiffrement par flux permet de changer la méthode de transformation à chaque symbole tout en restant rapide. De ce fait il est particulièrement utile lorsqu'on manque de mémoire tampon.
Le chiffrement par flux utilise des séquences de bits générées aléatoirement mais qui ne sont utilisées qu'une seule fois ce qui le rend très sûr.

- Les algorithmes utilisés

Le RC4 est le plus connu et fut créé par Ron Rivest dans les laboratoires RSA en 1987.Cet algorithme commence par initier une table remplie par la clé. Des opérations diverses sont appliquées sur les différents octets de cette table. Ainsi, on peut extraire des séquences pseudo-aléatoires sur lesquelles on pourrait ensuite appliquer des opérations de OU EXCLUSIF (XOR) avec le texte clair.

### 4.3.2 Le cryptage asymétrique

Ce cryptosystème utilise deux clés différentes : une publique et accessible, via les annuaires par exemple, et l'autre, privée, qui n'est connue que par l'utilisateur lui-même. Les deux sont liées par un algorithme qui fait qu'un texte crypté par une clé publique ne peut être décrypté que par la clé privée correspondante.

- Les algorithmes les plus utilisés

### i. Diffie-Hellman

Il fut créé en 1976 par Whitfield Diffie et Martin Hellman. Son principe consiste à permettre aux deux intervenants de créer une clé secrète conjointement et en ligne. On choisit un grand nombre premier p, et un élément primitif g qui génèrent un groupe Zp*. p et q sont des constantes publiques. Le premier intervenant choisit un nombre x d'une façon aléatoire puis envoie gxmodp. Le récepteur choisit à son tour un nombre aléatoire y dans Zp* et envoie par la suite gymodp la clé secrète serait
k = gxy.
Bien que choisir g et x correctement rend cet algorithme pratiquement inviolable ou du moins très difficilement, il reste cependant sensible à l'attaque de l'intercepteur (man in the middle).

### ii. RSA

Cet algorithme fut inventé en 1978 par le trio Ron Rivset, Adi Shamir et Léonard Adleman. Il présente l 'avantage de servir aussi bien à l'authentification qu'au chiffrement.
Il est basé sur une fonction à sens unique à porte dérobée. Etant donné l'information publique n et e, il est facile de calculer $m^e$ (mod n) à partir de n, mais dans l'autre sens. Cependant, si on connaît la factorisation de n, alors il est facile de faire le calcul inverse. La factorisation est la porte dérobée. Si on la connaît alors on peut inverser la fonction autrement c'est impossible.
Pour commencer, on choisit aléatoirement deux grands nombres premiers différents p et q et on calcule n=pq. On emploie deux exposants e et d tel que ed=1 (mod t) où t :=ppcm(p-1, q-1)
Pour chiffrer un message m, l'expéditeur calcule le texte chiffré c :=$m^e$ (mod n).
Pour déchiffrer un texte chiffré c, le destinataire calcule $c^d$ (mod n) qui revient à
$(m^e)^d = m^{ed}$ $m^{kt+1} = (m^t)^k.m = (1)^k.m = m$ (mod n). Ainsi, le destinataire peut déchiffrer le texte chiffré $m^e$ pour obtenir le texte clair m.
La paire (n, e) forme la clé publique. Les valeurs (p, q, t, d) constituent la clé privée.

## 5. Conclusion

Cette description du réseau 802.11 nous a permis d'apprécier l'importance de cette norme puisqu'elle a apporté plus de solutions aux utilisateurs, permettant ainsi de suivre le développement de technologie dans tous les domaines. Mais, il faut bien insister sur les limites de cette norme en terme de qualité de service (QoS) et de sécurité.

Notre travail présenté dans les chapitres suivants consiste à proposer et implémenter un ensemble de mécanismes pour remédier aux insuffisances en terme de qualité de service (QoS) et de sécurité du standard original IEEE 802.11.

# IV. Conception

## 1. Introduction

L'étape de modélisation constitue une partie fondamentale pour la réalisation de tout projet. Elle consiste à bien comprendre les besoins et les respecter tout au long de la construction du projet.

Pour ce faire, on va utiliser le langage de modélisation orienté objet UML (Unified Modeling Language). Au cours de ce présent chapitre, nous nous proposons de présenter le cadre méthodologique de notre projet, d'identifier les besoins fonctionnels et non fonctionnels du système. Nous entamons ensuite la phase d'analyse objet et de la conception qui représente la partie la plus importante pour la construction de notre application.

# 2. Spécification et Analyse des Besoins

## 2.1 Analyse de l'Existant

### 2.1.1 Environnement technique

L'équipe de recherche COMET dispose, dans son parc expérimental, d'un serveur proxy (CC : Cameras Controller) qui à son tour est attaché à plusieurs caméras utilisées pour la vidéosurveillance.

L'équipe utilise VLC média player 0.8.5 pour faire du streaming multimédia. Il s'agit d'une application qui s'inscrit dans le cadre de la solution VideoLAN. Cette solution fut développée par des étudiants de l'Ecole Centrale de Paris et d'autres informaticiens du monde entier, sous licence GNU General Public Licence (GPL).
La solution de diffusion VideoLAN comprend :

- VLS (VideoLAN Server)
- VLC (à l'origine VideoLAN Client) mais qui peut être utilisé comme serveur pour la diffusion de fichiers multimédia.

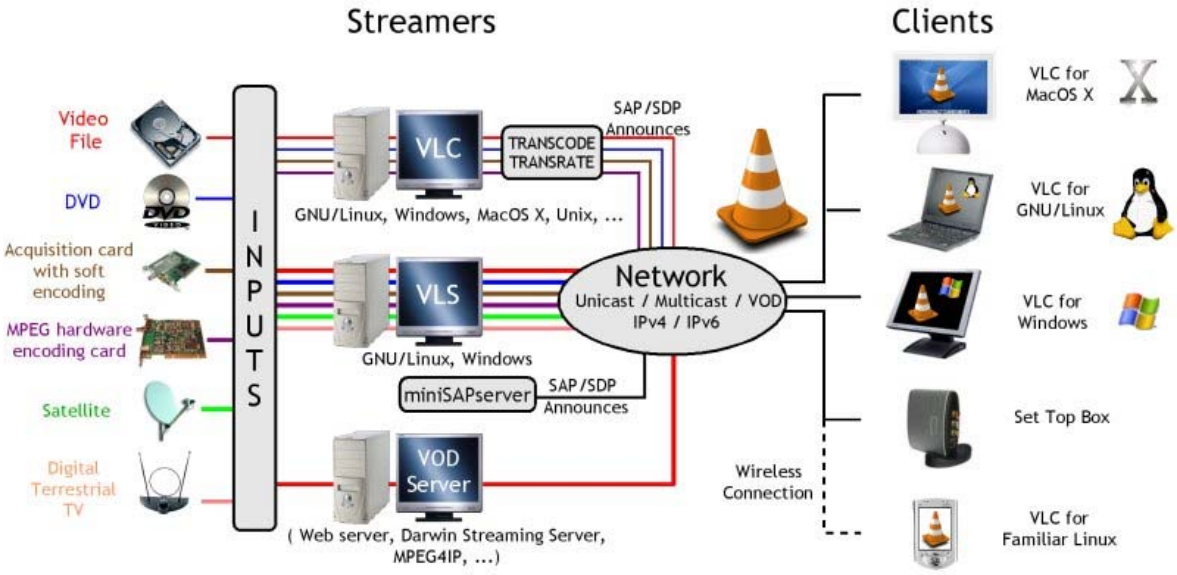


***Figure 18 : Solution de diffusion VideoLAN***

VLC permet la lecture et la diffusion multimédia. Il est capable de jouer de multiples formats audio et vidéo (MPEG-1, MPEG-2, MPEG4, DIVX, mp3, ogg…) ainsi que les DVDs, les VCDs, les CDs audio et plusieurs types de flux réseau. Il joue aussi le rôle d'un serveur de diffusion avec des capacités de transcodage (UDP unicast et multicast, http..) conçu essentiellement pour les réseaux haut débits.

VLC est multi-plateformes : il tourne sur de nombreux systèmes d'exploitation à savoir Linux, Windows, Mac OS X, BeOS, *BSD, Solaris, Familiar Linux, Yopy/Linupy et QNX.

### 2.1.2 Fonctionnement actuel

L'utilisateur sélectionne la caméra qui l'intéresse et choisit le débit de sa connexion. En fonction de ces paramètres, la diffusion du flux de la caméra se fait grâce à un serveur streaming multimédia. Un client VLC doit être installé pour que l'utilisateur puisse visualiser le flux multimédia qu'il reçoit.

## 2.2 Critique de l'existant

Bien que l'application actuelle soit fonctionnelle, elle ne répond pas aux exigences en matière de qualité de service (QoS) et de sécurité. En effet, aucun mécanisme de correction des erreurs n'est utilisé et aucun contrôle n'est effectué sur les clients susceptibles d'utiliser le service streaming. En outre, le transfert des flux n'étant pas sécurisé, toutes les données transmises peuvent être piratées à tout moment. Enfin, il est possible d'usurper l'identité du serveur proxy et attaquer ainsi les utilisateurs du site.

## 2.3 Spécification Fonctionnelle

### 2.3.1 Les besoins fonctionnels

Les besoins fonctionnels exprimés au début du projet peuvent être résumés, dans une première approche, dans les points suivants :

- Développement d'une technique FEC (Pro-MPEG) basée sur le XOR pour palier aux fluctuations aléatoires des liens sans-fil qui provoquent des pertes de paquets et, par la même, des dégradations importantes dans la qualité de la vidéo.

- Intégration des outils et mécanismes pour la sécurisation des flux vidéo à travers le déploiement d'une solution de sécurité qui soit la plus transparente possible vis-à-vis de l'utilisateur final. Le scénario de déploiement ciblé est la vidéosurveillance accessible (et contrôlable) via des terminaux WiFi.

- Implémentation d'un module de contrôle des caméras (zoom, déclinaison, rotation, etc.), par des clients distants, basé sur les signalisations RTSP.

- Implémentation d'une interface pour paramétrer le serveur de streaming au niveau du CC.

- Offrir la mobilité aux utilisateurs finaux.

### 2.3.2 Les besoins non fonctionnels

La solution doit répondre à certains critères qui même s'ils ne sont pas fonctionnels n'en demeurent pas moins importants, à savoir :

- Portabilité : Aucune restriction sur un système d'exploitation plutôt qu'un autre.
- Compatibilité avec les outils utilisés par la majorité de la clientèle cible.
- Coût minimum afin de respecter le budget alloué au laboratoire.
- Choix technologique : Open Source afin de se prémunir des éventuelles failles de sécurité.

## 2.4 Modèle de Spécification Fonctionnelle

Les spécifications fonctionnelles sont illustrées grâce à des cas d'utilisation. Les cas d'utilisation sont la réponse qu'apporte UML à la modélisation des réactions du système étudié aux différents scénarios pouvant émaner des utilisateurs.
En regardant de près les besoins recensés, on peut dégager trois modules essentiels :

- Module FEC pour protéger le flux vidéo contre les éventuelles pertes de paquets
- Module de transfert sécurisé
- Module de contrôle des caméras
- Module de paramétrage du serveur

Les principaux acteurs qui en ressortent sont :

- Le serveur de streaming (CC) dont le rôle est de proposer et diffuser le contenu multimédia
- Le client dont le rôle est de choisir le contenu voulu et le lire
- L'administrateur dont le rôle est de gérer les flux et paramétrer le serveur
- Les caméras qui présentent la source de flux à diffuser

# 3. Conception

## 3.1 Méthodologie

Pour faire la conception de notre travail, nous allons utiliser le langage de modélisation UML (Unified Modeling Language). Ce langage nous permettra non seulement de présenter les vues statiques décrivant l'application et ses objets mais également de mettre en relief les vues dynamiques incluant la composante temporelle.

## 3.2 Diagrammes des cas d'utilisation

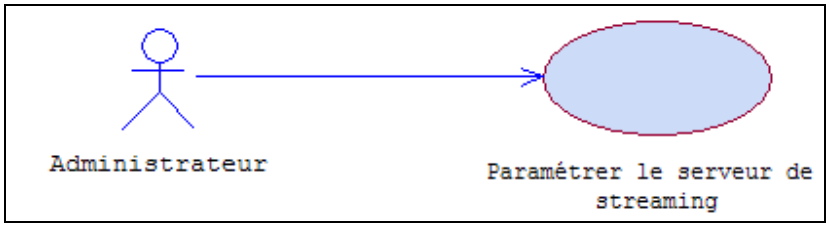


***Figure 19 : diagramme de cas d'utilisation de diffusion de flux***

Acteur : serveur
Pré-condition : diffuser un contenu multimédia
Description : Diffusion de flux des caméras vers les clients, le flux peut être avec ou sans cryptage et avec ou sans FEC
Post-condition : diffuser un contenu multimédia

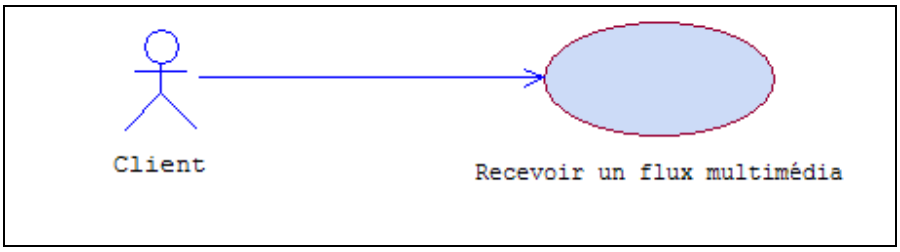


***Figure 20 : diagramme de cas d'utilisation de réception de flux***

Acteur : client
Pré-condition : visionner un flux de caméra parmi ceux proposés par le serveur
Description : Le client choisi le flux de la caméra parmi ceux proposés par le serveur, le client étant intelligent, il détecte automatiquement le type de flux (si le flux est crypté il le décrypte et si le flux est codé il le décode)
Post-condition : visionner un flux de caméra parmi ceux proposés par le serveur
Exception : seule la personne authentifiée peut accéder au site.

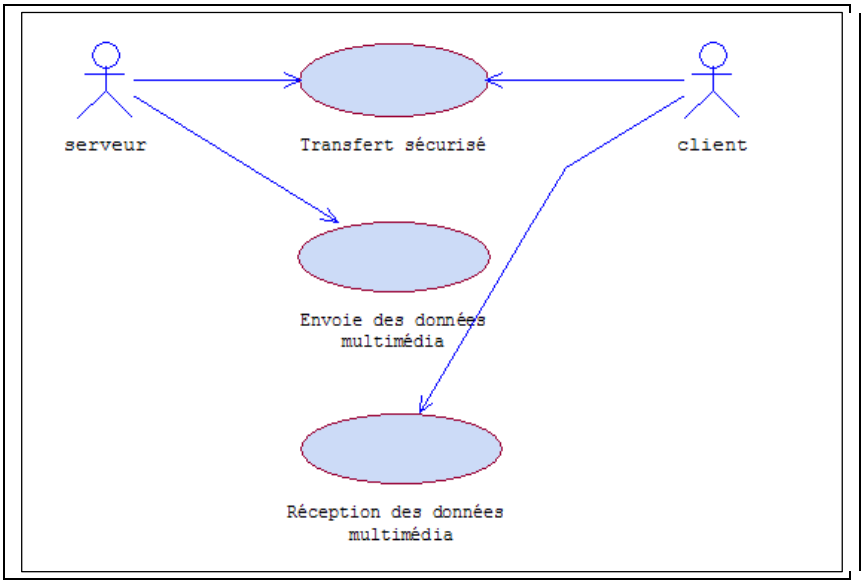


***Figure 21 : diagramme de cas d'utilisation du module transfert sécurisé***

Acteur : serveur
Pré-condition : envoyer un flux multimédia
Description : utiliser la solution pour effectuer un transfert sécurisé des données multimédia

Acteur : client
Pré-condition : visionner un flux de caméra parmi ceux proposés par le serveur
Description : utiliser la solution pour lire le flux transmis en mode sécurisé.

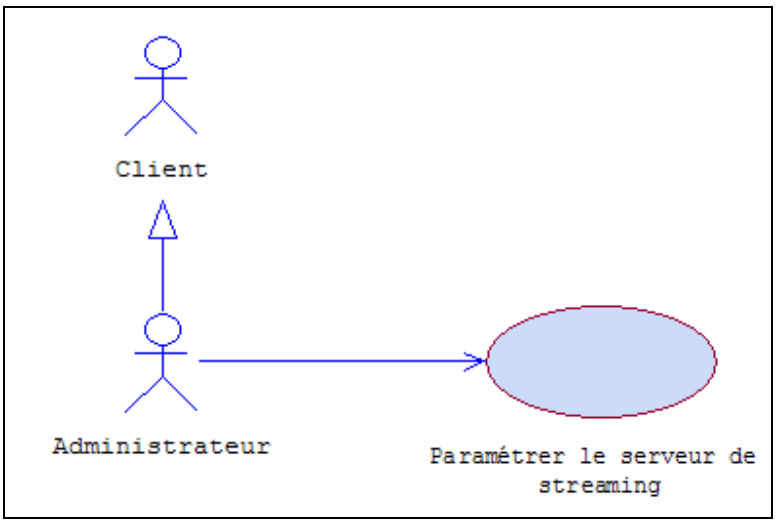

*Figure 22 : diagramme de cas d'utilisation du module paramétrage de serveur*

Acteur : administrateur
Description : paramétrer le serveur de streaming, (créer des VoD relatives aux caméras, activer ou désactiver le cryptage, activer ou désactiver la FEC, modifier les paramètres de la FEC).

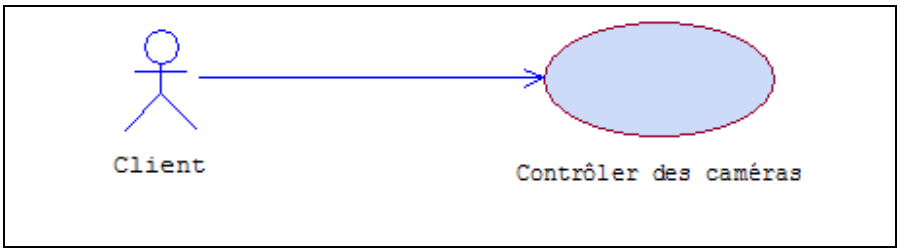


*Figure 23 : diagramme de cas d'utilisation du module contrôle de caméras*

Acteur : client
Description : contrôler des caméras a distance (zoom, déclinaison, rotation, etc.), en se basant sur les signalisations RTSP

## 3.3 Diagramme de séquences

Les diagrammes de séquence permettent d'introduire la composante temps. Le temps s'écoule de haut en bas ou de gauche à droite.

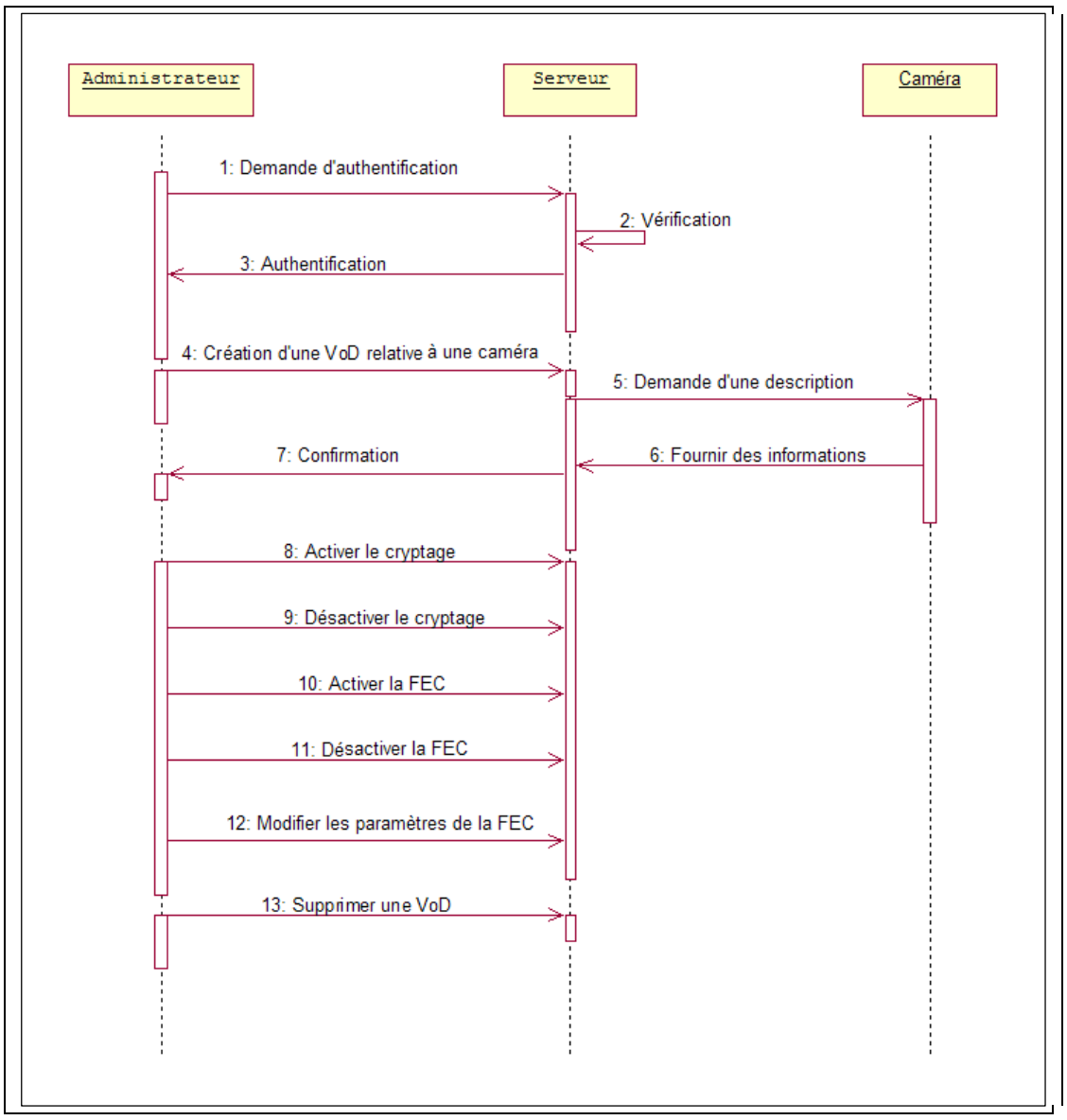


***Figure 24 : diagramme de séquence du module paramétrage de serveur***

Après authentification, l'administrateur peut créer une VoD (Vidéo à la demande) relative à une caméra, il peut ensuite activer le cryptage, ainsi le serveur crypte les données avant de les envoyer. Le client quand a lui, il détecte automatiquement que le flux est crypté, il le décrypte ensuite il l'affiche.

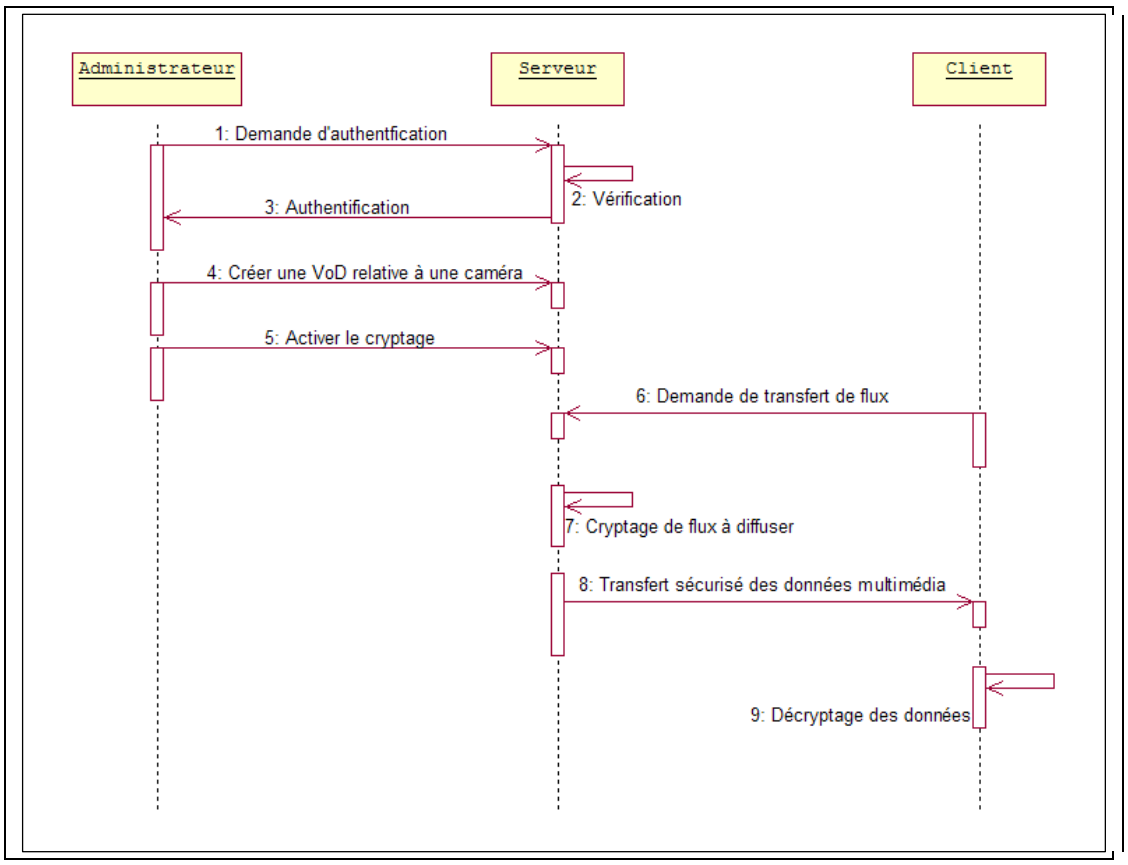


***Figure 25 : diagramme de séquences du module transfert sécurisé***

Si le cryptage est activé par l'administrateur, le serveur crypte les données avant de les envoyer au client, ce dernier décrypte le flux reçu ensuite il l'affiche.

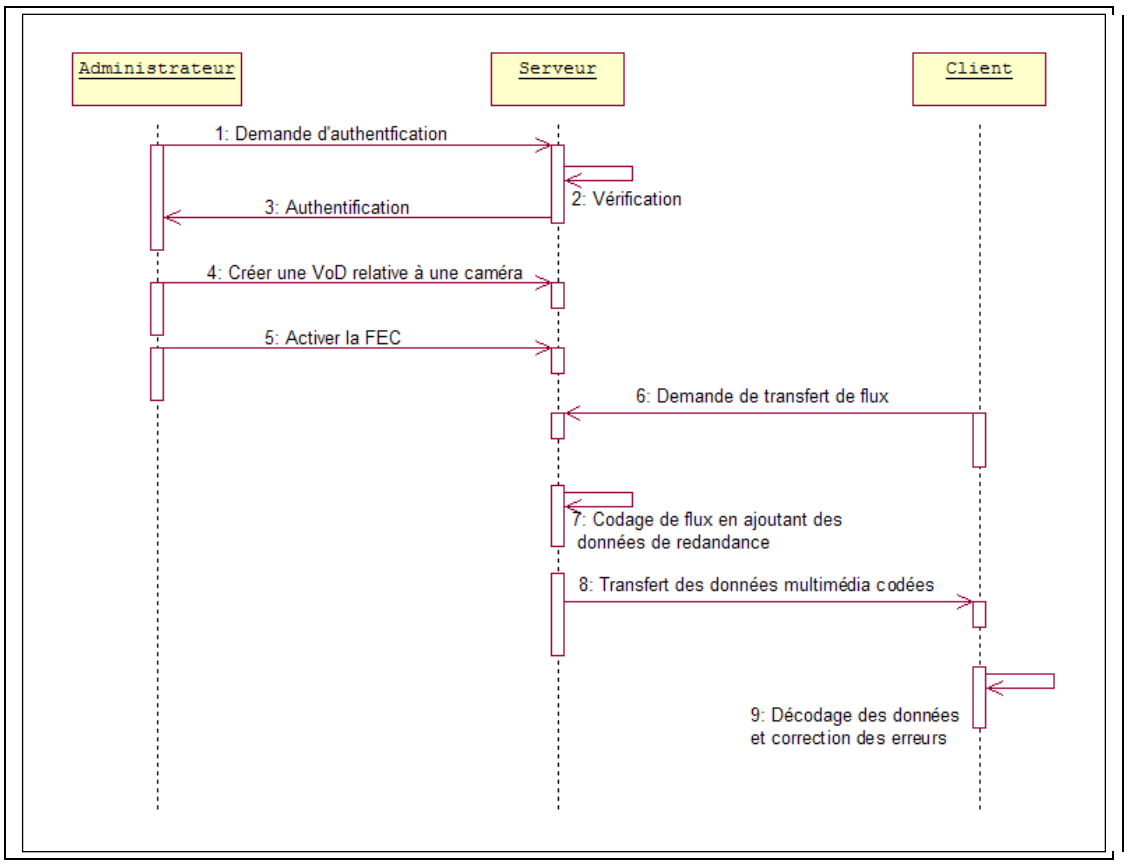


***Figure 26 : diagramme de séquences du module correction d'erreur (FEC)***

Seul l'administrateur qui peut activer la FEC, le serveur diffuse les données multimédia selon les paramètres fournis par l'administrateur lors de sa demande.
Le client décode le flux reçu, il fait les corrections possible ensuite il affiche le flux.

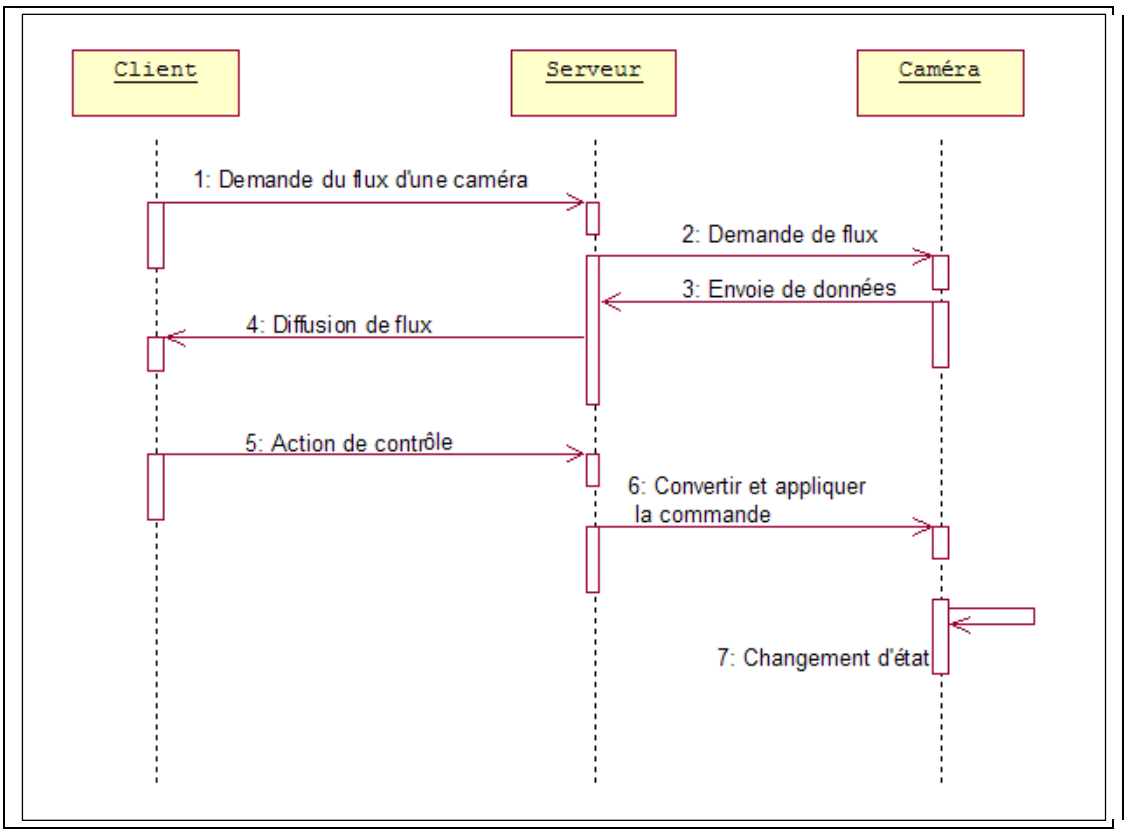


***Figure 27 : diagramme de séquences du module contrôle des caméras***

A la création d'une VoD relative à une camera, le CC interroge celle-ci et détermine son modèle, ses fonctionnalités, ainsi que son interface de contrôle (ex, commandes CGI supportés). Après cela, le CC affecte à la caméra un identificateur unique pour pouvoir par le suite l'associé à des commandes RTSP. Cet identifiant unique sera transmis avec le descripteur de la session vidéo dans le format SAP/SDP, ce qui permettra à un client donné d'envoyer des commandes de contrôle via RTSP en incluant l'identifiant unique de la caméra qu'il souhaite contrôler.

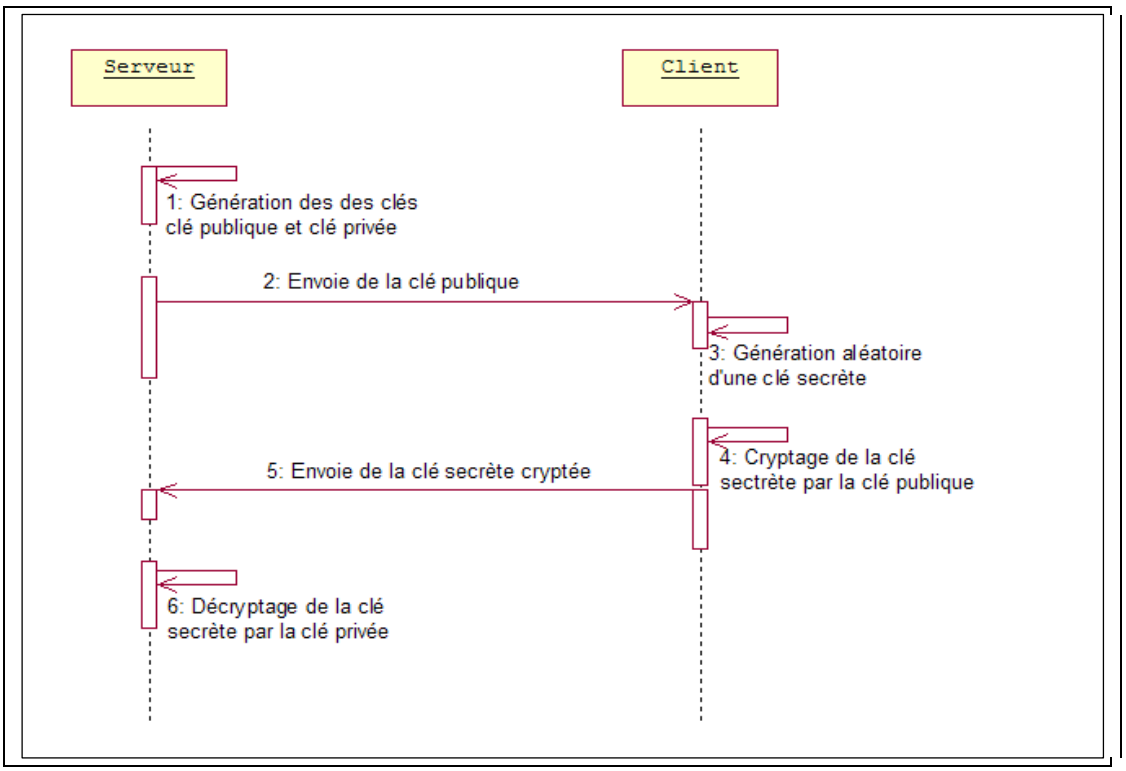


***Figure 28 : diagramme de séquences d'échange sécurisé des clés***

A chaque initialisation de connexion, le serveur envoi sa clé publique au client, ce dernier génère aléatoirement une clé secrète et il la crypte par la clé publique, ensuite il transmet cette clé secrète cryptée au serveur, celui-ci récupère la clé qui a été cryptée avec sa clé publique, la décrypte avec sa clé privée et récupère donc la clé secrète qui a été générée par le client. Les données ensuite sont cryptées par la clé secrète au niveau serveur et décryptées au niveau client.

# 4. Conclusion

L'étape de conception nous a permis de mettre en oeuvre la faisabilité de notre application à travers la détermination des principales fonctionnalités de notre système et l'évaluation des contraintes ; une étape qui s'avère très importante dans notre processus de développement vu que le reste du projet ne sera autre que d'essayer de satisfaire les besoins exprimés précédemment.

# I. Réalisation

## 1. Introduction

Dans ce chapitre, nous allons voir les étapes de mise en œuvre de la plateforme réalisée.
Nous spécifierons l'environnement de travail dans lequel nous avons opéré, les logiciels et langages de programmation utilisés pour arriver à bout de notre solution.
Nous finirons par décrire les scénarios effectués.

## 2. Environnement de travail

### 2.1 Environnement Matériel

L'architecture générale de notre plateforme de test disponible au laboratoire Labri est donnée ci-dessous (figure 29). L'architecture est constituée d'un serveur qui à son tour est attaché à plusieurs caméras réseaux capables d'encoder des flux MPEG-4. Des flux vidéo live peuvent également provenir de caméras analogiques pour être encoder en MPEG-4 au sein du serveur (encodage software ou hardware).

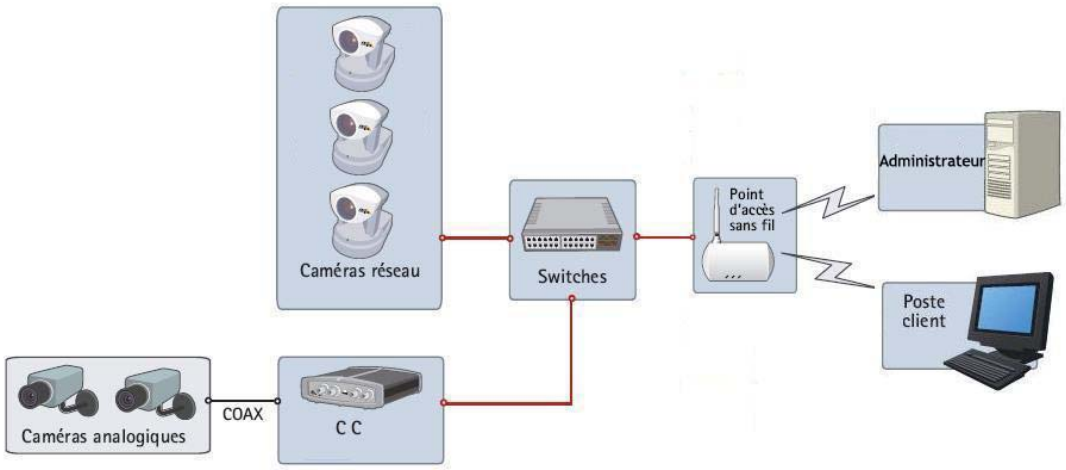


***Figure 1 : Architecture générale du système de streaming***

#### 2.1.1 Serveurs

Machines serveur ultra puissantes pour accueillir le serveur vidéo (basé sur VLC). Autres les interfaces WiFi, chaque serveur est équipé de cartes DVB-T (carte de réception de TNT – Télévision Numérique Terrestre) pour recevoir en temps réel les programmes télévisés qui sont transcodés (de MPEG-2 à MPEG-4) et puis retransmis sur

le réseau local sans fil (WiFi). Les serveur est aussi équipés de cartes d'encodage MPEG-4 (hardware encoder). Ses configurations matérielles sont les suivantes :

- Processeur Intel Xeon 3 Ghz
- RAM DDR 2 Go
- Disque dur 80Go
- Carte réseau intégrée 10/100 Mbps
- Carte sans Fil/ WiFi PC 54Mbps

Ces machines sont utilisées pour jouer le rôle de clients, administrateur et serveurs de test.

### 2.1.2 Cameras Controller (CC)

Le CC permet de migrer vers un système de vidéo sur IP en conservant les installations analogiques existantes et en leur octroyant de nouvelles fonctionnalités. Il permet par ailleurs d'éliminer certains équipements spécifiques (câbles coaxiaux, moniteurs ou enregistreurs numériques), ceux-ci devenant en effet superflus puisque les enregistrements vidéo peuvent se faire à l'aide de serveurs informatiques classiques.

Notre CC possède quatre ports analogiques pour la connexion de caméras analogiques, et cinq ports Ethernet pour la connexion au réseau. Tout comme les caméras réseau, un CC possède un serveur web intégré, une puce de compression et un système d'exploitation basé sur l'architecture PowerPC permettant la conversion des flux entrants en images vidéo numériques, ainsi que leur transmission et leur enregistrement sur le réseau informatique où elles pourront être visualisées et consultées plus facilement.

Outre ses fonctions d'entrée vidéo, un CC possède également d'autres fonctionnalités et permet de transporter d'autres types d'informations via la même connexion réseau : entrées et sorties numériques, audio, port(s) série pour les données série ou mécanismes de contrôle des mouvements en panoramique/inclinaison/zoom. Le CC permet également de connecter une multitude de caméras spécialisées.

Son rôle le plus important c'est qu'il représente un serveur proxy entre les différentes caméras et le réseau WiFi

Ses configurations matérielles sont les suivantes :

- Un processeur PowerPC
- 32 Mo de mémoire vive
- Deux cartes flash de 14Mo
- Une carte ethernet avec 5 ports
- Une carte d'acquisition de vidéo analogique

### 2.1.3 Les Caméras

Les caméras réseau ont rattrapé la technologie analogique et répondent aujourd'hui aux mêmes exigences et spécifications. Les caméras réseau ont même dépassé les

caméras analogiques en termes de performances, grâce à l'intégration d'un ensemble de fonctions avancées.

- Camera réseau (Axis 214)

La caméra réseau AXIS 214 PTZ dispose de la qualité et des performances qui la destinent à la surveillance professionnelle. Elle offre une grande souplesse d'utilisation grâce à ses fonctions de contrôle distant au plan directionnel et optique. Son zoom motorisé 18x auto-focus permet à l'utilisateur de visualiser avec une clarté exceptionnelle des objets éloignés ou de petite taille. Sa mobilité horizontale et verticale lui permettent de couvrir un champ de vision étendu.

Grâce à ses fonctions jour/nuit automatiques et à son capteur CCD de sensibilité élevée, elle permet d'obtenir des images aux couleurs irréprochables même par faible luminosité, ainsi que des images parfaitement nettes en noir et blanc, lorsque l'éclairage est dégradé. La fonction audio intégrée enrichit les options de surveillance en permettant aux utilisateurs distants non seulement de voir, mais aussi d'entendre, d'interpeller ou tout simplement de communiquer.

Ses principales caractéristiques sont les suivantes :

- Nombre max. d'images/seconde : 25
- Format d'images : JPEG, M-JPEG, MPEG-4
- Connexion réseau : RJ-45
- Support audio : duplex intégral
- Détection de mouvements
- Haute résolution 720x576 pixels (PAL)
- Capteur d'images : CCD 1/4"
- Balayage : Entrelacé
- Zoom : Pan/Tilt, zoom 18x optique et 12x numérique pilotés
- Sécurité par mot de passe

***Figure 2 : Caméra réseau : AXIS 214 PTZ***

- Camera analogique (MegaCam 1)

La MEGACAM 1 est une caméra couleur vidéo avec microphone incorporé. Le signal de la caméra peut être connecté à chaque entrée dite vidéo composite. Ce signal peut être connecté directement à l'entrée vidéo du CC, ce dernier assure le transcodage à la voler du signal vidéo avant transmission et l'ajustement de la qualité des flux vidéo (débit, taille/fréquence des images, etc.).

- Elément caméra : Capteur couleur 1/3"CMOS, PAL, objectif 64º
- Sensibilité lumineuse : 3 Lux
- Audio : microphone incorporé
- Température de service : -10º / +40º C
- Indice de protection : IP44, étanche aux projections d'eau,
- Alimentation : 12VDC, 500mA, Adaptateur secteur 230V/50Hz
- Dimensions : 130x60x72 mm

***Figure 3 : Camera Analogique : MegaCam 1***

## 2.2 Environnement logiciel

### 2.2.1 Système d'exploitation

Le système d'exploitation choisie, est le système Linux. Loin d'être arbitraire, notre choix était basé sur les différents avantages que présente ce système d'exploitation multitâche et Multi-Utilisateurs ; en effet, Linux est avant tout un logiciel libre, au code ouvert, qui a su frayer sa place sur les marchés des serveurs et de plus en plus sur les postes clients. Il est puissant, permettant de tirer profit au maximum des ressources de la machine, en réduisant notamment son utilisation propre des moyens systèmes pour laisser l'avantage aux programmes installés par l'utilisateur. Il est robuste, dans le sens, qu'il ne demande pas à être redémarré chaque fois qu'il y'a une modification des paramètres, comme est le cas pour d'autres systèmes. D'autre part, un poste sous linux, pourrait très

bien fonctionner tout le temps sans nécessiter un redémarrage ce qui est primordial pour une machine jouant le rôle d'un serveur. Finalement, la gratuité et disponibilité d'une multitude de versions ne laisse pas de doute compte à l'utilisation de ce système.

La version adaptée, est une Fedora Core 5, vu sa convivialité et le panorama d'outils inclus dans la distribution de base, à ne citer que le gestionnaire de packages YUM.

### 2.2.2 Serveur de streaming

Pour jouer le rôle du serveur de streaming, nous avons opté pour la solution VideoLanClient (VLC). C'est un lecteur multimédia très robuste permettant de lire une très grande variété de formats audio et vidéo ce sans téléchargement de codecs externes. En plus, VLC permet la diffusion sur réseau ou sauvegarde de flux en HTTP, HTTPS, UDP, RTP…

VLC est un logiciel multi-plateformes, fonctionnant sous Windows, MacOS X, GNU/Linux, BSD, un grand nombre d'Unix, sous Familiar Linux et Zaurus (Linux pour PDA).

Cette portabilité ajoutée à l'ouverture du code et la gratuité du programme, sans oublier la performance affirmée et la documentation disponible, font de VLC un très bon choix pour assurer les fonctions du serveur de streaming que nous voulons intégrer. Nous utiliserons la dernière version de VLC présente sur leur site officiel [18] et qui est la version 0.8.5.

### 2.2.3 Client de streaming

Pour le client, on utilisera pour les mêmes raisons citées ci haut, VLC.

### 2.2.4 Méthode de cryptage

L'AES, comme les autres mécanismes de chiffrement symétrique, a l'avantage d'être facile à implémenter et moins coûteux en temps de calcul, c'est pourquoi il est privilégié dans le chiffrement des données. Cependant, l'utilisation de la cryptographie symétrique et donc d'une clé de cryptage symétrique nécessite que les deux correspondants, dans notre cas le serveur et le client, se partagent cette clé avant l'échange des données. Ceci est un problème délicat dans la mesure où la sécurité du système de cryptage repose sur le secret de cette clé.

Pour des contraintes de rapidité et de ressources, nous avons utilisé l'AES-128 pour crypter et décrypter les paquets. Ce paramètre reste toujours paramétrable.

Pour un échange sécurisé des clés on a choisi l'algorithme RSA vu sa simplicité.

### 2.2.5 Langage de programmation

Le langage utilisé est le langage C/C++ et ce, initialement parce que le serveur et le client VLC sont écrits avec, puis à cause de sa portabilité et sa nature proche du langage machine, ce qui constitue un avantage parce que la solution va être porter sur un système embarqué (CC).

# 3. Exposition du Travail Réalisé

## 3.1 Installation de VLC

La mise en place du serveur de streaming de VLC, n'a pas été une tâche facile.
Il fallait installer VLC à partir des sources et y intégrer les modules dont nous avons besoin et qui ne se trouvent pas dans les versions pré compilées.
La nature modulaire de VLC et le fait qu'il soit bâti autour d'un core rattaché aux différents modules qui constituent la plateforme par des liens externes, facilite la tâche de programmer les modules qui seraient pris en main par différentes équipes de développement, mais pose des problèmes de versions et de compatibilité lors de l'installation. C'est ainsi qu'avoir une version VLC qui tourne du premier coup est loin d'être évident, et c'est pourquoi nous décrirons ci-dessous brièvement la procédure d'installation de VLC, en rapportant les problèmes qu'on a eus et la manière de les contourner. Un tutorial complet qu'on a élaboré, sera joint en annexe.

### 3.1.1 Les modules

VLC utilise une structure modulaire. Le noyau contrôle principalement la communication entre les modules. Tout le traitement multimédia est fait par les modules : Il y a des modules d'entrée, des démultiplexeurs, des décodeurs, des modules d'affichage vidéo, ...

Dans le fichier de configuration du VLC il y'a quelques modules qui sont activés ou désactivés par défaut. Si nous voulons utiliser un module qui est marqué désactivé par défaut, nous devrons lancée le script de configuration du VLC avec la commande suivante :

```
./configure --enable-nom_du_module
```

A l'inverse, si nous voulons désactiver un module qui est activé par défaut, on doit utiliser le script :

```
./configure --disable-nom_du_module
```

- Modules d'entrée
  Ces modules permettent à VLC de lire depuis différentes sources. VLC essaie de choisir le module le plus adapté au moment de la lecture

- Démultiplexeurs
  Dans un flux, les signaux vidéo et audio sont toujours inclus dans un format "conteneur". Les démultiplexeurs extraient les flux de ce conteneur et les envoient aux décodeurs.

- Décodeurs

Ces modules permettent à VLC de supporter de nombreux codecs (formats de compression).

- Module de sortie vidéo
  Les modules de sortie vidéo permettent d'afficher de la vidéo sur votre écran.

- Modules de sortie audio
  Ces modules permettent de choisir le système de sortie du son.

- Modules d'interface
  Ces modules permettent de choisir une ou plusieurs interfaces (interface graphique ou interface de contrôle).

### 3.1.2 Installation depuis la source

L'installation du VLC depuis la source, nécessite l'installation de quelques librairies :

- libdvbpsi (obligatoire),
- mpeg2dec (obligatoire),
- libdvb : Pour pouvoir diffuser depuis une carte DVB (satellite ou télévision numérique terrestre).
- ffmpeg, libmad, faad2 pour lire des fichiers MPEG 4/DivX
- libogg & libvorbis pour pouvoir lire des fichiers Ogg/Vorbis.

Pour toutes les librairies, nous avons fait la décompression du fichier source, compilation, configuration (sauf pour libdvb qui n'a pas de ./configure), compilation et installation (procédure standard) :

```
gunzip source.tar.gz (bzip2 source.tar.bz2)
tar -xvf source.tar
cd source
./configure
make
make install
```

Après l'installation des librairies requises, la phase finale est l'installation et la configuration du VLC, nous activons les options que nous aurons besoin.

Décompression :

```
tar xvzf vlc-version.tar.gz
cd vlc-version
```

Pour afficher la liste des options de configuration :

```
./configure -help
```

Configuration :

```
./configure --with-ffmpeg-tree=/home/installerVLC/ffmpeg-0.4.9-pre1/  --enable-faad --with-faad-tree=/home/installerVLC/faad2-20040923/ --enable-dvb -enable-dvbpsi
```

Puis, phase d'installation

```
make
make install
```

Après une installation réussie, on obtiendra deux interfaces possibles, selon qu'on appelle le système par la commande svlc, qui charge l'inteface skins2, ou wxvlc qui charge celle de widgets. On pourra voir ces deux interfaces respectivement dans les deux figures qui suivent.


***Figure 4 : Interface skins2 de VLC***

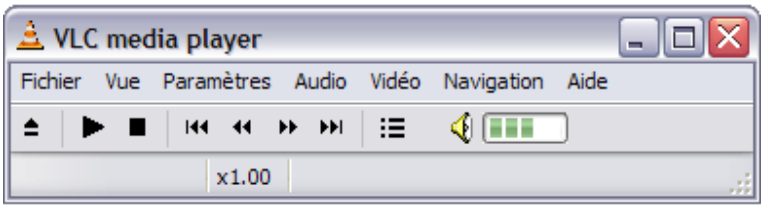


***Figure 5 : Interface wxwindows de VLC***

Pour lancer le serveur RTSP, on peut utiliser le mode commande, et entrer la ligne qui suit :

```
vlc -vvv input_stream -sout '#rtp{rtsp://239.255.1.1}'
```

- input_stream fait référence au flux à jouer.

- sout : Active la diffusion d'un stream audio élémentaire et d'un stream élémentaire vidéo.

ou graphiquement en utilisant l'assistant diffusion/transcodage

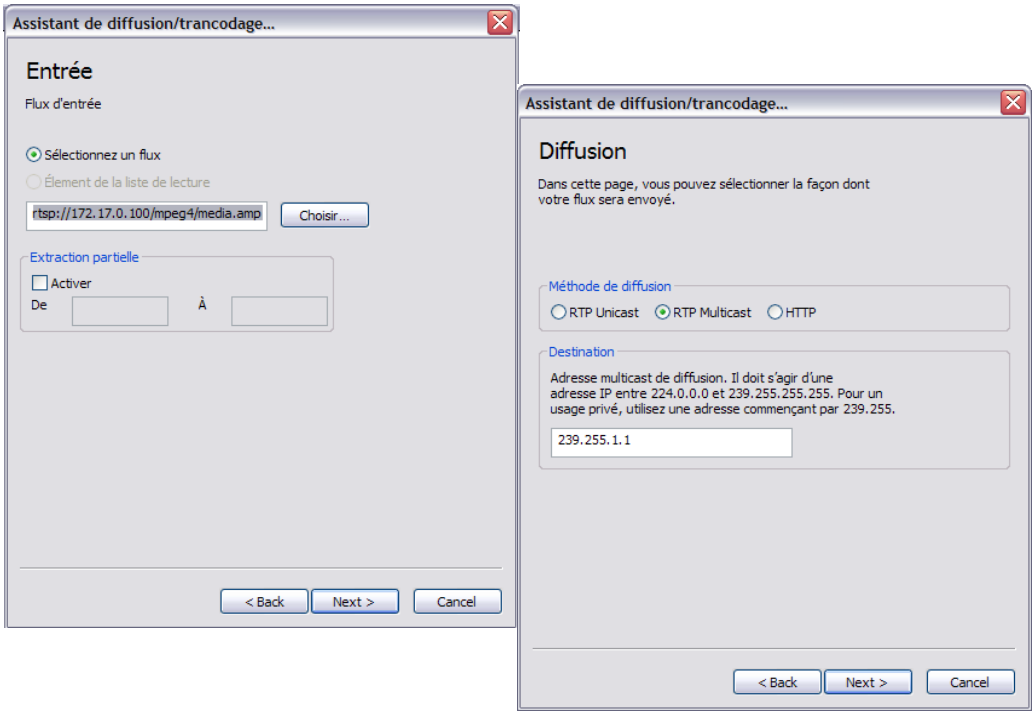


***Figure 6 : Lancement d'un serveur RTSP en mode graphique***

Un client RTSP peut recevoir un flux soit par la ligne de commande, comme suit :

```
vlc rtsp://172.17.0.2/axis214
```

soit graphiquement, comme illustré dans la figure suivante.

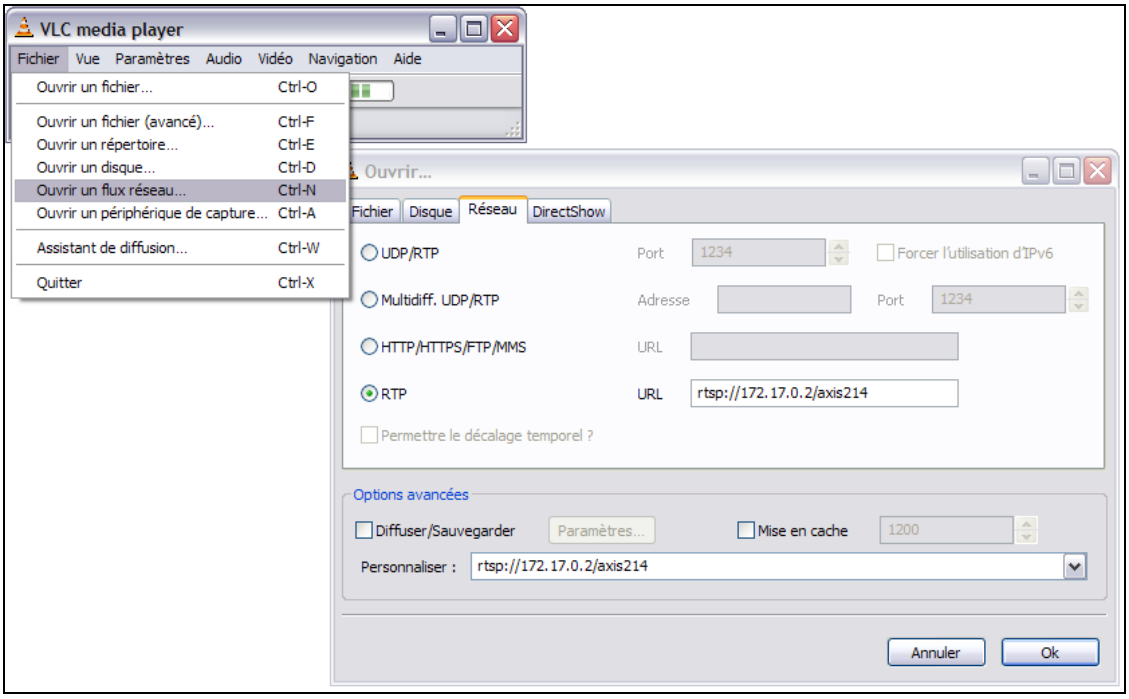


***Figure 7 : Etapes du lancement du client RTSP en mode graphique***

## 3.2 Module FEC

Le principe du FEC est de générer régulièrement des paquets de redondance appelés plus loin paquets FEC. A chaque paquet FEC correspond plusieurs paquets de données. Il permet de reconstruire un et un seul paquet qui lui est associé qui se serait perdu. Le principe est d'additionner les paquets de données en utilisant les règles du XOR (ou exclusif). L'avantage du XOR est que l'opération est réversible. Ainsi, après avoir additionné les paquets de données afin d'obtenir le paquet FEC, il suffit, une fois la perte d'un paquet constatée, d'additionner les paquets restant et le paquet FEC pour obtenir le paquet perdu. Cette méthode est appelée par la suite Simple_FEC.

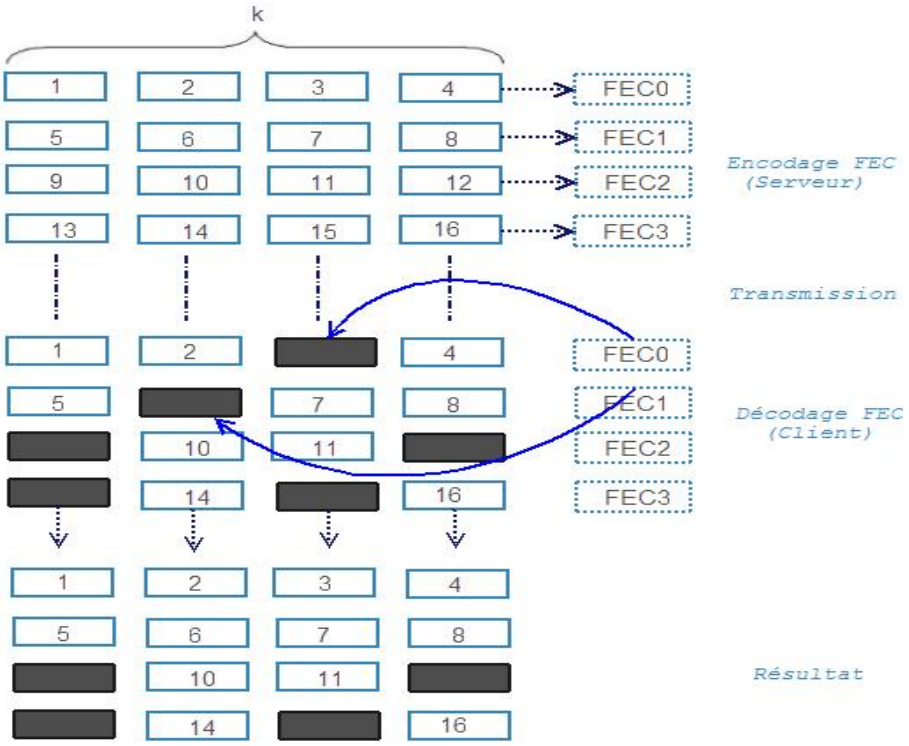


***Figure 8 : Principe de Simple_FEC***

L'inconvénient est que ce principe ne permet de récupérer qu'un seul paquet par paquet FEC. Si deux paquets de données associés au même paquet FEC sont perdus, ils ne pourront pas être reconstruits. Pour pallier à ce problème, il est nécessaire de générer plusieurs paquets FEC par paquets de données. Pour cela, nous schématisons les paquets de données par une matrice et nous créons un paquet FEC par ligne et par colonne, comme montré sur la figure ci-dessous.

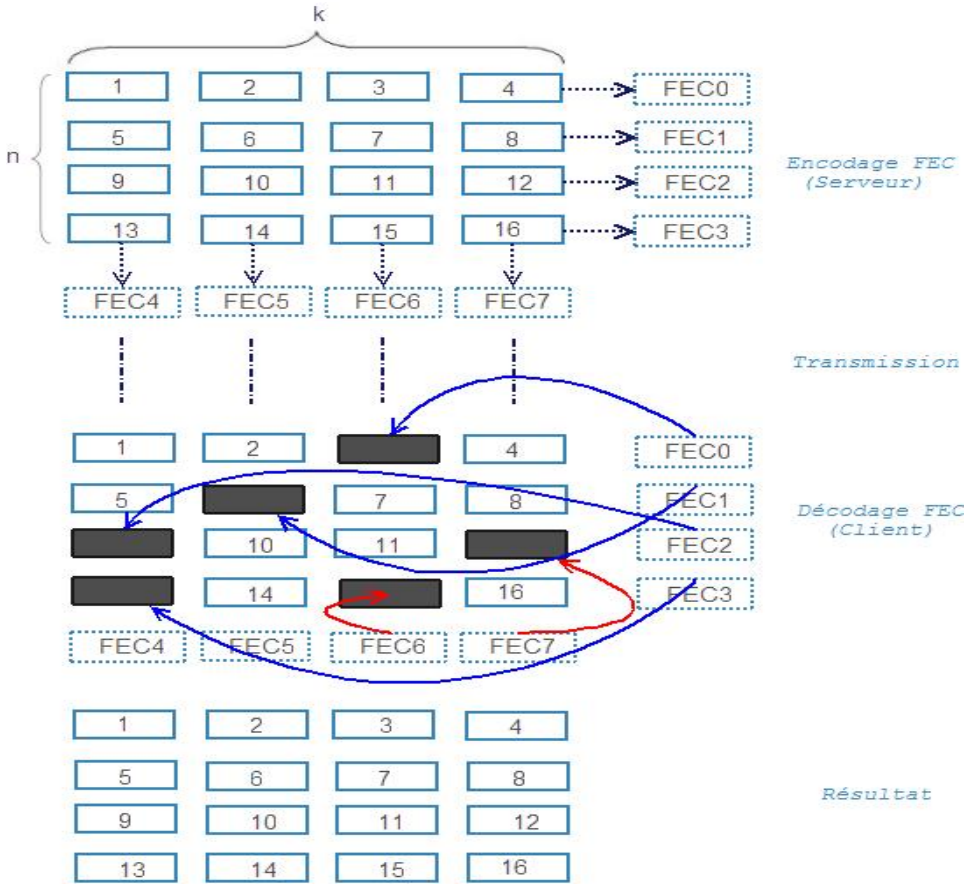


***Figure 9 : Principe de Pro-MPEG***

Cette technique (appelée par la suite Pro-MPEG) consiste à organiser d'une certaine manière les paquets vidéo avant la transmission au niveau du serveur (CC). Génération des paquets FEC (encodage FEC) puis la transmission du tout. Au niveau du récepteur, les pertes de paquet sont identifiées avant la reconstitution des blocs de paquets vidéo, s'il y'a perte, on procède au décodage FEC. Le serveur doit continuellement signaler aux récepteurs certains paramètres FEC afin de pouvoir recouvrir les pertes (décodage FEC) convenablement. Les paramètres en question sont la taille du bloc, l'identification de chaque paquet vidéo dans le bloc, n, k etc. Ce modèle permet de protéger chaque paquet RTP par deux paquets FEC.

### 3.2.1 Implémentation côté serveur

Au niveau du serveur, lors de l'envoi des paquets RTP, nous enregistrons ceux-ci et nous les additionnons via les règles du XOR. Lorsque touts les paquets relatifs à un paquet FEC ont été additionnées, un en-tête spécial est créé pour le paquet FEC et nous lui adjoignons les données générées par le XOR. Le paquet FEC est alors envoyé juste après l'envoi du dernier paquet de données correspondant au paquet FEC.

### 3.2.2 Implémentation côté client

Au niveau du client, un petit algorithme permet de reconnaître les paquets FEC parmi les paquets RTP classiques, grâce aux en-têtes spéciaux des paquets FEC créés au niveau du serveur. Si il y'a perte, on procède au décodage FEC.

Le décodeur FEC parcourt successivement les lignes et les colonnes d'un rectangle FEC tant qu'il est possible de corriger des paquets. Le décodage s'arrête quand tout a été corrigé ou lorsqu'il n'est plus possible de corriger de paquet.
On peut résumer le fonctionnement du décodeur FEC par le pseudo code suivant

```
Données : Une matrice Mat représentant le rectangle FEC à corriger, v
le nombre de paquets initialement perdus dans ce rectangle.
Résultat : Le rectangle FEC totalement ou partiellement réparé.

chgt = 1
colonnes_deja_visitees = 0
TantQue (v > 0 et chgt ≠ 0) faire
|     chgt = 0
|     /* Parcours des lignes */
|     num_ligne = 0
|     TantQue (num_ligne < n et v ≠ 0) faire
|     |     dp = NbErreursLigne(num_ligne)
|     |     Si (dp = 1) alors
|     |     |     Correction des erreurs sur la ligne num_ligne
|     |     |     chgt = 1
|     |     |     v = v - dp
|     |     Fin Si
|     |     num_ligne = num_ligne + 1
|     Fin TantQue
|     /* Parcours des colonnes */
|     Si (chgt = 1) ou (chgt = 0 et colonnes_deja_visitees = 0) alors
|     |     num_colonne = 0
|     |     TantQue (num_colonne < k et v ≠ 0) faire
|     |     |     dm = NbErreursColonne(num_colonne)
|     |     |     Si (dm =1) alors
|     |     |     |     Correction des erreurs sur la colonne num_colonne
|     |     |     |     chgt = 1
|     |     |     |     v = v - dm
|     |     |     Fin Si
|     |     |     num_colonne = num_colonne + 1
|     |     Fin TantQue
|     |     colonnes_deja_visitees = 1
|     Fin Si
Fin TantQue
Retourner Mat
```

### 3.2.3 Tests et résultas

Dans cette section nous allons tester les solutions de correction d'erreurs (Simple_FEC et Pro_MPEG) que nous avons implémenté. Nous allons comparer les résultats de nos solutions avec la solution existante. Ainsi nous allons faire une série de tests avec différent taux de perte.

Les premiers testes effectués sur notre système avaient pour but de démontrer que nos solutions s'intègrent d'une manière complètement transparente. Pour cet effet, nous avons effectué un premier test en utilisant un taux de perte nul. Les vidéos reçues au niveau client dans tous les scénarios sont pratiquement identiques à part un léger délai aperçu dans Simple_FEC (scénario 2) et le ProMPEG (scénario 3). Ceci peut être expliqué par le buffer de paquets ajouté aux niveaux de nos deux solutions. Ce buffer est nécessaire pour regrouper les paquets qui vont être utilisés par la suite dans la phase de régénération de paquets en cas de perte. Cependant, ce délai est pratiquement imperceptible et n'influence en aucun cas la qualité de la vidéo qui est identique dans tous les scénarios. La figure 38 donne un exemple d'image capturée à partir des vidéos reçus du coté client dans tous les scénarios. Nous voyons clairement que la qualité d'image est identique.

***Figure 10 : Qualité de la vidéo reçue  pour différent scénario (taux de perte nul)***

Après avoir validé que nos deux solutions étaient transparentes dans le cas où les taux de pertes dans le réseau sont nuls. Le deuxième test avait pour objectif de démontrer la valeur ajouter de nos deux systèmes, c'est-à-dire, améliorer la qualité de la vidéo avec la présence des taux de perte dans le réseau. Pour cela, nous avons fixé le taux de perte à 5% en utilisant « netem » et nous avons refait le test de la transmission pour tous les scénarios. Le choix du taux de perte a été effectué suivant le taux de redondance utilisé dans nos systèmes.
Dans ce cas, nous avons enregistré une différence notable entre la solution originale et nos deux solutions. Dans le premier scénario (solution originale), le taux de perte utilisé a eu un effet néfaste sur la qualité de la vidéo reçu au niveau du client. La perte de paquets au niveau réseau introduit des pertes de bloque au niveau image. Ceci dégrade clairement la qualité de la vidéo perçue au niveau du client dans le scénario 1. Cependant, cet effet n'est pas perçu dans les deux derniers scénarios qui représentent nos deux systèmes de corrections d'erreur. Nos deux solutions ont pu régénérer les paquets perdus dans le réseau grâce au système de correction d'erreur. La régénération des paquets perdus permet de restituer les blocs d'images et ainsi la qualité de la vidéo perçue par l'utilisateur n'est pas détériorée. Un exemple d'image est donnée dans la figure 39, nous constatons clairement la dégradation de la qualité d'image dans le scénario 1 dut aux pertes des paquets dans le réseau tandis que les images restituées dans le scénario 2 et 3 sont identiques à l'original.

***Figure 11 : Qualité de la vidéo reçue  pour différent scénario (taux de perte=10%)***

Pour confirmer cette constatation, nous avons conduits un autre test en augmentant les taux de pertes. Dans ce dernier test les pertes ont était fixées à un taux supérieur au taux

de redondance des deuxièmes et troisièmes scénarios. Les résultats obtenus confirment les résultats du deuxième test pour le scénario 1 qui se retrouve complètement dépassé par les pertes de paquets jusqu'au point ou l'affichage de la vidéo se fige. Ceci est expliqué par le fait que le taux de perte est tellement important que des images complètent sont perdu. Cependant, nous constatons une différence entre les deux derniers scénarios.
La qualité de la vidéo dans le scénario 2 est affectée par la perte de blocs d'image qui est dut à la perte de paquet. En effet, le taux de perte étant important, notre solution Simple_FEC n'arrive pas à régénérer tous les paquets perdus, elle a atteint ses limites et ceci est visible sur la qualité de la vidéo. Un exemple d'image est donné dans la figure 40.
En ce qui concerne la solution ProMPEG, la vidéo reçue n'est affectée par aucune perte de paquet et la qualité d'image est presque identique à l'originale. La capacité de notre système à régénérer les paquets perdus dans le scénario 3 est plus grande comparée au scénario 2 malgré l'utilisation du même taux de redondance. Ceci est expliqué par le système de correction qui utilise la matrice et qui peut régénérer plus d'un paquet perdu sur une ligne ou une colonne en utilisant une correction itérative.

***Figure 12 : Qualité de la vidéo reçue  pour différent scénario (taux de perte=20%)***

## 3.3 Module de transfert sécurisé

Afin d'alléger la charge du cryptage en terme de consommation de ressources au niveau du serveur et des clients, nous n'effectuons le cryptage que sur une partie (192 octets) du paquet pour rendre le décodage du flux impossible, mais le code a été fait de façon à pouvoir paramétrer cette valeur.

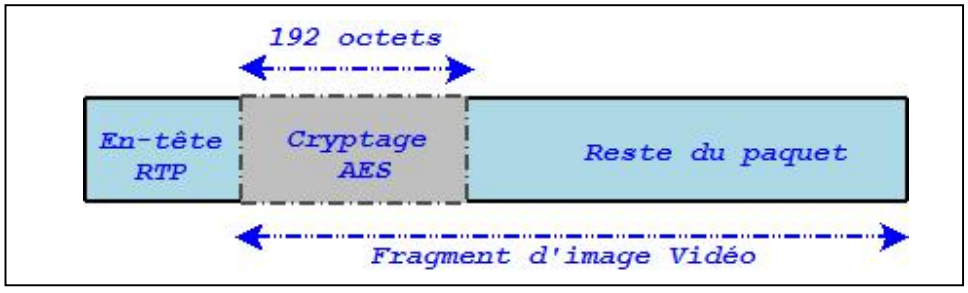


***Figure 13 : Cryptage AES des paquets RTP***

De la même manière, les flux de contrôle de session comme les paquets SAP/SDP et RTSP sont crypté avant envoi sur le réseau sans fils. Le nombre d'octets à crypté est également le même (192 octets).

### 3.3.1 Implémentation côté serveur

Pour le cryptage des données par le serveur, nous avons modifié le module *access_output* qui assure l'envoi du flux par le serveur. En effet, nous avons choisi d'appliquer le cryptage AES sur les paquets RTP juste avant qu'ils soient envoyés sur le réseau.

Après avoir récupéré et testé la librairie de L'AES-128, nous avons implémenté une fonction *crypt()* qui effectue le cryptage des paquets RTP par blocs de 16 octets comme le montre la figure suivante.

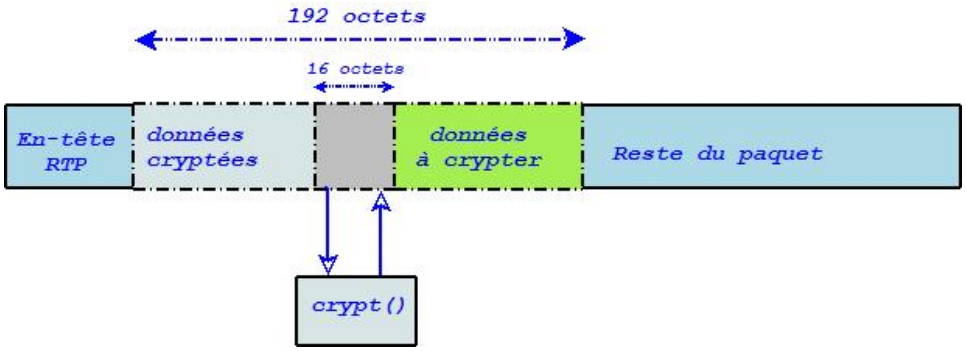


***Figure 14 : Principe du cryptage par blocs des paquets RTP***

### 3.3.2 Implémentation côté client

Pour le client, nous avons modifié le module live où se passe la réception des paquets RTP.
Tout comme pour le serveur, nous avons créé une fonction *decrypt()* qui décrypte la partie cryptée du paquet. Le décryptage est fait de la même façon que le cryptage, c'est-à-dire directement sur le paquet reçu.

### 3.3.3 Echange sécurisé des clés

Après l'implémentation de l'algorithme RSA, le client et le serveur s'échangent de manière sécurisée la clé secrète de cryptage.
La figure suivante montre quelques échanges RSA qui ont été implémentées au niveau des commandes RTSP.

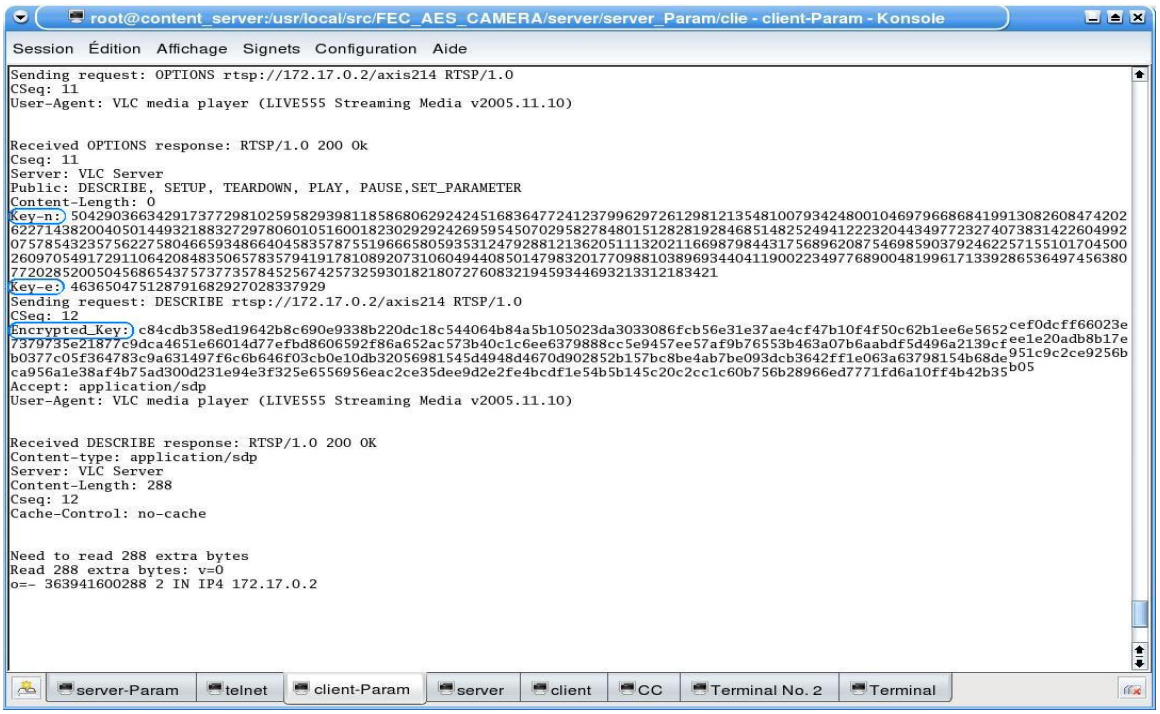


***Figure 15 : Echange sécurisé des clés***

### 3.3.4 Données interceptées dans le réseau

Ce cas de figure représente le cas où les données cryptées par le serveur sont interceptées sur le réseau par une personne tierce non autorisée.

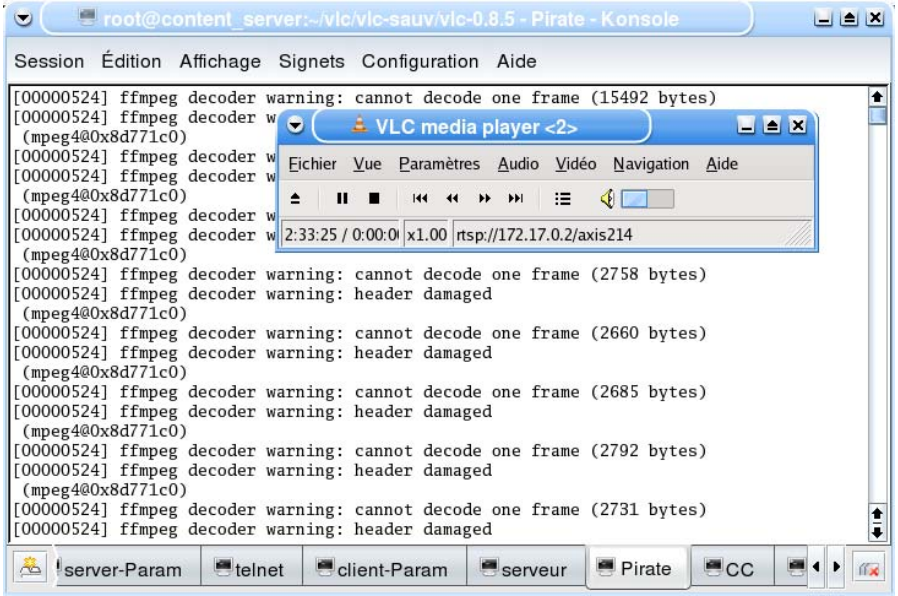


***Figure 16 : Données interceptées dans le réseau***

Le client représentant la tierce personne reçoit les paquets RTP mais n'arrive pas à les décrypter. Il lui est donc impossible d'afficher le flux multimédia.

## 3.4 Module de contrôle des caméras

L'objectif de ce module est d'intégrer des fonctionnalités de contrôle au niveau de l'interface utilisateur du client. Des commandes de contrôle seront désormais envoyées depuis le client vers le CC pour être interprétés et répercutés au niveau des caméras concernées.

Pour avoir une grande liberté dans le contrôle de la caméra, nous avons choisi la requête SET PARAMETER du protocole RTSP pour envoyer les commandes du client au CC. Ainsi chaque mouvement de camera devra être décrit par un nom et une valeur.

| Variables | Min | Max |
|---|---|---|
| PAN | -100 | 100 |
| TILT | -100 | 100 |
| ZOOM | 0 | 1000 |
| FOCUS | 0 | 1000 |
| IRIS | 0 | 1000 |
| AUTOFOCUS | – | – |
| AUTOIRIS | – | – |
| MODE | Day | Night |
| RAZ | – | – |

***Table 1 : Paramètres de contrôle des caméras***

Ainsi si le client envoie la requête :

```
SET_PARAMETER rtsp:/172.17.0.2/axis214 RTSP/1.0
ZOOM : 600
```

Le CC répondra par

```
RTSP/1.0 200 OK
```

Et la caméra positionnera son zoom a 60% de son maximum.

Pour traduire (au niveau du CC) les commandes de contrôle RTSP en commandes CGI à destination des caméras, nous avons créé un dictionnaire en utilisant XML.

```
<?xml version="1.0"?>
<!DOCTYPE CameraModuleConfiguration SYSTEM "cameras.dtd">
<CameraModuleConfiguration>
<camera name="axis214">
        <!-- Settings for camera test -->
        <Location ip="172.17.0.100"/><!-- Or DNS attribute-->
        <Pan Min="-100" Max="100"><![CDATA[/axis-cgi/com/ptz.cgi?camera=1&pan=]]></Pan>
        <Tilt Min="-100" Max="100"><![CDATA[/axis-cgi/com/ptz.cgi?camera=1&tilt=]]></Tilt>
        <Zoom Min="0" Max="1000"><![CDATA[/axis-cgi/com/ptz.cgi?camera=1&zoom=]]></Zoom>
        <Center><![CDATA[/axis-
cgi/com/ptz.cgi?camera=1&imagewidth=&imageheight=480&center=?]]></Center>
        <RaZ><![CDATA[/axis-
cgi/com/ptz.cgi?camera=1&pan=0&tilt=0&zoom=0&autofocus=on&autoiris=on&ircutfilter=on]]></RaZ>
        <AutoFocus><![CDATA[/axis-cgi/com/ptz.cgi?camera=1&autofocus=on]]></AutoFocus>
        <Focus Min="0" Max="1000"><![CDATA[/axis-cgi/com/ptz.cgi?camera=1&focus=]]></Focus>
        <AutoIris><![CDATA[/axis-cgi/com/ptz.cgi?camera=1&autoiris=on]]></AutoIris>
        <Iris Min="0" Max="1000"><![CDATA[/axis-cgi/com/ptz.cgi?camera=1&iris=]]></Iris>
        <IRvision><![CDATA[/axis-cgi/com/ptz.cgi?camera=1&ircutfilter=]]></IRvision>

</camera>

...

</CameraModuleConfiguration>
```

Maintenant il nous reste que lancer le client et utiliser l’interface étendue du vlc comme le montre la figure suivante.

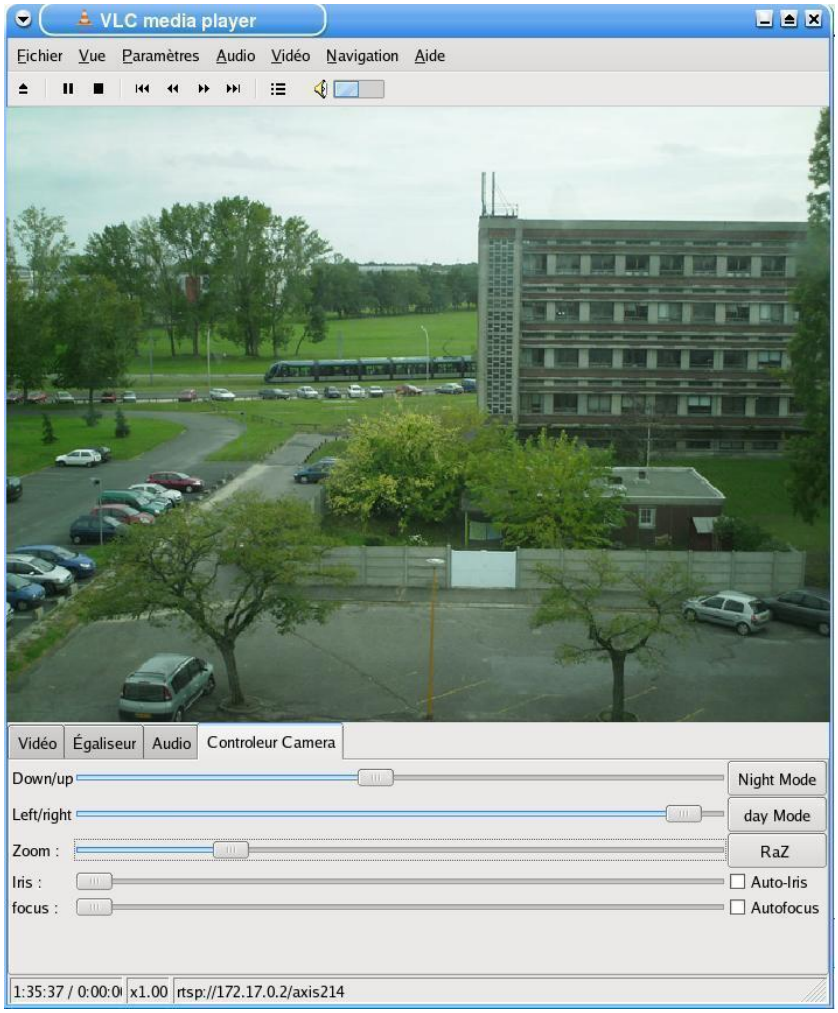


***Figure 17 : Interface de contrôle des caméras***

## 3.5 Module de paramétrage du serveur

Seul l'administrateur qui peut paramétrer le serveur (CC), il se connecte via l'interface telnet de ce dernier et après authentification il peut faire les configurations souhaitées.


***Figure 18 : Authentification de l'administrateur***

L'administrateur peut créer des VoD (Vidéo à la demande) relatives à des caméras (analogique et numérique).

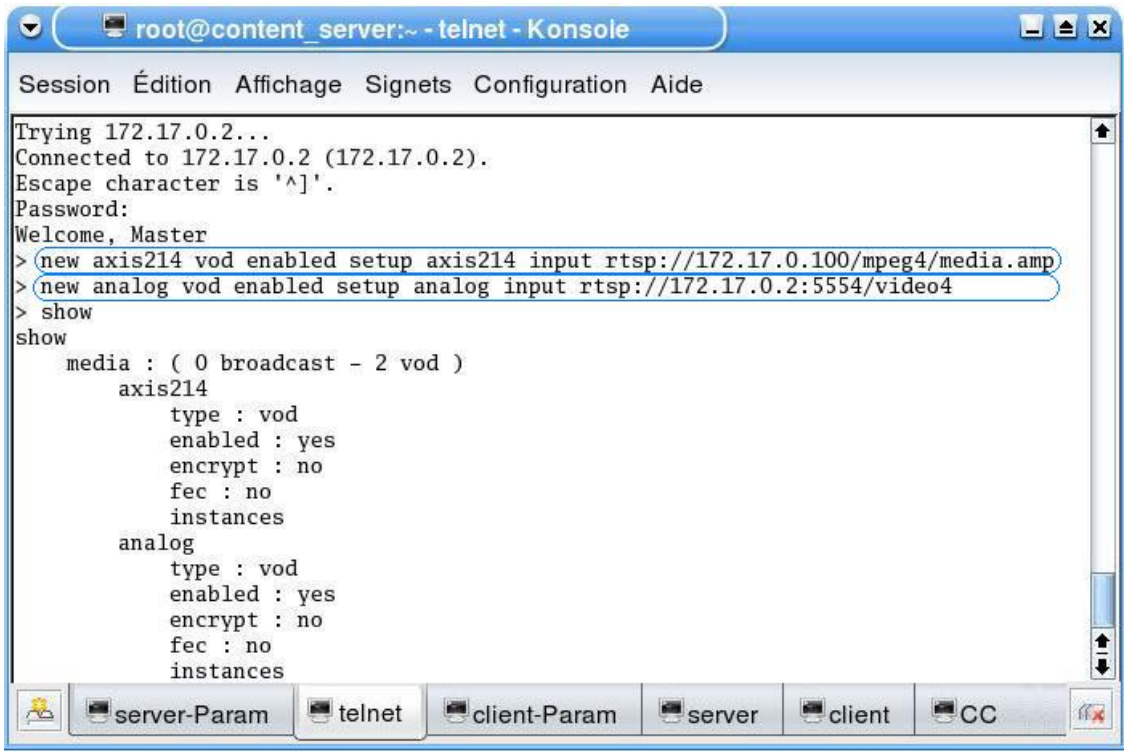


***Figure 19 : Création des VoD***

Une fois les VoD sont crées, l'administrateur peut activer le cryptage et la FEC qui sont par défaut désactivés, il peut également choisir le type de la FEC et modifier ses paramètres.

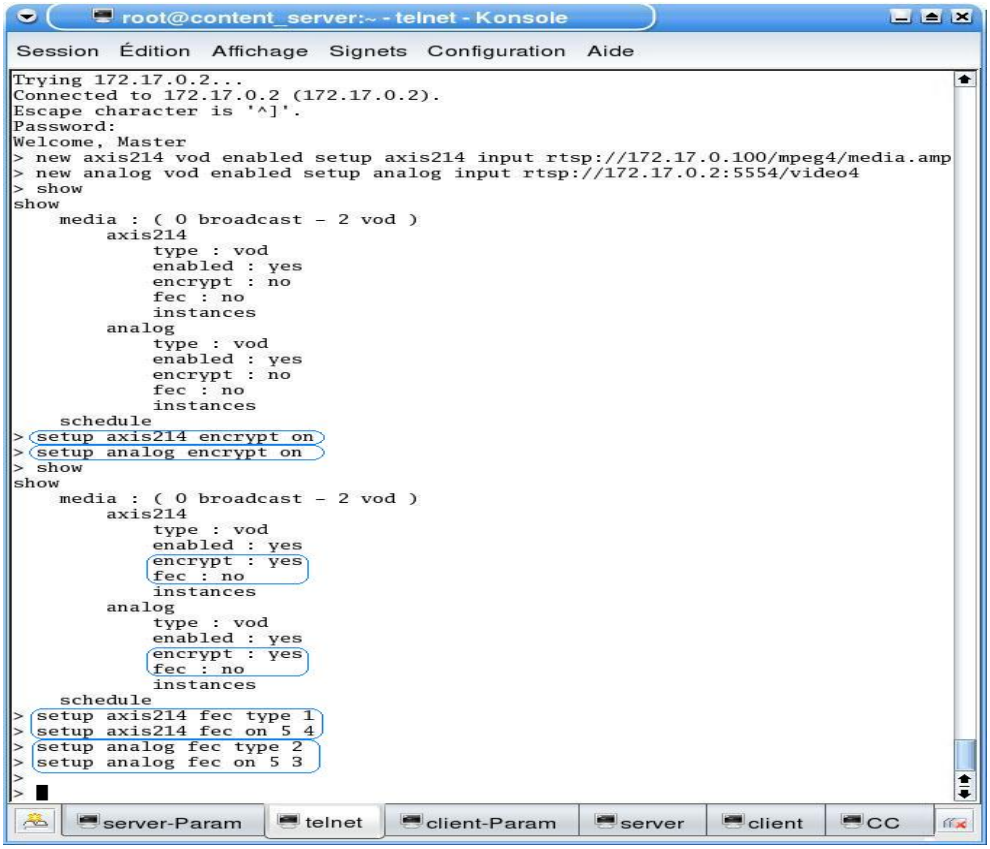


***Figure 20 : Paramétrage du CC***

L'administrateur peut visualiser sa configuration en utilisant la commande *show*.

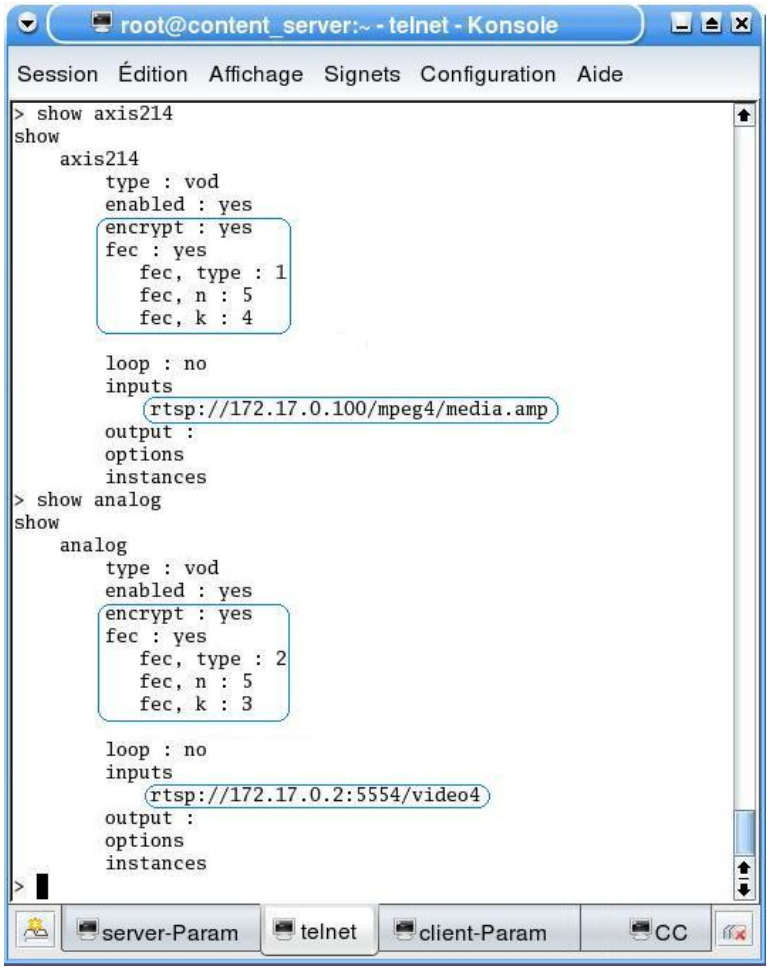


***Figure 21 : Consultation des paramètres***

## 3.6 Portage sur le CC

Le CC tourne sur un système embarqué de type PowerPC. L'instance VLC pour le CC est donc obtenue grâce à la cross-compilation. Les différentes librairies nécessaires (livemedia, libdvbpsi, iconv, libxml, gmp, etc.) ont été également compilées pour Linux PowerPC afin que power-pc-GCC  puisse faire les liens lors de la compilation de VLC.

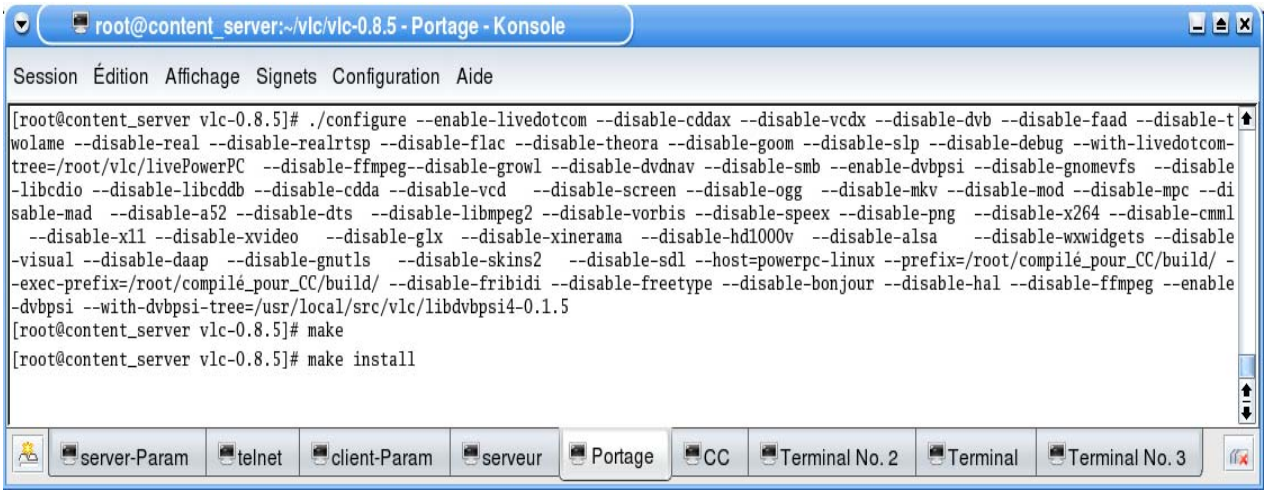


***Figure 22 : Portage sur le CC***

## 3.7 Génération des patches

La procédure pour générer et appliquer les patches est la suivante :


***Figure 23 : génération des patches***

# 4. Conclusion

Ce dernier chapitre nous a permis de présenter les différentes fonctionnalités de l'application, tel que la FEC, transfert sécurisé de flux, paramétrage de serveur, et détection des attaques.

Les modules développés sont opérationnelles et ont apportés une valeur ajoutée remarquable par rapport à la solution existante. En effet, nos solutions offre une panoplie de fonctions qui permettent de palier aux problèmes des pertes de paquets et d'assurer un transfert sécurisé de flux multimédia dans un environnement WiFi.

# II. Conclusion Générale

Pas encore

# Liste des références


[1] http://www.sit.ulaval.ca/pp/rva/presteleconf/radiotvencodeursmulticast.html

[2] Site officiel du leader européen de la diffusion de la TNT, www.tdf.fr, novembre 2005

[3] Henning Schulzrinne, S. Casner, R. Frederik, and V. Jacobson.

RTP : A Transport Protocol for Real-Time Applications. Internet

Draft, RFC 1889, http://www.cis.ohio-state.edu/htbin/rfc/

rfc1889.html, march 1996.

[4] C. Huitema, RTCP (Real Time Control Protocol), IETF RFC 1889 et 1890, October 2003

[5] Henning Schulzrinne, A. Rao, and R. Lanphier. Real Time Streaming Protocol

(RTSP). Internet Draft, RFC 2326, http://info.internet.isi.

edu/in-notes/rfc/files/rfc2326.txt, Avril 1998.

[6] Guillaume Rincé, http://guillaume.rince.free.fr/, rapport technique, Avril 2005.

[7] http://screspin.free.fr/mpeg/rapport.htm

[8] Chad Fogg, http://www.mpeg.org, 07 Jul 04.

[9] Site oficiel : http ://www.wi-fi.com/OpenSection/index.asp

[10] http://www.ieeexplore.ieee.org

[11] IEEE standard 802.11, 1999 edition : WLAN MAC and PHY layers specifications.

[12] H.F. Mattson and G. Solomon. A new treatment of Bose-Chaudhuri codes. Journal of the Society of Industrial and Applied Mathematics (SIAM), 9(4) :654{669, December 1961. Cit_e page(s) 8

[13] http://www.freeradius.org/

[14] Bruce Potter, Bob Fleck, 802.11 Security, O'REILLY, décembre 2002

[15] Axel Sikora, Wireless personal and local area networks, John Wiley & sons, 2003

[16] http://www.uqtr.ca/~delisle/Crypto/prives/blocs_default.php

[17] H.Fallon, A.Lattre, J.Bilien, A.Daoud, M.Gautier, C.Stenac, “Guide de l’utilisateur de VLC” http://salug.hosting.cri74.org/Multimedia/VLC/vlc-user-guide-fr.pdf, 2003.

[18] Site officiel des développeurs de VLC, http://www.videolan.org, 20 décembre **2005.**

# Glossaire

**ANSI**, American National Standards Institute
**HTTP**, HyperText Transport Protocol, protocole de transport des fichiers hypertextes entre un serveur WEB et un client (navigateur);
**HTTPS**, HyperText Transport Protocol Secured;
**IETF Internet Engineering Task Force** un groupe informel, international, ouvert à tout individu, qui participe à l'élaboration de standards pour Internet. L'IETF produit la plupart des nouveaux standards d'Internet.
**IP**, Internet Protocol ;
**ISO**, International Standardisation Organisation ;
**ISO/IEC**, ISO, the International Organization for Standardization, and IEC, the International Electrotechnical Commission
**ITU**, International communication union est une organisation internationale du système des Nations unies au sein de laquelle les Etats et le secteur privé coordonnent les réseaux et services mondiaux de télécommunication.
**LDAP**, Lightweight Directory Access Protocol est un protocole permettant l'accès des annuaires. C'est un protocole défini à l'IETF pour simplifier l'accès (consultation, modification) aux annuaires supportant les modèles d'information X.500 et pour favoriser les implémentations et l'usage des annuaires.
**NIST**, National Institute of Standards and Technology. Service du ministère américain de l'économie, définissant les normes utilisées par le gouvernement US
**TCP**, Transmission Control Protocol;
**UDP**, User Datagram Protocol;
**UML**, Unified Modeling Langage ; est un langage graphique de modélisation des données et des traitements, c'est une formalisation très aboutie et non-propriétaire de la modélisation objet utilisée en génie logiciel.
**URL**, Uniform Ressource Locator, format de l'adresse (localisation) d'un document à télécharger: protocole://nom-du-serveur/répertoire/nom-du-fichier
**XML**, eXtensible Markup Langage;

# Annexe A

**Tutorial**

# VideoLanClient

c'est quoi:

VLC media player, le chef d'oeuvre du projet VideolanClient, initie au départ par des étudiants de la Centrale Paris, puis après sa mise sous licence GPL permettant ainsi une coopération des développeurs du monde entier

c'est, tout d'abord , un lecteur multimédia,multi plalteformes, très léger,qui intègre déjà presque tous les types de codecs
Mais les fonctions du lecteur ne se limitent pas à la simple lecture de fichiers multimédias, puisque VLC peut être utilisé pour lire des DVD Vidéo, les VCD, SVCD et certains protocoles de streaming. L'application peut également être configurée en tant que serveur unicast ou multicast sur des réseaux IPv4 ou IPv6, à la bande passante conséquente.

## Installation

VlC est compose d un module principale et de plusieurs autres plugins qui doivent être compiles separement, c'est ainsi que compiler VLC est loin d être une tache facile,
Si vous installez VLC par un fichier binaire, vous aurez tous les modules par défaut. Si vous désirez peronnaliser VLC, vous devrez le compiler depuis ses sources.

-installer depuis la source

Modules d'entrée
Ces modules permettent à VLC de lire depuis différentes sources. VLC essaie de choisir le module le plus adapté au moment de la lecture. Si vous voulez forcer un module particulier, lancez VLC avec l'option : *--access modulename*
**cdda**
*Activé par défaut*
Module de lecture de CDs audio. Si vous avez la bibliothèque libvcdinfo dans une version assez récente et une bibliothèque libcdio assez récente, vous aurez le nouveau module gérant la navigation. L'ancien module est appelé à disparaître.
- **dvb**

*Desactivé par défaut*

*Seulement pour GNU/Linux*
Ce module permet de lire depuis des cartes DVB-S, DVB-T et DVB-C, de satellite, télévision numérique terrestre ou câble. Il utilise l'API Video4Linux 2, qui n'existe que dans les versions 2.5 et 2.6 du noyau Linux .
-
- **dvd**

*Activé par défaut*
Ceci est l'ancien module de lecture de DVD. Vous aurez besoin de *libdvdcss* pour décrypter les DVD .

- **dvdplay**

*Activé par défaut*
Ceci est le module de lecture de DVD standard. Vous aurez besoin de *libdvdcss* pour décrypter les DVD et de *libdvdplay* pour les menus DVD.

- **dvdread**

*Desactivé par défaut*
Ce module, alternative au précédent, utilise *libdvdread* pour lire les DVDs et *libdvdcss* pour décrypter les DVD .

- **http,ftp,udp,file,directory,mms**

*Toujours activé*
Ces module sont les modules d'entrée standards. Le module HTTP peut être utilisé pour la vidéo à la demande (VOD) .

- **pvr**

*Desactivé par défaut*
*Seulement pour GNU/Linux*
Ce module permet de lire depuis les cartes Hauppauge PVR .

- **satellite**

*Desactivé par défaut*
Ce modulee d'entrée permet de lire depuis une carte Hauppauge WinTV Nova sous GNU/Linux.

**-slp**
*Activé par défaut*
Ce module vous permet de récupérer les noms et addresses pour les flux annoncés à l'aide du protocole SLP.

- **v4l**

*Desactivé par défaut*
*Seulement pour GNU/Linux*
Ce module vous permet de lire des flux Video4Linux .

- **vcd**

*Activé par défaut*
Module de lecture de VideoCD. Si vous avez la bibliothèque libvcdinfo dans une version assez récente (de vcdimager) et une bibliothèque libcdio assez récente, vous aurez le nouveau module gérant la navigation. L'ancien module est appelé à disparaitre. Voir doc/intf-vcd.txt pour plus de détails.

## Démultiplexeurs

Dans un flux, les signaux vidéo et audio sont toujours inclus dans un format "conteneur". Les démultiplexeurs extraient les flux de ce conteneur et les envoient aux décodeurs .
Par exemple, un fichier AVI peut contenir une vidéo encodée en MPEG-4, ou une vidéo non compressée. AVI est seulement un format de stockage, pas un format de compression.

- **avi**

*Toujours activé*
Ce module vous permet de lire les fichiers .avi .

- **asf**

*Toujours activé*
Ce module vous permet de lire les fichiers .asf
- **aac**

*Toujours activé*
Ce module vous permet de lire les fichiers AAC
- **ogg**

*Activé par défaut*
Ce module vous permet de lire les fichiers .ogg
- **rawdv**

*Toujours activé*
Ce module vous permet de lire les fichiers DV
- **dvbpsi**

*Activé par défaut*
Ce module vous permet de lire les flux d'une carte satellite.

- **mp4**

*Toujours activé*
Ce module vous permet de lire les fichiers .mp4

- **mkv**

*Activé par défaut*
Ce module permet de lire les fichiers utilisant le format Matroska .

- **ps,ts**

*Toujours activé*

Ces modules vous permettent de lire les flux encapsulés en MPEG2-Program Stream ou Transport Stream .
Options:

- **id3,m3u**

*Toujours activé*
Ces modules vous permettent de lire les playlists M3U, B4S, PLS, et ASX, ainsi que les tags ID3.

## Décodeurs

Ces modules permettent à VLC de supporter de nombreux codecs (formats de compression).

- a52

*Activé par défaut*
Ce décodeur utilise liba52

- **cinepak**

*Activé par défaut*
Ce décodeur lit le format Cinepak .

- **faad**

*Desactivé par défaut*
Faad est un décodeur audio MPEG-4 .il faut installer la librairie Faad2,

- **ffmpeg**

*Activé par défaut*
- Ceci est un décodeur MPEG-4/DivX/OpenDivX/Xvid
-
- **libmpeg2**

*Activé par défaut*
Ceci permet de lire les fichiers MPEG2 .

- **mad**

*Activé par défaut*
Ce décodeur décode les MP3 en n'utilisant que des nombres entiers, ce qui permet de l'utiliser sur les PDAs .

- **ogt**

*Activé par défaut*
OGT gères les sous-titres SVCD (Philips Overlay Graphics Text ou OGT) et les sous-titres utilisés par les VCD Chaoji (connus sous le nom de CVD).

- **spudec**

*Activé par défaut*
spudec gère les sous-titres DVD. Les sous-titres en vraies couleurs et la transparence ne sont pas pris en charge.

- **tarkin**

*Desactivé par défaut*
tarkin est décodeur expréimental du projet Ogg

- **theora**

*Desactivé par défaut*
theora est décodeur expréimental du projet Ogg

- **tremor**

*Desactivé par défaut*
Ce codec Ogg/Vorbis ne fait que des calculs entiers, ce qui permet de l'utiliser sur des CPUs qui ne supportent pas la virgule flottante

- **vorbis**

*Activé par défaut*
Ce décodeur vous permet de lire les fichiers audio encodés en Vorbis .

- **xvid**

*Desactivé par défaut*
Ce décodeur vous permet de décoder les fichiers Xvid grâce à la librairie xvidcore.

## Modules de sortie vidéo

Les modules de sortie vidéo permettent d'afficher de la vidéo sur votre écran. Au lancement, VLC essaie de deviner le module de sortie vidéo le plus adapté à votre système. Si vous désirez toutefois forcer un module particulier, lancez VLC avec l'option: --vout modulename

- **directx**

*Activé par défaut sur Windows*
*Pour Windows seulement*
Cette sortie vidéo utilise les librairies Microsoft Direct X. Elle est recommandée sous Windows.
Vous pouvez indiquer le chemin des librairies et en-têtes DirectX, avec l'option **--with-directx=PATH** du script de configuration.

- **x11**

*Activé par défaut*
*Seulement sur systèmes Unix avec serveur X11*
Ceci est la sortie vidéo X11 basique. Elle ne requiert pour fonctionner qu'un serveur X11. Vous aurez besoin des en-têtes xlibs pour la compiler (paquetage *xlibs-dev* sur le système Debian GNU/Linux)

- **xvideo**

*Activé par défaut*
*Pour systèmes GNU/Linux seulement*

Cette sortie vidéo, qui utilise l'accélération matérielle pour la transformation YUV et la mise à l'échelle, nécessite une carte vidéo supportant xvidéo. (C'est le cas de presque toutes les cartes modernes).

- **sdl**

*Activé par défaut*
Cette sortie vidéo utilise les librairies SDL. Vous avez besoin d'une version supérieure ou égale à 1.1.6 de ces librairies
Vous pouvez indiquer le chemin du programme *sdl-config*, avec l'option **--with-sdl-config-path=PATH** du script de configuration, si vous voulez compiler VLC.

- **wingdi**

*Activé par défaut sur Windows*
*Pour Windows seulement*
Cette sortie vidéo utilise la bibliothèque GDI. Elle est connçue pour les utilisateurs n'ayant pas Direct X. Ses performances étant mauvaises, ne l'utilisez pas si vous pouvez utiliser DirectX.

- **fb**

*Activé par défaut sur GNU/Linux*
*Pour GNU/Linux seulement*
Cette sortie vidéo utilise le framebuffer. Vous devez activer le support du framebuffer dans votre noyau Linux pour l'utiliser.

- **glide**

*Desactivé par défaut*
Cette sortie vidéo utilise les librairies Glide, qui fournissent une accélération matérielle pour les cartes 3Dfx).
Vous pouvez indiquer le chemin de la librairie, avec l'option --with-glide=PATH du script de configuration si vous voulez compiler VLC.

- **mga**

*Desactivé par défaut*
*Pour GNU/Linux seulement*
Ce module permet une accélération matérielle avec les cartes Matrox sous GNU/Linux.

- **ggi**

*Desactivé par défaut*

- **aa**

*Desactivé par défaut*
Cette sortie vidéo affiche de l'ASCII art, en utilisant la bibliothèque aalib. Vous avez besoin des en-têtes aalib (paquet *aalib1-dev* sous Debian GNU/Linux) pour la compilation.

- **svgalib**

*Desactivé par défaut*

*Pour GNU/Linux seulement*
Cette sortie vidéo utilise la librairie SVGAlib.

- **qte**

*Desactivé par défaut*
*Pour iPaq seulement*
Cette sortie vidéo utilise QT Embedded, une librairie graphique spécifique à l'iPaq. .
Modules de filtre vidéo
Ces modules vous permettent de modifier l'image (désentrelacement, réglage du trio teinte/constraste/saturation, recadrage etc.). Pour les activer, utilisez l'option suivante pour VLC : --filter filter1,filter2,...

- **adjust**

*Toujours activé*
Ce filtre vous permet de modifier le contraste, la teinte, la saturation, et la luminosité.

- **crop**

*Toujours activé*
Ce filtre vous permet de rogner des parties de l'image.

- **transform**

*Toujours activé*
Ce filtre vous permet de tourner l'image de plusieurs façons.

- **distort**

*Toujours activé*
Ce filtre créée un effet de distortion de la vidéo.

**- invert**
*Toujours activé*
Ce filtre inverse les couleurs.

- **motionblur**

*Toujours activé*
Ce filtre donne un effet de "flou de mouvement" à l'image.

- **wall**

*Toujours activé*
Ce filtre vous permet de découper la vidéo sur plusieurs fenêtres que vous pouvez déplacer indépendamment. Vous pouvez l'utiliser pour générer des murs d'images avec plusieurs sources.

- **clone**

*Toujours activé*
Ce filtre vous permet de dupliquer l'image.

Modules de sortie audio

Ces modules vous permettent de choisir le système de sortie du son. VLC essaie de deviner le module de sortie audio le plus adapté à votre système. Si vous souhaitez toutefois forcer l'utilisation d'un module spécifique, lancer VLC avec l'option suivante : --aout modulename

- **coreaudio**

*Activé par défaut sous Mac OS X*
*Pour Mac OS X seulement*
Cette sortie audio utilise CoreAudio, sous Mac OS X

- **directx**

*Activé par défaut sous Windows*
*Pour Windows seulement*
Cette sortie audio utilise directX under Windows

- **oss**

*Activé par défaut sur GNU/Linux*
*Pour GNU/Linux et Unix seulement*
Cette sortie son utilise OSS (Open Sound System) (/dev/dsp, par exemple, sous GNU/Linux). Votre noyau doit avoir été compilé avec le support de votre carte mère, ou, si vous utilisez ALSA (Advanced Linux Sound System), la couche d'émulation OSS doit être activée

- **alsa**

*Desactivé par défaut*
*Pour GNU/Linux seulement*
Cette sortie audio, qui utilise ALSA (Advanced Linux Sound Architecture), ne fonctionne que sous GNU/Linux, et nécessite l'installation préalable des pilotes et librairies ALSA

- **esd**

*Desactivé par défaut*
*Pour GNU/Linux & Unix seulement*
Cette sortie audio utilise ESD (Enlightened Sound Daemon) qui est généralement utilisé avec Gnome. Vous devez avoir le démon et ses librairies.

- **arts**

*Desactivé par défaut*
*Pour GNU/Linux & Unix seulement*
Cette sortie audio utilise aRts (le serveur de son de KDE). Vous devez avoir le démon et ses librairies.

- **waveout**

*Activé par défaut sous Windows*
*Pour Windows seulement*
Cette sortie WAV est utilisée sous Windows.

- **sdl**

*Activé par défaut*
Cette sortie audio utilise SDL. Voir dans les sorties vidéo

## Modules d'interface

Ces modules vous permettent de choisir une ou des interfaces (interface graphique ou interface de controle).

- **wxwindows**

*Activé par défaut*
L'interface wxWindows est une interface graphique portable qui fonctionne sous GNU/Linux et Windows. C'est maintenant l'interface graphique la plus à jour pour ces deux systèmes .
- **skins**

*Activé par défaut*
Cette interface skinnable, qui fonctionne sous Win32 et X11 vous permet de très simplement créer vos propres skins, à l'aide de fichiers XML.

**gtk**

*Activé par défaut*
Ceci est l'interface GTK+, qui peut également être utilisée sous Windows. Vous devez disposer des librairies GTK et de en-têtes pour la compilation. Attention, cette interface n'est plus maintenue, et ne présente pas toutes les possibilités actuelles de VLC. .
- **gnome**

*Desactivé par défaut*
*Pour systèmes GNU/Linux seulement*
Ceci est l'interface Gnome. Vous aurez besoin des librairies Gnome (*libgnome32* sous Debian GNU/Linux) et des en-têtes pour la compilation (*libgnome-dev* sous Debian GNU/Linux). Attention, cette interface n'est plus maintenue, et ne présente pas toutes les possibilités actuelles de VLC. .
- **qt**

*Desactivé par défaut*
Ceci est le module d'interface QT. Vous aurez besoin des librairies QT (*libqt2* sous Debian GNU/Linux) et des en-têtes pour la compilation (*libqt-dev* sous Debian GNU/Linux). Attention, cette interface n'est plus maintenue, et ne présente pas toutes les possibilités actuelles de VLC. .
- **kde**

*Desactivé par défaut*
*Pour systèmes GNU/Linux seulement*
Ceci est le module d'interface KDE. Vous aurez besoin des librairies Kde (*kdelibs3* sous Debian GNU/Linux) et des en-têtes pour la compilation ((*kde-devel* sous Debian GNU/Linux). Attention, cette interface n'est plus maintenue, et ne présente pas toutes les possibilités actuelles de VLC. .

- **rc**

*Toujours activé*

Ceci est un module de contrôle à distance, qui fonctionne en mode texte depuis une console. Vous pouvez ainsi contrôler le VLC par des scripts ou des commandes, comme *play*,*stop*, etc.... .

- **http**

*Toujours activé*
Ce module vous permet de contrôler VLC à distance par un navigateur web. Vous pouvez créer vos propres pages web de contrôle. .

- **ncurses**

*Desactivé par défaut*
*Pour systèmes GNU/Linux seulement*
Cette interface en mode texte utilise la bibliothèque ncurses. Pour la compiler, vous devez avoir les en-têtes ncurses (*libncurses5-dev* sous Debian GNU/Linux) .

- **lirc**

*Desactivé par défaut*
*Pour systèmes GNU/Linux seulement*
Ce module permet de contrôler VLC par une télécommande. Afin de vous aider, un fichier lircrc est fourni

- **opie**

*Desactivé par défaut*
Ceci est un plugin d'interface pour la librairie QT Embedded (pour iPaq) .

- **gestures**

*Toujours activé*
Ce module vous permet de contrôler VLC par des mouvements de souris.

- **joystick**

*Desactivé par défaut*
*Pour systèmes GNU/Linux seulement*
Ce module vous permet de contrôler VLC à l'aide d'un joystick et est hautement configurable.

- **Divers**

Cette section décrit quelques modules qui n'entrent dans aucune des catégories décrites.

- **sout**

*Activé par défaut*
Le Stream Output est une fonctionnalité de VLC qui lui permet de devenir serveur de streaming MPEG, DivX, ou DVD.

- **test-suite**

*Desactivé par défaut*
Ceci fait un VLC spécial, pour des test .

- **mozilla**

*Desactivé par défaut*
Ceci n'est pas un module, mais permet de créer un plugin Mozilla VLC.

- **xosd**

*Desactivé par défaut*
*Pour Unix seulement*
Ceci envoie le flux sur un "OSD" (On Screen Display) .

Pour toutes les librairies il faut avant de faire la compilation, lire le fichier INSTALL ou README, pour les options de configuration, ou alors faire un. /configure - -help

La démarche à suivre pour installer les sources:

```
gunzip     source.tar.gz (bzip2 source.tar.bz2)
tar -xvf source.tar
cd source
./configure
make
make install
```

NB: pour ffmpeg, il faut le compiler avec –enable-pp
Pour wxwindows, il faut le compiler avec –with-x11

Pour live.com; faites comme suit :genMakefiles
Make
Cp –r live.com /usr/lib
Si vous rencontrez des problèmes avec ffmpeg, genre type invalide, éléments manquants ; vérifiez la version de votre compilateur avec gcc –version ; ffmpeg est optimisé pour des versions 3 .x ; prenez donc la peine d installer un gcc approprie ou appelez un gcc 3.x si vous en avez deja ;avec
export cc=gcc3.X

Si pour le pre,eir lancement de VLC ;vous avez ce message ;alors ne paniquez pas ;copiez juste les fichiers requis vers /usr/lib

Cp /usr /lib/local/libdvbpsi8 /usr/lib

**Phase finale**

Compilez vlc après avoir installer les sources requises, en activant les options dont vous aurez besoin.
Une compilation possible est :

```
./configure --enable-x11 --enable-xvideo --disable-gtk --enable-sdl
--enable-ffmpeg --with-ffmpeg-mp3lame --enable-mad --enable-
libdvbpsi --enable-a52 --enable-dts --enable-libmpeg2 --enable-
dvdnav --enable-faad --enable-vorbis --enable-ogg --enable-theora -
-enable-faac --enable-mkv --enable-freetype --enable-fribidi --
enable-speex --enable-flac --enable-livedotcom --with-livedotcom-
tree=/usr/lib/live --enable-caca --enable-skins --enable-skins2 --
enable-alsa --disable-kde --disable-qt --enable-wxwindows --enable-
ncurses --enable-release
```

## Annexe B

# VLC Stream output

| | |  |  |  |  |  |  |
|---|---|---|---|---|---|---|---|
| **Inputs** | | See the VLC features page | | | | | |
| | UDP Unicast / Multicast | Yes | Yes | Yes | unicast only | Yes | Yes |
| **Output** | RTP Unicast / Multicast | Yes | Yes | Yes | unicast only | Yes | Yes |
| | File | Yes | Yes | Yes | Yes | Yes | Yes |
| | HTTP | Yes | Yes | Yes | Yes | Yes | Yes |
| | MMSH | Yes | Yes | Yes | Yes | Yes | Yes |
| | Transcoding | Yes | Yes | Yes | Yes | Yes | Yes |
| **Misc** | Send DVD subtitles | Partial | Partial | Partial | Partial | Partial | No |
| | Send SAP announces | Yes | Yes | Yes | Untested | Yes | Untested |
| **Interfaces and more** | | See the VLC features page | | | | | |