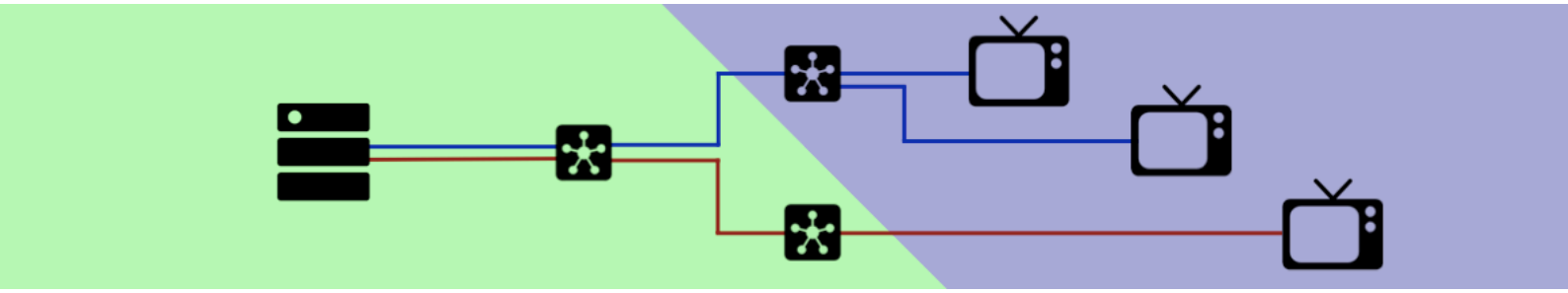
Haute école d'ingénierie et d'architecture Fribourg
Hochschule für Technik und Architektur Freiburg

Bd de Pérolles 80
case postale 32
CH-1705 Fribourg
t. +41 (0)26 429 66 11
f. +41 (0)26 429 66 00
www.heia-fr.ch

net+ SimBox

Projet de Semestre 6



Titre du projet	Simulateur de Set-Top Box	
Filière	Télécommunications orientation Réseaux et Sécurité	
Etudiant	Romain Froidevaux	<romain.froidevaux@edu.hefr.ch>
Professeurs	François Buntschu	<francois.buntschu@hefr.ch>
	Patrick Gaillet	<patrick.gaillet@hefr.ch>
Mandants	Alain Schmoutz	<alain.schmoutz@netplusfr.ch>
	Patrick Gaudin	<patrick.gaudin@netplusfr.ch>
Classe	T-3a	
Date	20.05.2015	
Version	1.1	

Hes·so FRIBOURG FREIBURG
Haute Ecole Spécialisée de Suisse occidentale
Fachhochschule Westschweiz
University of Applied Sciences and Arts Western Switzerland

Historique du document

Version	Auteur	Description	Date
0.1	Romain Froidevaux	Création du document	04.05.2015
1.0	Romain Froidevaux	Publication du document	13.05.2015
1.1	Romain Froidevaux	Ajout des sources manquantes	20.05.2015

Résumé

Depuis quelques années maintenant, les principaux fournisseurs d'accès du pays déploient des services de diffusion IPTV sur leurs réseaux respectifs. Ces nouveaux types de trafics impliquent un volume croissant de données qui doit être traité tant dans le réseau de l'opérateur qu'au niveau des équipements chez les clients finaux.

En ce sens l'entreprise fribourgeoise net+ FR impliquée dans le développement de la fibre optique du canton de Fribourg fait face à la nécessité de pouvoir simuler des flux IPTV depuis un réseau client afin de dimensionner au mieux son réseau d'accès ainsi que les équipements distribués aux abonnés.

Dans ce projet, les architectures de diffusion des différents types de flux TV dans les réseaux IP traditionnels ont tout d'abord été étudiées. Ensuite les aspects techniques ont été décrits au travers des différents protocoles utilisés dans le domaine.

Cette analyse a ensuite permis de spécifier et concevoir un simulateur de Set-Top Box dans le but de s'abonner à des flux TV Live et Replay de la même manière qu'un client et ce sur n'importe quelle infrastructure de télévision IP. Il donne ainsi la possibilité au mandant de réaliser un test de charge sur les équipements d'accès en simulant divers profils, allant de l'abonné traditionnel à un hôtel d'une centaine de chambres toutes équipées en téléviseurs.

Finalement l'application a été portée de l'infrastructure pilote de la HEIA-FR sur le réseau de production du mandant, net+ FR.

Le projet ayant été pensé pour être étendu par la suite, ce rapport contient pour chacun des choix effectués une justification et recense les difficultés rencontrées. De futurs travaux pourront être encore envisagés du fait des nombreuses possibilités d'innovations et d'améliorations.

Abstract

For some years now, the main ISPs are deploying IPTV broadcast services on their respective networks. These new types of traffic involve an increasing volume of data that must be considered both in the operator's network level as the final customers equipment.

In this sense the net+ FR company, involved in the development of fiber optics in the canton of Fribourg, is facing the need to simulate IPTV streams from a client's network in order to plan as good as possible its access network as well as the equipment distributed to subscribers.

The delivery architectures for different types of TV streams in traditional IP networks were first studied. Then the technical aspects have been described through the various protocols used in the field.

This analysis was then used to specify and design a Set-Top box simulator in order to subscribe to Live and Replay TV streams in the same way that a client on any IPTV infrastructure. Thus, it offers the possibility to the project mandate to perform a load test on the access equipment by simulating various profiles, ranging from the traditional subscriber to a hotel with a hundred rooms all equipped with televisions.

Finally, the application was ported from the HEIA-FR pilot infrastructure to the production network of net+ FR.

The project was thought to be extended thereafter. The current report contains for each choice a justification and identifies the difficulties met. Future works on the topic may still be considered due to many opportunities of innovations and improvements.

Table des matières

1 Introduction

net+ FR est un fournisseur de services de télécommunications actif dans le canton de Fribourg depuis janvier 2013. L'entreprise fournit entre autre des services B2C Triple play sur une multitude de technologies d'accès telles que le FTTH, DOCSIS et xDSL.

Afin d'optimiser son réseau de transport, net+ FR a besoin d'un client permettant de simuler des Set-Top Box pour réaliser des tests de charge, tant pour des flux Multicast *(TV Live)* que Unicast *(TV Replay)*.

Le but principal de ce projet est de spécifier, réaliser, mettre en place et tester un tel outil. Dans un second temps, l'ajout de fonctionnalités supplémentaires telles que la mesure de la qualité des flux ou la mesure de l'efficience de la QoS pourront être développées.

L'outil devra pouvoir être utilisé au sein d'une architecture réseau de type B2C Triple play telle que schématisée dans la figure 1.

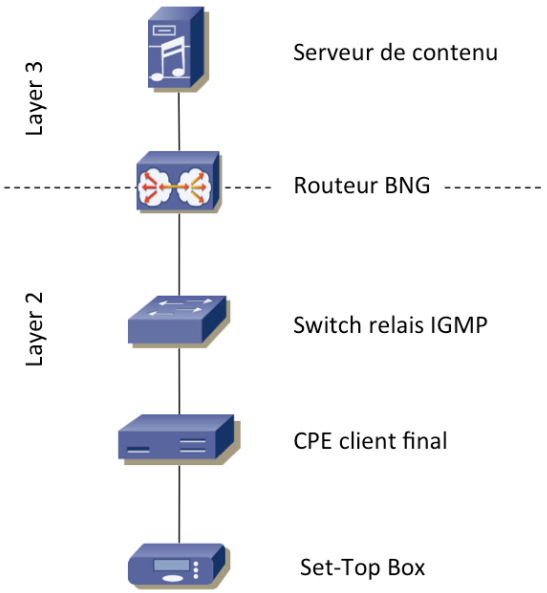


Figure 1 : Réseau IPTV standard

1.1 Déroulement projet

Ce projet a été divisé en plusieurs phases, chacune de ces phases correspondent à un chapitre de ce rapport.

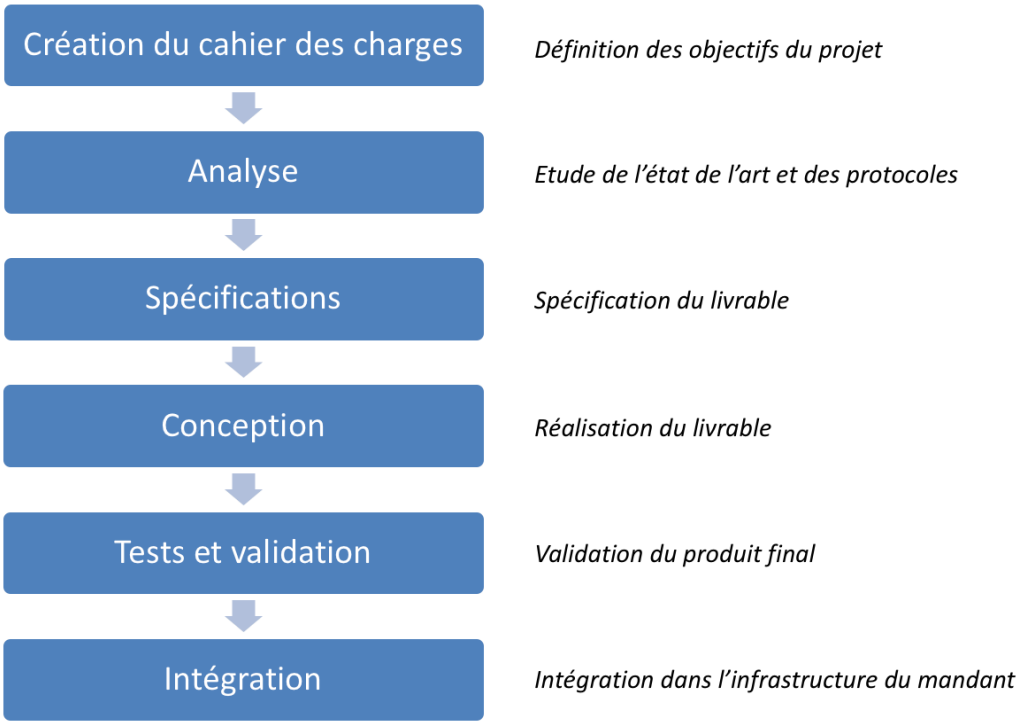


Figure 2 : Déroulement du projet

2 Cahier des charges

Le cahier des charges a pour but de fixer le cadre et les objectifs du projet sur la base de la donnée reçue. Il contient une planification détaillée, une énumération des étapes, ainsi qu'une description des différents objectifs devant être atteints d'ici la fin du projet.

2.1 Objectifs primaires

Afin de mener à bien le projet, 3 objectifs clés ont été formulés. Tous ceux-ci devront être atteints à la fin du travail :

1. Enumération, analyse et documentation des différentes solutions software et hardware de génération de paquets ainsi que des architectures de diffusion de contenu multimédia de type IPTV.
2. Mise en place d'une maquette de diffusion Multicast dans le but de pouvoir tester le simulateur au sein d'un environnement de tests.
3. Conception et développement du simulateur de Set-Top Box selon les spécifications établies afin de pouvoir l'utiliser dans le réseau de net+ FR.

2.2 Objectifs secondaires

Dans un second temps et selon l'avance du projet par rapport à la planification initiale, les objectifs secondaires suivants ont été formulés et pourront être implémentés :

1. Prise en charge d'IGMPv2 en plus d'IGMPv3 pour donner le choix du protocole à utiliser lors des mesures.
2. Réalisation d'une interface graphique pour afficher le nombre de flux actuellement acquis par le simulateur.
3. Développement d'une interface permettant de manipuler indépendamment chaque flux généré par le simulateur *(changement de chaîne, mise en pause de la diffusion, arrêt du flux, etc…)*.
4. Ajout d'une possibilité de vérification de la qualité des flux reçus par rapport à ceux émis sur la base de critères qualitatifs *(jitter, latence, taux de perte, etc…)*.

2.3 Description des phases du projet

2.3.1 Analyse

Cette première phase consiste à étudier l'état de l'art *(outils disponibles sur le marché)* et les technologies *(multicast et distribution IPTV)* ainsi que collecter des informations permettant de bien saisir le contexte et l'environnement du projet. Elle est séparée en deux étapes :

a) Analyse technique orientée flux TV

Afin de bien comprendre les mécanismes de distribution de contenus multimédias dans les réseaux, les points suivants seront analysés :

- Multicast : Distribution de flux de point à multipoints
- PIM : Protocole de routage des flux Multicast
- IGMP : Gestion des transmissions Multicast en couche 2
- RTSP : Protocole applicatif pour le contrôle des flux de streaming

b) Analyse des outils existants

Il existe déjà aujourd'hui des outils permettant de réaliser l'une ou l'autre des opérations demandées par le mandant. Cependant, l'outil idéal les regroupant toutes n'existe pas.

Il s'agira d'étudier ces différents outils afin de voir dans quelle mesure ils pourraient être intégrés au travail final, tant sur le plan matériel que logiciel.

2.3.2 Spécification

Cette seconde étape à pour but de définir une architecture logicielle en vue de l'implémentation, ainsi que décrire les comportements et sorties du simulateur selon les différentes entrées.

Dans un second temps il s'agira également de réaliser sur la base de l'analyse des outils existants, un choix de la plateforme de développement qui sera utilisée *(langage, librairies, bases existantes, etc…)*. Pour ce faire, un tableau multicritères de décision sera élaboré.

2.3.3 Mise en place de la maquette de tests

A cet échelon du projet, il s'agira de monter un serveur de distribution de contenus multimédia au sein d'un réseau de test. Celui-ci permettra de diffuser des flux en Multicast *(TV live*) et interprétera les messages IGMP de même version que le serveur de net+. Quant aux flux Unicast *(TV Replay)*, l'analyse technologique dira s'il est raisonnable ou pas de mettre en place un tel serveur en fonction du temps à disposition.

2.3.4 Conception

La conception est la phase principale du projet où il s'agira de développer le simulateur de Set-Top Box selon le canevas défini dans l'étape de spécification.

2.3.5 Tests et validation

Afin de valider le fonctionnement de l'outil tel que décrit dans les spécifications, un plan de test complet sera élaboré puis exécuté. Les corrections nécessaires seront apportées au code.

2.3.6 Intégration net+ FR

Une fois le simulateur testé et validé sur l'infrastructure de la HEIA-FR, une intégration sera réalisée au sein du réseau de net+ FR. Cette opération pourra être faite plus tôt en parallèle à d'autres tâches si une connexion net+ FR est disponible dans le laboratoire de l'école. Ainsi, les problèmes d'intégration potentiels seront réduits.

2.4 Organisation

2.4.1 Gestion des documents

Tous les rendus de documents *(rapports, invitations aux séances, procès-verbaux, etc…)* seront déposés sur la Forge de l'école, consultable à l'adresse suivante :

https://forge.tic.eia-fr.ch/projects/net-simbox

2.4.2 Gestion de projet

La méthode de gestion de projets et développement séquentielle SDLC Waterfall sera appliquée avec les phases telles que décrites précédemment.

2.4.3 Personnes liées au projet

Les personnes ci-dessous sont impliquées dans le projet de semestre selon les rôles suivants :

Etudiant	Froidevaux Romain	<romain.froidevaux@edu.hefr.ch>
Professeurs	Buntschu François	<francois.buntschu@hefr.ch>
	Gaillet Patrick	<patrick.gaillet@hefr.ch>
Mandants	Schmoutz Alain	<alain.schmoutz@netplusfr.ch>
	Gaudin Patrick	<patrick.gaudin@netplusfr.ch>

2.4.4 Planification

Afin de mener à bien ce projet et d'avoir une vision globale du temps à disposition, une planification détaillée a été réalisée. Une marge de sécurité a été prévue pour certains jalons en cas d'imprévus.

La planification des tâches à effectuer est disponible en annexe.

3 Analyse

La phase d'analyse consiste à étudier les différents types de transmissions utilisés pour la diffusion de contenus multimédias, tels que le Multicast pour les flux de TV live respectivement l'Unicast pour les flux de TV Replay. Chacune des deux topologies de distribution utilise des protocoles particuliers, permettant de gérer les flux de streaming. Le fonctionnement de ceux-ci sera décrit afin de pouvoir comprendre l'architecture IPTV d'un opérateur ainsi que les interactions entre les différents éléments du réseau.

Dans une seconde partie, il s'agira d'étudier les solutions de génération de flux IPTV matérielles et logicielles disponibles sur le marché. Il existe énormément d'outils, mais aucun ne rempli intégralement tous les critères du mandant. Cette phase permettra de faciliter le choix des outils et librairies à considérer dans la partie de spécifications.

3.1 Technologies de diffusion IPTV

3.1.1 Unicast et Multicast

Le multicast est une architecture de diffusion apparue au milieu des années 80 déjà. Tout d'abord réservé au domaine de la recherche, il s'est peu à peu démocratisé depuis le début des années 2000 auprès des opérateurs et des entreprises privées pour la diffusion de contenus multimédias.

Il permet la distribution d'un flux unique depuis la source à de multiples destinataires abonnés au groupe de diffusion en question, contrairement à l'architecture unicast qui duplique le flux émis par le nombre de destinataires.

3.1.1.1 Architecture

Une diffusion peut avoir lieu indépendamment en unicast ou multicast. Sur le plan architectural, seul le nombre de flux émis diffère.

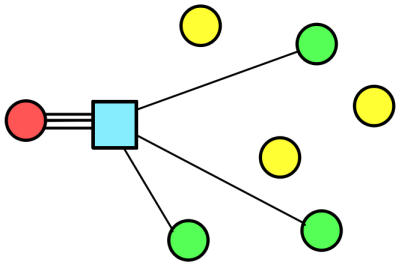


Figure 3 : Architecture unicast [UMC01]

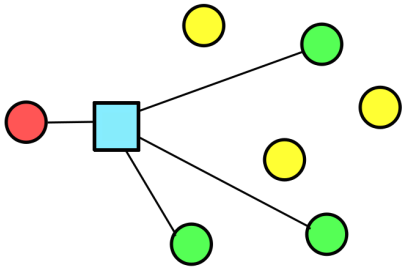


Figure 4 : Architecture multicast [UMC01]

Du point de vue d'un fournisseur IPTV, l'architecture multicast est utilisée pour diffuser les flux live tels que les chaînes de télévision ou des évènements sportifs en direct. L'architecture unicast quant à elle est utilisée pour les flux émis en différé comme la TV Replay ou la vidéo à la demande par exemple.

3.1.1.2 Avantages et inconvénients du multicast

Avantages

La diffusion de flux en multicast possède les propriétés suivantes :

- Meilleure efficience par le contrôle du trafic réseau et réduction de la charges des CPU *(serveur et éléments réseaux)*.
- Performances optimisées par l'élimination du trafic redondant.
- Possibilité d'exploiter des applications générant du contenu distribué de manière efficace.

Ci-dessous, un graphique illustrant l'efficacité d'une architecture de diffusion de type multicast par rapport à l'unicast, pour un flux audio de 8 Kbps acquis par chacun des clients :

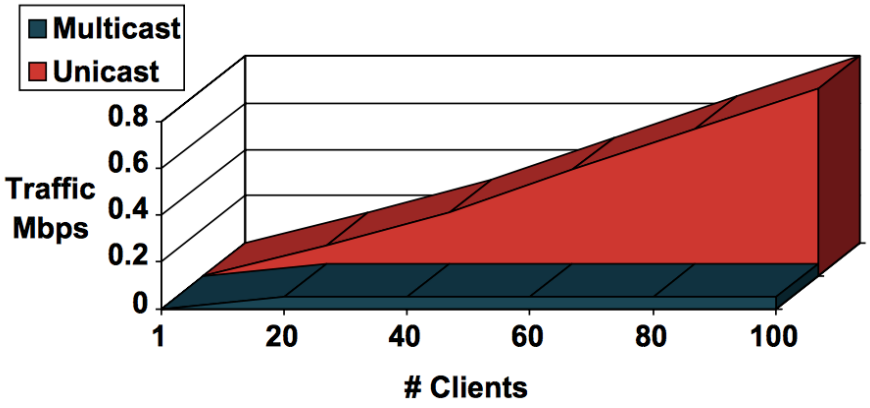


Figure 5 : Performances multicast [UMC02]

Inconvénients

Comparé à l'architecture unicast qui peut être exploitée indépendamment en TCP ou UDP, le multicast ne supporte que l'UDP, ne pouvant ainsi délivrer que du contenu en Best Effort. Il existe tout de même un protocole permettant de garantir une intégrité des données reçues, appelé PGM, mais n'est qu'en phase expérimentale par l'IETF. Il fonctionne par le principe inverse du TCP, en numérotant les paquets par des numéros de séquences et le client se manifeste par un NACK *(Acquittement négatif)* pour demander la rediffusion d'un paquet.

Ainsi, le protocole UDP apporte les inconvénients suivants à la diffusion multicast :

- Il n'est pas possible de réaliser au niveau du réseau un algorithme d'évitement de la congestion, tel qu'implémenté dans TCP avec le Slow Start. Cette fonction doit être mise en place au niveau des applications directement si elle est souhaitée.
- La génération de paquets doublons est possible dans le réseau, notamment lors de la réalisation de l'algorithme Shortest-Path Tree. Les paquets UDP n'étant pas identifiés par des numéros de séquences, les applications clientes doivent être prévues pour contrer cette possibilité.
- Les codecs vidéo de type MPEG utilisent beaucoup d'algorithmes prédictifs tels que le Motion Compensation. La perte ou l'arrivée hors-séquence de paquets peut conduire à des freezes chez le client le temps de resynchronisation du flux.

3.1.1.3 Concept de groupes Multicast

Le principe de diffusion multicast s'appuie sur des groupes, définis en couche 3 par des adresses IP de classe D :

- En envoyant une charge utile avec l'adresse multicast du groupe en destination, tous les membres s'étant préalablement inscrits au groupe la recevront.
- Il est nécessaire d'être inscrit au groupe pour recevoir les données de ce dernier si les switchs réalisent de l'IGMP Snooping *(Ce point sera décrit plus tard)*.
- Il est possible d'envoyer des données à un groupe sans y être inscrit.

3.1.1.4 Adressage IP et MAC Multicast

Comme indiqué précédemment, les groupes multicast sont identifiés par des adresses IP de la classe D *(224.0.0.0 à 239.255.255.255)*. Elles sont facilement reconnaissables, les bits de poids fort étant toujours `1110`.

La IANA, autorité d'assignements des adresses sur Internet, a découpé la plage selon les sections suivantes :

Plage	Description
224.0.0.0 – 224.0.0.255	Réservé au réseau local
224.0.1.0 – 224.0.1.255	Réservé inter-réseaux
224.0.2.0 – 224.0.255.255	Réservé ad-hoc
224.1.0.0 – 255.1.255.255	Non assigné
224.2.0.0 – 255.2.255.255	Réservé SDP/SAP
224.3.0.0 – 231.255.255.255	Non assigné
232.0.0.0 – 232.255.255.255	Réservé source spécifique
233.0.0.0 – 233.255.255.255	Réservé GLOP *(AS statique)*
234.0.0.0 – 238.255.255.255	Non assigné
239.0.0.0 – 239.255.255.255	En usage privé libre

Tableau 1 : Assignation des places multicast [UMC05]

Dans la première plage d'adresses réservées et assignées pour les réseaux locaux, certaines sont utiles à relever :

Adresse	Description
224.0.0.1	Tous les périphériques du réseau
224.0.0.2	Tous les routeurs
224.0.0.5	Tous les routeurs OSPF
224.0.0.6	Tous les routeurs OSPF élus DR
224.0.0.9	Tous les routeurs RIPv2
224.0.0.12	Tous les serveurs et relais DHCP
224.0.0.13	Tous les routeurs PIM
224.0.0.22	Tous les routeurs Multicast IGMPv3

Tableau 2 : Assignation des adresses multicast clés [UMC05]

Chaque paquet multicast contient un champ TTL, permettant de définir sa portée sur le réseau. Généralement les routeurs de l'Internet ne routent pas le multicast ; son nombre n'a que peu d'importance en dehors du LAN. Cependant, la norme a défini les seuils suivants :

TTL	Portée
1	Réseau local
<16	Site local
<32	Régional
<48	National
<64	Continental
>64	Mondial

Tableau 3 : Portée multicast

Sur les routeurs susceptibles de transporter des flux multicast, il est possible de définir un seuil limite du TTL, après lequel les paquets seront jetés.

Sur un routeur de la marque Cisco, cette opération s'effectue à l'aide de la commande suivante :

```
ip multicast ttl-threshold number
```

Cependant pour réduire la charge CPU, Cisco recommande de réaliser cette opération à l'aide d'une ACL étendue [UMC02].

En couche 2, chaque adresse IP multicast correspond à une adresse MAC multicast. L'algorithme de conversion est le suivant :

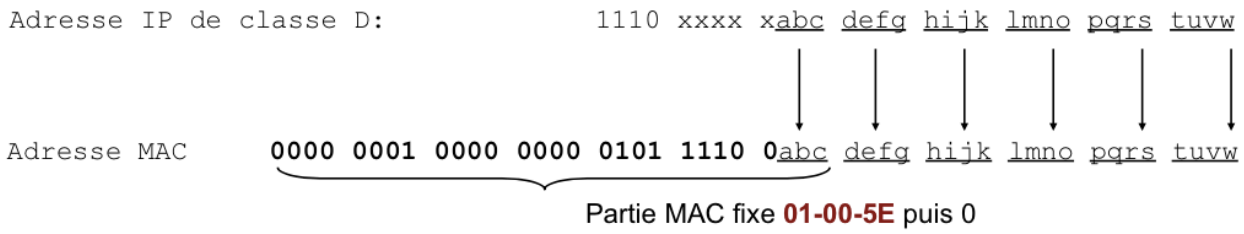


Figure 6 : Algorithme de conversion multicast IP vers MAC [UMC03]

Du fait que les bits 5 à 9 de l'adresse IP ne sont pas pris en compte dans l'algorithme, 32 adresses IP peuvent correspondre à une même adresse MAC. Cependant, pour une adresse IP, une seule adresse MAC est possible.

3.1.2 IGMP (Couche 2)

Dans une architecture Multicast, comment font les clients pour gérer leurs appartenances aux différents groupes de diffusion ? C'est dans ce contexte qu'intervient le protocole IGMP.

Protocole de couche 3, il permet de gérer la distribution des trames sur le réseau. Il implémente entre autre des méthodes d'abonnement *(Join)* et de désabonnement *(Leave)* permettant aux clients de s'abonner ou se désabonner à des flux multi-diffusés.

Trois versions d'IGMP ont été développées. C'est actuellement la version 3 qui est implémentée dans les réseaux d'opérateurs tels que celui de net+ FR. C'est donc cette version qui sera analysée et décrite. Finalement, IGMPv2 sera également étudié, principalement sous l'angle des différences avec son successeur.

3.1.2.1 Types de distributions Multicast

Deux types de distributions Multicast ont été rendus possibles grâce au protocole IGMP :

Any Source Multicast

Distribution implémentée dès la mise en place du protocole qui consiste à s'abonner à un groupe et recevoir les flux émis depuis toutes les sources disponibles.

Source Specific Multicast

Type de distribution implémenté depuis début des années 2000, permettant de s'abonner à un groupe en désignant une source spécifique de laquelle le client souhaite recevoir le flux multicast. Ce choix s'effectue au moyen d'un champ `Source Address` prévu dans la requête d'abonnement. Il sera décrit dans le prochain chapitre.

3.1.2.2 Structure des messages IGMPv3

L'IGMP est encapsulé dans un paquet IP et est identifié par le numéro de protocole `0x02` de l'en-tête IP.

IGMPv3 Membership Report Message

Les messages de type Membership Report sont toujours envoyés depuis un client vers une adresse multicast *(224.0.0.22 en IGMPv3, l'adresse du groupe en IGMPv2)* pour indiquer un état de réception. C'est avec ce type de message qu'un client peut s'abonner, confirmer son adhésion ou quitter un groupe multicast.

La structure des messages, encapsulés dans un paquet IP, est la suivante :

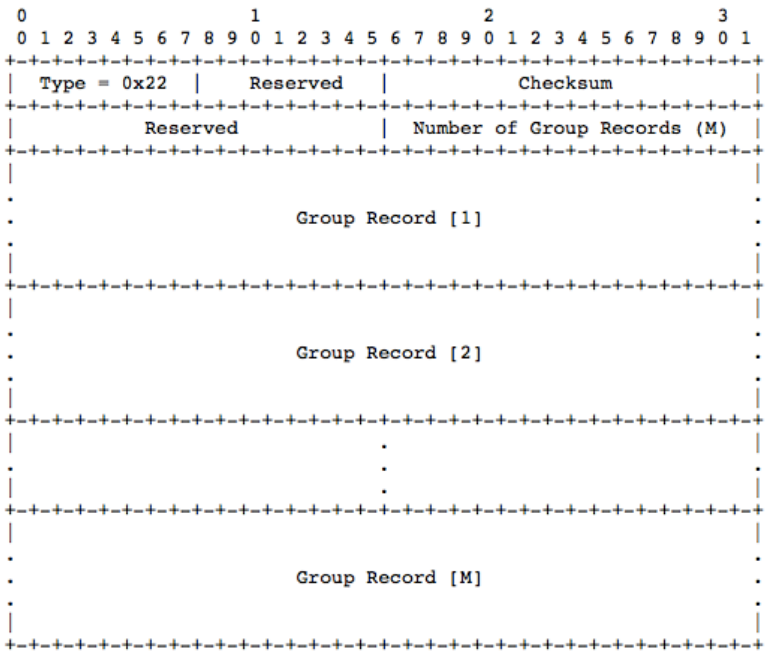


Figure 7 : Structure d'un message IGMPv3 Membership Report

Type :

`0x22` Membership Report

Reserved :

Pour usage futur *(que des zéros)*

Checksum :

Contrôle de tout le message IGMP

Number of Group Records :

Indique le nombre de groupes présents dans ce rapport

Group Record :

Chaque Group Record contient des informations de partenariat. C'est une nouveauté d'IGMPv3 qui permet de multiplexer les annonces dans un seul message.

Pour chacun des Group Record, les informations suivantes sont indiquées :

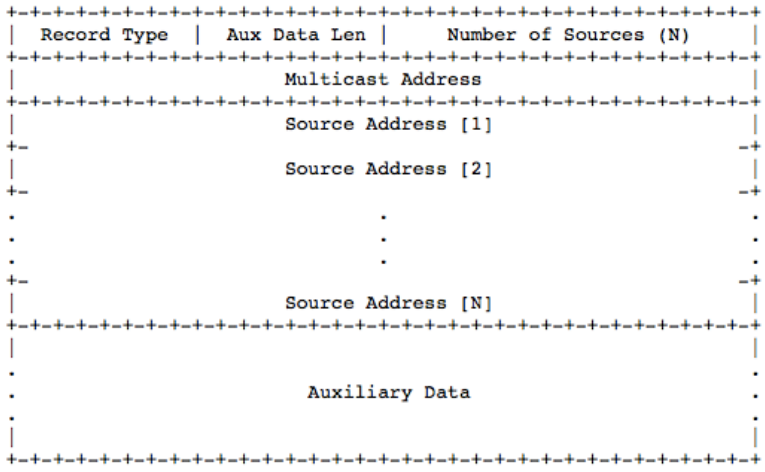


Figure 8 : Structure d'un message Group Record

Record Type :

Type de message. Cf. description ci-après.

Aux Data Len :

Longueur des données auxiliaires

Number of Sources :

Indique le nombre de sources spécifiées dans le message

Multicast Address :

Adresse de groupe multicast concernée par le message

Source Address :

Tableau spécifiant les adresses IP des sources à considérer

Auxiliary Data :

Champ prévu pour une utilisation future. Il doit impérativement contenir que des zéros dans l'implémentation d'IGMPv3.

Signification des valeurs pour le champ Record Type :

- 1 (MODE_IS_INCLUDE) :
  Actuellement non abonné au groupe multicast
- 2 (MODE_IS_EXCLUDE) :
  Actuellement abonné au groupe multicast
- 3 (CHANGE_TO_INCLUDE_MODE) :
  Demande de désabonnement au groupe multicast
- 4 (CHANGE_TO_EXCLUDE_MODE) :
  Demande d'abonnement au groupe multicast

IGMPv3 Membership Query Message

Les messages de type Membership Query sont toujours envoyés par un élément multicast de couche 3 *(généralement un routeur)* avec son adresse IP unicast à destination du groupe multicast en question pour définir l'état de réception de ses abonnés.

Les messages sont définis de la manière suivante :

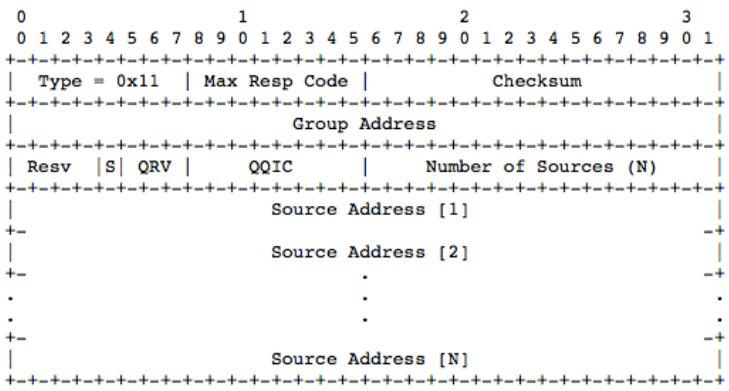


Figure 9 : Structure d'un message IGMPv3 Membership Query

Type :

`0x11` Membership Query

Max Resp Code :

Temps maximal autorisé avant l'envoi d'une réponse au rapport

Checksum :

Contrôle de tout le message IGMP

Group Address :

Adresse du groupe multicast concerné. Si tout le monde *(general)* → 0.0.0.0

Resv :

Réservé pour un usage futur *(zéros)*

S :

A 1, indique aux routeurs de ne pas tenir compte du Max Resp Code

QRV :

Indication de la fiabilité dans le sous-réseau *(valeur entre 2 et 7)*

QQIC :

Intervalle temporel entre deux messages Query

Number of Sources :

Indique le nombre de sources spécifiées dans le message

Source Address :

Liste des sources concernées par le message

Le protocole IGMPv3 définit 3 types de requêtes

- General Query :

Ces messages sont envoyés à tout le monde *(224.0.0.1)* et permettent de voir si un quelconque client est abonné à un flux muticast.

- Group Specific Query

Idem que le message de type General Query, mais ciblé à un groupe multicast donné.

- Group and Source Specific Query

Idem que le message de type Group Specific Query, mais ciblé à un groupe multicast donné, généré par les sources mentionnées.

3.1.2.3 Illustration des messages IGMPv3

Le protocole IGMPv3 est basé directement sur IP, orienté sans connexion et ne nécessite pas d'acquittement après envoi.

Pour illustrer tous ces types de messages, des captures ont été effectuées directement sur l'infrastructure de net+ FR.

Abonnement à un groupe Multicast

L'abonnement d'un client à un groupe multicast s'effectue à l'aide d'un message IGMPv3 Membership Report :

```
Internet Protocol Version 4, Src: 192.168.1.135 (192.168.1.135), Dst: 224.0.0.22 (224.0.0.22)
Internet Group Management Protocol
    [IGMP Version: 3]
    Type: Membership Report (0x22)
    Header checksum: 0xe06a [correct]
    Num Group Records: 1
  ▼ Group Record : 239.101.10.46  Change To Exclude Mode
      Record Type: Change To Exclude Mode (4)
      Aux Data Len: 0
      Num Src: 0
      Multicast Address: 239.101.10.46 (239.101.10.46)
```

Figure 10 : Abonnement à un groupe Multicast

Champ	Valeur	Remarque
Src	192.168.1.135	Adresse IP du client
Dst	224.0.0.22	Groupe multicast des routeurs IGMPv3
Type	Membership Report	-
Record Type	Change To Exclude Mode	Demande de rejoindre le groupe *(Join)*
Multicast Address	239.101.10.46	Adresse du groupe

Désabonnement à un groupe Multicast

De manière similaire à un abonnement, le désabonnement d'un client auprès d'un groupe multicast s'effectue de la manière suivante :

```
Internet Protocol Version 4, Src: 192.168.1.135 (192.168.1.135), Dst: 224.0.0.22 (224.0.0.22)
Internet Group Management Protocol
    [IGMP Version: 3]
    Type: Membership Report (0x22)
    Header checksum: 0xe16a [correct]
    Num Group Records: 1
  ▼ Group Record : 239.101.10.46  Change To Include Mode
      Record Type: Change To Include Mode (3)
      Aux Data Len: 0
      Num Src: 0
      Multicast Address: 239.101.10.46 (239.101.10.46)
```

Figure 11 : Désabonnement à un groupe Multicast

Champ	Valeur	Remarque
Src	192.168.1.135	Adresse IP du client
Dst	224.0.0.22	Groupe multicast des routeurs IGMPv3
Type	Membership Report	-
Record Type	Change To Include Mode	Demande de quitter le groupe *(Leave)*
Multicast Address	239.101.10.46	Adresse du groupe

Découverte des clients encore connectés au groupe Multicast

Périodiquement, le routeur envoie un message de type Membership Query au groupe afin de savoir si des clients y sont toujours connectés. Ceci afin d'éviter qu'un flux soit diffusé dans le vide, alors que le client l'a quitté sans avoir eu le temps de le faire proprement (*p.ex. arrêt inopiné*).

Cette demande est effectuée à l'aide du message suivant :

```
Internet Protocol Version 4, Src: 172.16.0.1 (172.16.0.1), Dst: 239.101.10.46 (239.101.10.46)
Internet Group Management Protocol
   [IGMP Version: 3]
   Type: Membership Query (0x11)
   Max Resp Time: 1.0 sec (0x0a)
   Header checksum: 0xeb25 [correct]
   Multicast Address: 239.101.10.46 (239.101.10.46)
▶  QRV=2 S=SUPPRESS router side processing
   QQIC: 60
   Num Src: 0
```

Figure 12 : Message Membership Query

Champ	Valeur	Remarque
Src	172.16.0.1	Adresse IP du routeur
Dst	239.101.10.46	Adresse IP du groupe multicast
Type	Membership Query	-
Multicast Address	239.101.10.46	Adresse IP du groupe multicast

Confirmation d'abonnement à un groupe Multicast

En réponse aux Membership Query, les clients doivent confirmer leur adhésion au groupe multicast en question. Ceci s'effectue à l'aide du message Membership Report suivant :

```
Internet Protocol Version 4, Src: 192.168.1.135 (192.168.1.135), Dst: 224.0.0.22 (224.0.0.22)
Internet Group Management Protocol
   [IGMP Version: 3]
   Type: Membership Report (0x22)
   Header checksum: 0xf06e [correct]
   Num Group Records: 1
▼  Group Record : 239.101.10.46  Mode Is Exclude
      Record Type: Mode Is Exclude (2)
      Aux Data Len: 0
      Num Src: 0
      Multicast Address: 239.101.10.46 (239.101.10.46)
```

Figure 13 : Confirmation d'adhésion

Champ	Valeur	Remarque
Src	192.168.1.135	Adresse IP du client
Dst	224.0.0.22	Groupe multicast des routeurs IGMPv3
Type	Membership Report	-
Record Type	Mode Is Exclude	Confirmation d'adhésion au groupe
Multicast Address	239.101.10.46	Adresse du groupe

Si les abonnés venaient à ne pas répondre aux messages Membership Query du routeur, un timeout va automatiquement arrêter la diffusion du flux.

General Membership Query

Les routeurs peuvent également faire une requête générale pour tout le sous-réseau, afin de connaître l'état des abonnements aux différents groupes. Ce type de message s'appelle General Membership Query :

```
Internet Protocol Version 4, Src: 172.16.0.1 (172.16.0.1), Dst: 224.0.0.1 (224.0.0.1)
Internet Group Management Protocol
    [IGMP Version: 3]
    Type: Membership Query (0x11)
    Max Resp Time: 10.0 sec (0x64)
    Header checksum: 0xec5f [correct]
    Multicast Address: 0.0.0.0 (0.0.0.0)
  ▶ QRV=2 S=Do not suppress router side processing
    QQIC: 60
    Num Src: 0
```

Figure 14 : Requête de demande des adhésions en cours

Champ	Valeur	Remarque
Src	172.16.0.1	Adresse IP du routeur
Dst	224.0.0.1	Tous les périphériques du réseau
Type	Membership Query	-
Multicast Address	0.0.0.0	Tous les groupes Multicast

Concaténation de plusieurs Group Record dans un message

Depuis la version 3 de IGMP, il est possible de concaténer plusieurs Group Records dans un seul et unique message. Pour illustrer cet exemple, voici un message d'une Set-Top box changeant de chaîne TV :

```
Internet Protocol Version 4, Src: 192.168.1.135 (192.168.1.135), Dst: 224.0.0.22 (224.0.0.22)
Internet Group Management Protocol
    [IGMP Version: 3]
    Type: Membership Report (0x22)
    Header checksum: 0xe3c2 [correct]
    Num Group Records: 2
  ▼ Group Record : 239.101.10.47  Change To Include Mode
      Record Type: Change To Include Mode (3)
      Aux Data Len: 0
      Num Src: 0
      Multicast Address: 239.101.10.47 (239.101.10.47)
  ▼ Group Record : 239.101.10.64  Change To Exclude Mode
      Record Type: Change To Exclude Mode (4)
      Aux Data Len: 0
      Num Src: 0
      Multicast Address: 239.101.10.64 (239.101.10.64)
```

Group Record 1, désabonnement du groupe multicast 239.101.10.47

Group Record 2, abonnement au groupe multicast 239.101.10.64

Figure 15 : Concaténation de plusieurs messages

Champ	Valeur	Remarque
Src	192.168.1.135	Adresse IP du client
Dst	224.0.0.22	Groupe multicast des routeurs IGMPv3
Type	Membership Report	-
Num Group Records	2	Nombre de Groups records inclus

3.1.2.4 IGMP Snooping

Par défaut toutes les trames reçues par un switch à destination d'une adresse MAC multicast sont broadcastées sur tous les ports *(sauf celui par lequel elles ont été reçues)*. En effet, la table CAM ne peut pas associer une adresse MAC multicast à un port, celle-ci désignant un groupe et non pas une machine comme en unicast.

Pour éviter que le sous-réseau tout entier soit inondé par des flux multicast auxquels seuls quelques clients sont abonnés, le principe de l'IGMP Snooping a été mis en place. Il consiste à dire aux switchs d'écouter les messages adressés au groupe multicast 224.0.0.22 et ainsi construire une table de corrélation entre les flux multicast et les ports sur lesquels des clients y sont abonnés *(member ports)*. Cette opération peut être effectuée au niveau logiciel sur des switchs d'accès ou directement au niveau matériel *(sur un hardware ASIC)* pour les switchs de distribution, leur permettant de réduire la charge CPU dédiée pour cette tâche.

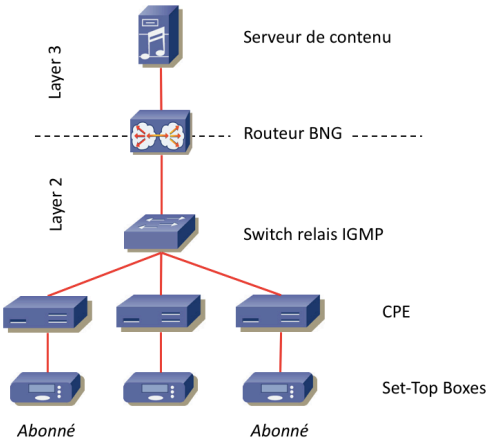


Figure 16 : Multicast sans IGMP Snooping

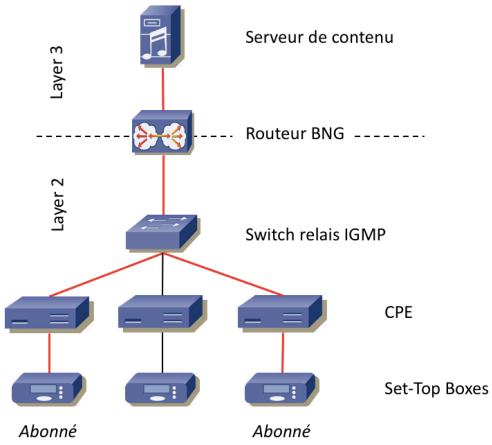


Figure 17 : Multicast avec IGMP Snooping

3.1.2.5 Particularités IGMPv2

La version 2 d'IGMP diffère par rapport à la version 3 sur quelques points, bien que les fondamentaux soient les mêmes :

- Impossibilité de sélectionner des sources spécifiques à l'abonnement à un groupe. Seule la distribution de type Any Source Multicast est prise en charge.
- Les rapports Membership Report sont envoyés directement à l'adresse du groupe Multicast et non pas à 224.0.0.22.
- Le désabonnement à un groupe Multicast s'effectue à l'aide d'un message IGMPv2 Leave Group et non pas Membership Report.
- Un seul abonné par sous-réseau répond au message Membership Query en IGMPv2 pour confirmer son adhésion alors que tous répondent en IGMPv3. Les autres ignorent simplement le message.

A noter également que l'IGMP Snooping est beaucoup moins optimisé en IGMPv2. Du fait que les messages d'adhésion aux groupes sont envoyés directement aux adresses multicast de ces derniers, les switchs doivent "snooper" tous les paquets multicasts. Ceci représente une charge de travail énorme et croît linéairement au nombre de flux transitant, ce qui conduit rapidement les switchs à un effondrement des performances. En IGMPv3 par contre, les switchs n'ont qu'à "snooper" les trames à destination de 01:00:5e:00:00:16 *(224.0.0.22)* pour maintenir la table.

3.1.2.6 CGMP

Le protocole propriétaire Cisco CGMP a été mis en place entre autre pour améliorer le problème de performances lié au Snooping sur IGMPv1 et IGMPv2. Il permet aux switchs et routeurs d'échanger entre eux des messages de signalisation et ainsi d'activer le snooping dynamiquement sur un switch selon que des flux multicasts sont envoyés par un voisin ou pas.

Selon le guide des bonnes pratiques de Cisco ce protocole doit être activé sur un réseau de distribution IGMP version 1 et 2, mais pas nécessairement sur IGMPv3 pour les raisons décrites précédemment.

3.1.3 PIM (Couche 3)

Par principe la fonction clé du routage est de déterminer l'interface de sortie d'un paquet selon son adresse IP de destination. C'est en effet ce qui est appliqué pour un trafic de type unicast.

En multicast cette logique est inversée. L'adresse de destination désigne un groupe et non plus un périphérique cible ; il est donc nécessaire de router les paquets en fonction de leur source.

Pour réaliser cette tâche, le protocole de routage PIM a été spécialement conçu pour les architectures multicast et fonctionne indépendamment du protocole de routage unicast.

3.1.3.1 Rôles du protocole de routage multicast

Le rôle principal du protocole de routage multicast est de créer et maintenir à jour un arbre de distribution des flux. Le routage est ainsi orienté connexion *(un client doit d'abord se "connecter" à une source)*.

Son second rôle est d'informer les autres routeurs sur les sources actives et les groupes présents dans les différents segments du réseau.

Par définition, ces deux points doivent pouvoir être adaptés et recalculés automatiquement de manière rapide lors d'un changement de topologie.

3.1.3.2 Types de protocoles de routage multicast

Mode dense

Ce mode est traduit en français par *"inondation et élagage*". Comme son nom l'indique, il consiste tout d'abord à inonder le réseau puis ensuite élaguer les branches non optimales :

1. Chaque routeur diffuse le trafic multicast à tous ses voisins sauf par l'interface sur laquelle il l'a reçue.
2. Lorsqu'un routeur sait qu'aucune de ses branches ne contient de client abonné au groupe en question, il demande un élagage au routeur adjacent.

Ce mode permet de diffuser très rapidement le flux, mais n'est cependant pas optimal en terme de charge réseau.

Les protocoles de routage utilisant ce mode sont entre autre PIM-DM, MOSPF ou DVMPR.

Mode épars

Ce second mode consiste à attendre une demande avant de router le flux multicast par le chemin emprunté par le message. Cette opération est réalisée par un routeur de périphérie, relayant un message IGMP Membership Report au travers du protocole de routage multicast.

Ce mode est beaucoup plus optimisé en terme de performance et d'efficacité que le premier mais demande légèrement plus de temps pour router le flux multicast depuis la source vers la destination.

Les protocoles de routage utilisant ce mode sont entre autre PIM-SM, OCBT ou QoSMIC.

3.1.3.3 L'algorithme RPF

Pour construire les arbres de distribution et s'assurer que les paquets soient relayés par le chemin le plus court, le protocole de routage PIM utilise l'algorithme RPF *(Reverse Path Forwarding)*. Son fonctionnement est le suivant :

1. Sur chaque interface, par laquelle le routeur reçoit un flux multicast, extraction de l'adresse IP source.
2. Si le chemin le plus court vers la source passe par l'interface de réception du paquet, le flux est considéré comme optimal.
   Sinon, un élagage de la branche est réalisé *(Prune)* et une jointure *(Join)* est envoyée à la source via le chemin le plus court.

Sur l'illustration ci-dessous, le flux violet ne représente pas le chemin optimal. L'algorithme RPF va donc effectuer un élagage de cette branche.

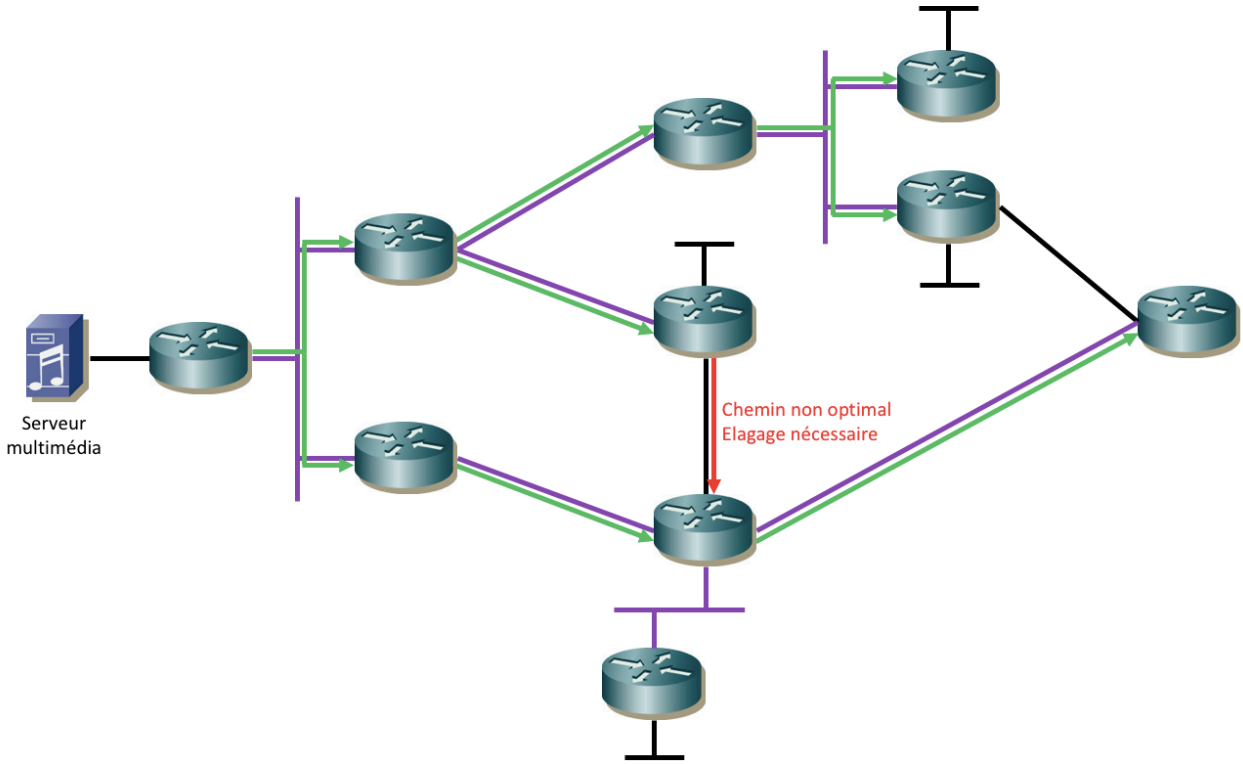


Figure 18 : Exemple d'application de l'algorithme RPF

3.1.3.4 Arbres de distribution

Dans une topologie de routage multicast, deux arbres de distribution sont possibles.

Arbre basé sur la source

Cette première topologie consiste à envoyer depuis le routeur de périphérie la requête *Join* à la source, en utilisant le routage unicast traditionnel.

Cette solution a l'avantage de créer directement le chemin le plus optimal entre une source et les abonnés à un flux multicast. Cependant, la charge nécessaire en terme de mémoire est relativement élevée.

Arbres partagés

Cette seconde topologie fait intervenir un routeur RP *(Rendez-vous Point)*. Celui-ci peut être défini statiquement ou élu dynamiquement *(plus haute adresse IP)*. Il fait office de point de contact pour tous les routeurs désirant s'abonner à un groupe multicast.

L'établissement du flux depuis l'envoi d'une demande *Join* du routeur de périphérique est le suivant :

1. Le routeur de périphérie reçoit une requête d'abonnement à un groupe multicast en IGMP. Celui-ci envoie une requête *Join* au routeur RP en utilisant le routage unicast.
2. Le RP établit une branche jusqu'au routeur de périphérie en utilisant le même chemin qu'utilisé pour la demande.
3. La source émet les données à destination du groupe multicast en question. Le flux est temporairement encapsulé en unicast à destination du routeur RP.
   En parallèle, une requête de type *Register* à destination du router RP est envoyée depuis le routeur de périphérie derrière lequel se trouve la source.
4. Le routeur RP renvoie en sens inverse une requête *Join*.
5. Le flux multicast arrive au routeur RP, celui-ci envoie donc une requête *Register Stop* indiquant qu'il n'a plus besoin du flux unicast temporaire.

Le flux multicast est maintenant établi entre la source et la destination en passant par le routeur RP. L'algorithme RPF est ensuite activé permettant d'établir un lien plus direct s'il en a la possibilité.

Cette solution est plus avantageuse en terme de ressources mémoires sur les routeurs de périphérie, n'ayant que le routeur RP à contacter. Cependant, l'algorithme RPF est nécessaire pour obtenir le chemin le plus optimal. Cette topologie a également l'inconvénient d'introduire un délai supplémentaire pour l'établissement d'un chemin de par le nombre d'opérations requises.

3.1.4 RTSP

RTSP *(Real Time Streaming Protocol)* est un protocole applicatif permettant de contrôler un flux streaming. Il contient des méthodes similaires aux systèmes VCR telles que *Play*, *Pause* et *Record* avec lesquelles il est possible de piloter le serveur de diffusion.

La diffusion du contenu multimédia lui-même ne fait pas partie du protocole RTSP *(plan de contrôle)*. Celui-ci est donc couplé avec les protocoles du plan utilisateur tels que RTP/RTCP ou RDT.

3.1.4.1 Méthodes RTSP disponibles

De manière analogue au protocole SIP, RTSP définit une série de méthodes :

Méthode	Sens	Description
DESCRIBE	C > S	Demande de la description du média
ANNOUNCE	C > S	Envoi de la description du média
	C < S	Annonce de mise à jour de la description d'un média en live
GET_PARAMETER	C<>S	Demande de paramètres sur un média
OPTIONS	C > S	Demande des types de requêtes acceptées par le serveur
	C < S	Envoi au client des types de requêtes acceptées
PAUSE	C > S	Demande de mise en pause du média
PLAY	C > S	Demande de jouer le média
RECORD	C > S	Demande d'enregistrement du média
REDIRECT	C < S	Indication d'un nouveau serveur auquel le client doit se connecter
SETUP	C > S	Spécification du plan utilisateur *(encodage, port, etc…)*
SET_PARAMETER	C<>S	Définition de paramètres sur un média
TEARDOWN	C > S	Demande de terminer proprement une session

Tableau 4 : Méthodes RTSP

Définition des termes :

- Media : Contenu multimédia accessible à l'aide d'une URI RTSP
- Description : Ensemble de paramètres décrivant un média *(p.ex. titre, auteur, etc…)*
- Paramètre : Champ de la description

3.1.4.2 Machine d'état

Le protocole RTSP est basé sur une machine d'états :

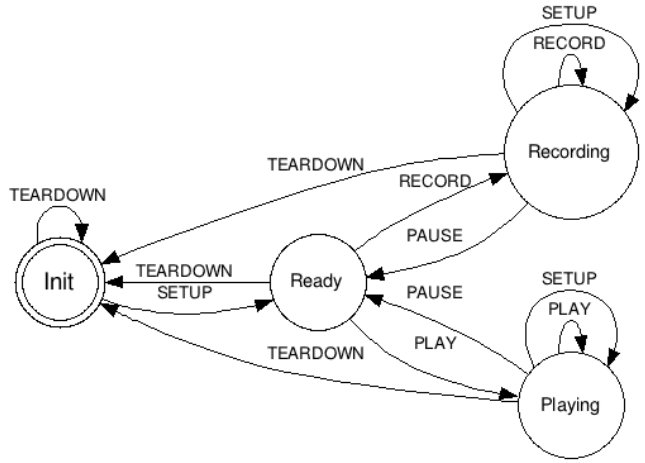


Figure 19 : Machine d'état RTSP complète

Dans le cas de ce travail de semestre, le simulateur ne prend pas en compte la notion d'enregistrement.

Sur les anciennes Set-Top Box possédant un disque dur, le flux multicast était enregistré localement puis rejoué ultérieurement depuis le disque. Sur les boîtiers plus récents sans disques durs, l'enregistrement n'a plus lieu chez le client mais directement auprès de l'opérateur. Le fait de revisionner un enregistrement génère un flux unicast identique à la TV Replay.

La machine d'états peut donc être simplifiée ainsi :

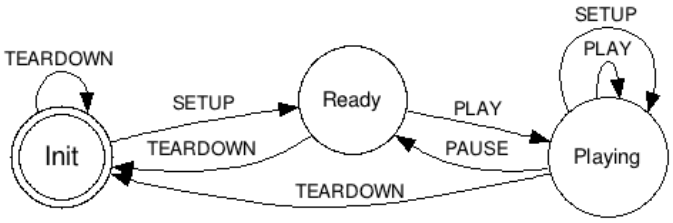


Figure 20 : Machine d'états de RTSP

A noter que d'après la RFC la première méthode appelée de la part d'un client doit impérativement être SETUP. C'est dans celle-ci que sont définis les ports d'émission du flux côté serveur et de réception côté client.

La machine d'état est gérée au niveau du serveur et les différentes instances des clients sont identifiées à l'aide de numéros de session.

3.1.4.3 Structure des messages RTSP

Les informations relatives aux méthodes RTSP sont transmises au format SDP. De plus, toutes sont systématiquement acquittées par un message 200 OK.

Afin d'illustrer les différentes structures des messages, des captures ont été effectuées directement avec la Set-Top Box de net+ FR.

Méthode SETUP

La méthode SETUP permet d'initialiser le plan utilisateur. Il est nécessaire de l'appeler avant chaque diffusion d'un nouveau flux *(p.ex. changement de chaîne Replay)*.

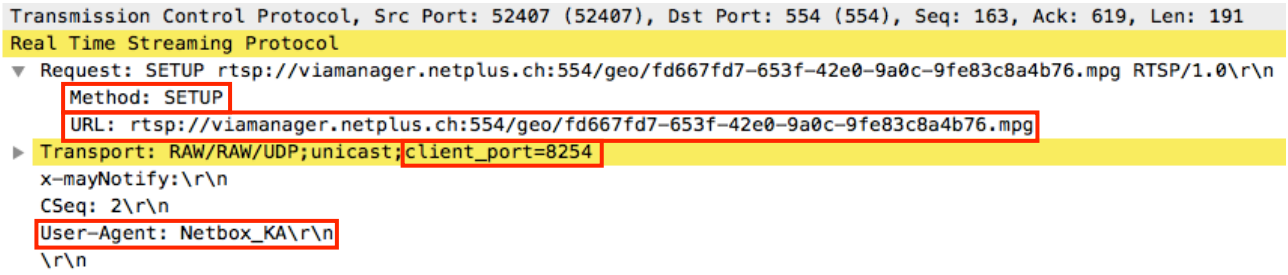


Figure 21 : Méthode RTSP Setup

Champ	Valeur	Remarque
Method	SETUP	-
URL	rtsp://viamanager.netplus…	URL du flux de replay
User-Agent	Netbox_KA	User Agent de la Set-Top Box net+
Client_port	8254	Port à utiliser pour le flux RTP

Ce message est acquitté de la part du serveur par une réponse de type 200 OK.

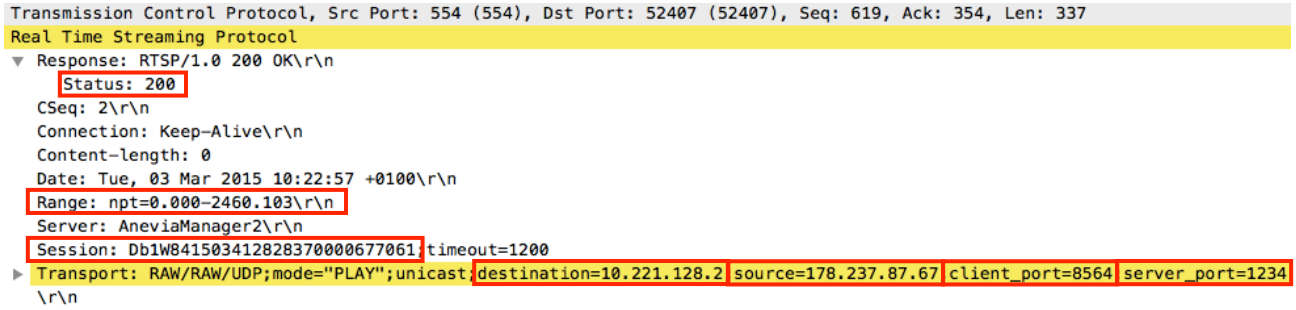


Figure 22 : RTSP Setup OK

Champ	Valeur	Remarque
Status	200	200 OK
Range	0.000-2460.103	Durée du flux
Session	Db1W84150…	Identifiant de la session
Destination	10.221.128.2	Adresse de destination du flux RTP *(IP publique du CPE client)*
Source	178.237.87.67	Adresse de source du flux RTP *(Serveur multimédia de net+)*
Client_port	8564	Port de destination du flux RTP
Server_port	1234	Port de source du flux RTP

Méthode PLAY

Cette méthode indique au serveur d'envoyer en streaming unicast le contenu passé en paramètre.

```
Transmission Control Protocol, Src Port: 52407 (52407), Dst Port: 554 (554), Seq: 354, Ack: 956, Len: 185
Real Time Streaming Protocol
▼ Request: PLAY rtsp://viamanager.netplus.ch:554/geo/fd667fd7-653f-42e0-9a0c-9fe83c8a4b76.mpg RTSP/1.0\r\n
    Method: PLAY
    URL: rtsp://viamanager.netplus.ch:554/geo/fd667fd7-653f-42e0-9a0c-9fe83c8a4b76.mpg
  Range: npt=0.000-\r\n
  CSeq: 3\r\n
  Session: Db1W8415034128283700006770 61
  User-Agent: Netbox_KA\r\n
  \r\n
```

Figure 23 : Méthode RTSP Play

Champ	Valeur	Remarque
Method	PLAY	-
URL	rtsp://viamanager.netplus…	URL du flux de replay
Range	0.000-	Emplacement temporel de diffusion *(du début à la fin dans ces cas)*
Session	Db1W84150…	Identifiant de la session
User-Agent	Netbox_KA	User Agent de la box net+

Le message d'acquittement est le suivant :

```
Transmission Control Protocol, Src Port: 554 (554), Dst Port: 52407 (52407), Seq: 956, Ack: 539, Len: 223
Real Time Streaming Protocol
▼ Response: RTSP/1.0 200 OK\r\n
    Status: 200
  CSeq: 3\r\n
  Connection: Keep-Alive\r\n
  Content-length: 0
  Date: Tue, 03 Mar 2015 10:22:57 +0100\r\n
  Range: npt=0.000-2460.103\r\n
  Scale: 1\r\n
  Server: AneviaManager2\r\n
  Session: Db1W8415034128283700006770 61;timeout=1200
  \r\n
```

Figure 24 : RTSP Play OK

Champ	Valeur	Remarque
Status	200	200 OK
Range	0.000-2460.103	Plage de diffusion du flux
Scale	1	Vitesse de streaming *(1=normal)*
Session	Db1W84150…	Identifiant de la session

Méthode PAUSE

Cette méthode permet au client de mettre en pause la diffusion actuelle d'un flux.

```
Transmission Control Protocol, Src Port: 36604 (36604), Dst Port: 554 (554), Seq: 418, Ack: 506, Len: 168
Real Time Streaming Protocol
▼ Request: PAUSE rtsp://viamanager.netplus.ch:554/geo/557e44f1-d683-471a-b08e-b4f92d0d31a6.mpg RTSP/1.0\r\n
    Method: PAUSE
    URL: rtsp://viamanager.netplus.ch:554/geo/557e44f1-d683-471a-b08e-b4f92d0d31a6.mpg
  CSeq: 25\r\n
  Session: vYDXK1150341282836000067685O
  User-Agent: Netbox_KA\r\n
  \r\n
```

Figure 25 : Méthode RTSP Pause

Champ	Valeur	Remarque
Method	PAUSE	-
URL	rtsp://viamanager.netplus…	URL du flux de replay
Session	vYDXK11503…	Identifiant de la session
User-Agent	Netbox_KA	User Agent de la box net+

Ce message est acquitté par la réponse 200 OK suivante :

```
Transmission Control Protocol, Src Port: 554 (554), Dst Port: 36604 (36604), Seq: 506, Ack: 586, Len: 217
Real Time Streaming Protocol
▼ Response: RTSP/1.0 200 OK\r\n
    Status: 200
  CSeq: 25\r\n
  Connection: Keep-Alive\r\n
  Content-length: 0
  Date: Tue, 03 Mar 2015 10:20:45 +0100\r\n
  Range: npt=209.683-13905.587\r\n
  Server: AneviaManager2\r\n
  Session: vYDXK1150341282836000067685O;timeout=1200
  \r\n
```

Figure 26 : RTSP Pause OK

Champ	Valeur	Remarque
Status	200	200 OK
Range	209.683-13905.587	Emplacement temporel de la pause *(emplacement_pause-duree_totale)*
Session	vYDXK11503…	Identifiant de la session

Dans ce cas de figure, le serveur mémorise l'emplacement temporel de la pause *(première valeur du champ Range)* et l'associe à la session. Au prochain appel de la méthode PLAY du client, le serveur va automatiquement commencer la diffusion à partir du dernier emplacement mémorisé.

Ci-après, voici le message correspondant à la réponse du serveur après que le client ait rappuyé sur le bouton Play de sa télécommande.

```
Transmission Control Protocol, Src Port: 554 (554), Dst Port: 36604 (36604), Seq: 991, Ack: 1003, Len: 227
Real Time Streaming Protocol
▼ Response: RTSP/1.0 200 OK\r\n
     Status: 200
  CSeq: 27\r\n
  Connection: Keep-Alive\r\n
  Content-length: 0
  Date: Tue, 03 Mar 2015 10:21:19 +0100\r\n
  Range: npt=209.683-13939.202\r\n
  Scale: 1\r\n
  Server: AneviaManager2\r\n
  Session: vYDXK11503412828360000676850;timeout=1200
  \r\n
```

Figure 27 : RTSP Replay OK

Les champs sont identiques à la réponse d'un message PLAY standard. On notera cependant une différence dans le champ Range. La valeur de la première partie *(emplacement du début de la diffusion)* correspond à l'endroit où la pause a eu lieu.

Méthode TEARDOWN

A l'extinction de la Set-Top Box, l'appel de la méthode TEARDOWN permet d'indiquer au serveur que le client se déconnecte et qu'il peut ainsi supprimer la machine d'états relative à la session.

```
Transmission Control Protocol, Src Port: 52407 (52407), Dst Port: 554 (554), Seq: 775, Ack: 1451, Len: 170
Real Time Streaming Protocol
▼ Request: TEARDOWN rtsp://viamanager.netplus.ch:554/geo/fd667fd7-653f-42e0-9a0c-9fe83c8a4b76.mpg RTSP/1.0\r\n
    Method: TEARDOWN
    URL: rtsp://viamanager.netplus.ch:554/geo/fd667fd7-653f-42e0-9a0c-9fe83c8a4b76.mpg
  CSeq: 5\r\n
  Session: Db1W8415034128283700000677061
  User-Agent: Netbox_KA\r\n
  \r\n
```

Figure 28 : Méthode RTSP Teardown

Champ	Valeur	Remarque
Method	TEARDOWN	-
URL	rtsp://viamanager.netplus…	URL du flux de replay
Session	Db1W84150…	Identifiant de la session
User-Agent	Netbox_KA	User Agent de la box net+

Cette méthode est finalement acquittée par le message suivant :

```
Transmission Control Protocol, Src Port: 554 (554), Dst Port: 52407 (52407), Seq: 1451, Ack: 945, Len: 186
Real Time Streaming Protocol
▼ Response: RTSP/1.0 200 OK\r\n
    Status: 200
  CSeq: 5\r\n
  Connection: Keep-Alive\r\n
  Content-length: 0
  Date: Tue, 03 Mar 2015 10:23:26 +0100\r\n
  Server: AneviaManager2\r\n
  Session: Db1W8415034128283700000677061 timeout=1200
  \r\n
```

Figure 29 : RTSP Teardown OK

Champ	Valeur	Remarque
Status	200	200 OK
Session	Db1W84150…	Identifiant de la session

Méthode GET_PARAMETER

Ce message permet au client d'interroger le serveur pour récupérer des informations sur le flux en cours de diffusion. Dans un but de synchronisation, les Set-Bop Box telles que celles de net+ demandent régulièrement l'emplacement temporel de la diffusion *(appelé position)*.

```
Transmission Control Protocol, Src Port: 52407 (52407), Dst Port: 554 (554), Seq: 539, Ack: 1179, Len: 236
Real Time Streaming Protocol
▼ Request: GET_PARAMETER rtsp://viamanager.netplus.ch:554/geo/fd667fd7-653f-42e0-9a0c-9fe83c8a4b76.mpg RTSP/1.0\r\n
    Method: GET_PARAMETER
    URL: rtsp://viamanager.netplus.ch:554/geo/fd667fd7-653f-42e0-9a0c-9fe83c8a4b76.mpg
  CSeq: 4\r\n
  Session: Db1W8415034128283700006770 61
  User-Agent: Netbox_KA\r\n
  Content-type: text/parameters
  Content-length: 10
  \r\n
▼ Line-based text data: text/parameters
    position\r\n
```

Figure 30 : Méthode RTSP GET_PARAMETER

Champ	Valeur	Remarque
Method	GET_PARAMETER	-
URL	rtsp://viamanager.netplus…	URL du flux de replay
Session	Db1W84150…	Identifiant de la session
User-Agent	Netbox_KA	User Agent de la box net+
Parameters	position	Demande de la position actuelle

Ce message est acquitté par la méthode 200 OK, contenant la valeur des paramètres demandés, en plus des en-têtes SDP usuelles :

```
Transmission Control Protocol, Src Port: 554 (554), Dst Port: 52407 (52407), Seq: 1179, Ack: 775, Len: 272
Real Time Streaming Protocol
▼ Response: RTSP/1.0 200 OK\r\n
    Status: 200
  CSeq: 4\r\n
  Connection: Keep-Alive\r\n
  Content-length: 17
  Content-type: text/parameters
  Date: Tue, 03 Mar 2015 10:22:57 +0100\r\n
  Range: npt=0.099-2460.103\r\n
  Scale: 1\r\n
  Server: AneviaManager2\r\n
  Session: Db1W8415034128283700006770 61 timeout=1200
  \r\n
▼ Line-based text data: text/parameters
    position: 0.099\r\n
```

Figure 31 : RTSP Get_parameter OK

Champ	Valeur	Remarque
Status	200	200 OK
Range	0.099-2460.103	Etendue temporelle de diffusion *(emplacement_actuel-duree_totale)*
Scale	1	Vitesse de streaming *(1=normal)*
Session	Db1W84150…	Identifiant de la session
Position	0.99	Position temporelle actuelle

Méthode DESCRIBE

Cette méthode client permet de demander au serveur des informations sur un média en question. D'après les mesures effectuées sur la Set-Top Box de net+, ce message n'est jamais généré. Cependant, il est possible de le capturer à l'aide du logiciel VLC.

```
Transmission Control Protocol, Src Port: 14096 (14096), Dst Port: 554 (554), Seq: 162, Ack: 221, Len: 187
Real Time Streaming Protocol
▼ Request: DESCRIBE rtsp://viamanager.netplus.ch:554/pvrreadonly/test_content_vds20.mpg RTSP/1.0\r\n
    Method: DESCRIBE
    URL: rtsp://viamanager.netplus.ch:554/pvrreadonly/test_content_vds20.mpg
  CSeq: 3\r\n
  User-Agent: LibVLC/2.1.5 (LIVE555 Streaming Media v2014.05.27)\r\n
  Accept: application/sdp\r\n
  \r\n
```

Figure 32 : Méthode RTSP DESCRIBE

Champ	Valeur	Remarque
Method	DESCRIBE	-
URL	rtsp://viamanager.netplus…	URL du flux de replay de test
User-Agent	LibVLC/2.1.5	User Agent de VLC
Accept	application/sdp	Format de réponse accepté

Ce message est acquitté par une réponse 200 OK comme suit :

```
Transmission Control Protocol, Src Port: 554 (554), Dst Port: 14096 (14096), Seq: 221, Ack: 349, Len: 433
Real Time Streaming Protocol
▼ Response: RTSP/1.0 200 OK\r\n
    Status: 200
  CSeq: 3\r\n
  Connection: Keep-Alive\r\n
  Content-Base: rtsp://viamanager.netplus.ch:554/pvrreadonly/test_content_vds20.mpg/\r\n
  Content-length: 182
  Content-type: application/sdp
  Date: Tue, 03 Mar 2015 11:44:33 +0100\r\n
  Server: AneviaManager2\r\n
  \r\n
▼ Session Description Protocol
    Session Description Protocol Version (v): 0
  ▶ Owner/Creator, Session Id (o): - 1234567 0 IN IP4 0.0.0.0
    Session Name (s): 1234567
    Session Information (i): RTS Deux HD
  ▶ Connection Information (c): IN IP4 0.0.0.0
  ▶ Bandwidth Information (b): AS:10412
  ▶ Session Attribute (a): control:*
  ▶ Session Attribute (a): charset:UTF-8
  ▶ Session Attribute (a): range:npt=0-24359.6
  ▶ Media Description, name and address (m): video 0 UDP 33
  ▶ Media Attribute (a): fmtp:33 video=5;
```

Figure 33 : RTSP Describe OK

Champ	Valeur	Remarque
Status	200	200 OK
Content-Base	rtsp://viamanager.netplus…	Média concerné par le message
SDP Content	-	Attributs du média

Méthode OPTIONS

Ce type de message permet au client de demander quelles méthodes sont supportées par le serveur. Les Set-Top Box de net+ FR étant configurées et adaptées spécialement pour le serveur de net+, elles n'ont pas besoin de faire ce type de requête. Cependant, cette méthode peut être capturée avec un logiciel générique tel que VLC :

```
Transmission Control Protocol, Src Port: 14098 (14098), Dst Port: 554 (554), Seq: 1, Ack: 1, Len: 161
Real Time Streaming Protocol
▼ Request: OPTIONS rtsp://viamanager.netplus.ch:554/pvrreadonly/test_content_vds15.mpg RTSP/1.0\r\n
    Method: OPTIONS
    URL: rtsp://viamanager.netplus.ch:554/pvrreadonly/test_content_vds15.mpg
  CSeq: 2\r\n
  User-Agent: LibVLC/2.1.5 (LIVE555 Streaming Media v2014.05.27)\r\n
  \r\n
```

Figure 34 : Méthode RTSP OPTIONS

Champ	Valeur	Remarque
Method	OPTIONS	-
URL	rtsp://viamanager.netplus…	URL du flux de replay de test
User-Agent	LibVLC/2.1.5	User Agent de VLC

Ce message est suivi d'une réponse du serveur de type 200 OK, listant toutes les méthodes qu'il prend en charge :

```
Transmission Control Protocol, Src Port: 554 (554), Dst Port: 14098 (14098), Seq: 1, Ack: 162, Len: 220
Real Time Streaming Protocol
▼ Response: RTSP/1.0 200 OK\r\n
    Status: 200
  CSeq: 2\r\n
  Connection: Keep-Alive\r\n
  Content-length: 0
  Date: Tue, 03 Mar 2015 11:44:58 +0100\r\n
  Public: SETUP, TEARDOWN, PLAY, PAUSE, RECORD, OPTIONS, GET_PARAMETER, PING, DESCRIBE\r\n
  Server: AneviaManager2\r\n
  \r\n
```

Figure 35 : RTSP Options OK

Champ	Valeur	Remarque
Status	200	200 OK
Public	SETUP, TEARDOWN, …	Méthodes acceptées par le serveur
Serveur	Anevia Manager 2	Type de serveur

D'après ces informations, on peut voir que toutes les méthodes actuellement supportées par le serveur ont été décrites dans cette analyse.

A noter que la méthode PING n'est pas décrite dans la RFC. Il s'agit probablement d'une implémentation particulière du serveur Anevia Manager utilisée à des fins de tests ou diagnostiques. Aucun message de ce type n'a été capturé en utilisation normale de la Set-Top Box de net+ FR.

Avancer ou reculer rapidement

Il n'y a pas de méthodes spécifiques pour avancer ou reculer rapidement dans une diffusion en cours *(boutons FF et REW des télécommandes)*. Il s'agit simplement d'un message PLAY, avec une valeur du champ *Scale* particulière :

Valeur	Désignation
<1	Retour en arrière
0	*Non implémenté*
1	Vitesse normale
>1	Avance rapide

Selon l'implémentation du serveur, les valeurs pour avancer/reculer plus ou moins rapidement peuvent différer. Afin de garantir une compatibilité entre plusieurs Set-Top Box, la RFC préconise d'arrondir les valeurs intermédiaires vers le mode implémenté le plus proche.

Ci-dessous, un exemple d'un retour en arrière effectué depuis une Set-Top Box de net+ FR :

```
Transmission Control Protocol, Src Port: 36604 (36604), Dst Port: 554 (554), Seq: 1240, Ack: 1496, Len: 181
Real Time Streaming Protocol
▼ Request: PLAY rtsp://viamanager.netplus.ch:554/geo/557e44f1-d683-471a-b08e-b4f92d0d31a6.mpg RTSP/1.0\r\n
    Method: PLAY
    URL: rtsp://viamanager.netplus.ch:554/geo/557e44f1-d683-471a-b08e-b4f92d0d31a6.mpg
  Scale: -5.00\r\n
  CSeq: 29\r\n
  Session: vYDXK11503412828360000676850
  User-Agent: Netbox_KA\r\n
  \r\n
```

Figure 36 : RTSP retour en arrière

En plus des en-têtes SDP usuelles, on distingue la valeur du champ *Scale*, égale à -5.

Le serveur répond à l'aide de la méthode 200 OK, indiquant l'emplacement à partir duquel le retour à lieu :

```
Transmission Control Protocol, Src Port: 554 (554), Dst Port: 36604 (36604), Seq: 1496, Ack: 1421, Len: 228
Real Time Streaming Protocol
▼ Response: RTSP/1.0 200 OK\r\n
    Status: 200
  CSeq: 29\r\n
  Connection: Keep-Alive\r\n
  Content-length: 0
  Date: Tue, 03 Mar 2015 10:21:30 +0100\r\n
  Range: npt=220.415-13950.418\r\n
  Scale: -5\r\n
  Server: AneviaManager2\r\n
  Session: vYDXK11503412828360000676850;timeout=1200
  \r\n
```

Figure 37 : RTSP Retour en arrière OK

Méthode ANNOUNCE

Les méthodes suivantes ne sont pas implémentées sur le serveur de net+ FR. Ainsi, elles seront dorénavant basées sur les échanges décrits dans la RFC.

Le message ANNOUNCE permet à un client d'annoncer un nouveau contenu à disposition du serveur :

```
C->S: ANNOUNCE rtsp://example.com/media.mp4 RTSP/1.0
      CSeq: 7
      Date: 23 Jan 1997 15:35:06 GMT
      Session: 12345678
      Content-Type: application/sdp
      Content-Length: 332

      v=0
      o=mhandley 2890844526 2890845468 IN IP4 126.16.64.4
      s=SDP Seminar
      i=A Seminar on the session description protocol
      u=http://www.cs.ucl.ac.uk/staff/M.Handley/sdp.03.ps
      e=mjh@isi.edu (Mark Handley)
      c=IN IP4 224.2.17.12/127
      t=2873397496 2873404696
      a=recvonly
      m=audio 3456 RTP/AVP 0
      m=video 2232 RTP/AVP 31
```

Figure 38 : Méthode RTSP ANNOUNCE

Champ	Valeur	Remarque
Method	ANNOUNCE	-
URL	rtsp://example.com/media.mp4	URL d'un flux d'exemple
SDP Content	v=0, o=mhandley 289…	Description du média

Cette méthode est acquittée par le serveur à l'aide d'un simple message 200 OK, avec aucun paramètre particulier :

```
S->C: RTSP/1.0 200 OK
      CSeq: 7
```

Figure 39 : RTSP Announce OK

La méthode ANNOUNCE peut également être utilisée dans le sens Serveur à Client, pour annoncer une modification *(p. ex. changement de codec)* dans le flux actuellement en cours de diffusion.

Méthode REDIRECT

Similaire au message 301 du protocole HTTP, la méthode REDIRECT permet au serveur d'indiquer au client qu'un nouveau serveur est disponible et doit être dorénavant contacté. Un champ du format SDP permet d'indiquer si cette redirection est temporaire ou définitive.

La redirection peut avoir lieu au cours d'une diffusion. Si tel est le cas, le champ *Range* indique l'emplacement temporel du changement.

Le message 301 est le suivant :

```
S->C: REDIRECT rtsp://example.com/media.mp4 RTSP/1.0
      CSeq: 11
      Location: rtsp://bigserver.com:8001
      Range: clock=19960213T143205Z-
```

Figure 40 : Méthode RTSP REDIRECT

Champ	Valeur	Remarque
Method	REDIRECT	-
URL	rtsp://example.com/media.mp4	URL d'un flux d'exemple
Location	rtst://bigserver.com:8001	URL du nouveau serveur à contacter
Range	19960213T143205Z-	Estempille temporelle du changement au format clock

Ce message envoyé du serveur ne nécessite pas d'acquittement de la part du client.

Méthode RECORD

La méthode RECORD permet à un client RTSP d'envoyer au serveur l'ordre d'enregistrer le média indiqué.

Celle-ci est très ouverte et peu spécifiée ; son utilisation dépend beaucoup de l'implémentation du serveur de contenus multimédia. Selon les cas, il peut être nécessaire de spécifier un ID du flux à enregistrer, un timestamp avec une durée ou encore le lien du média tout simplement si ceux-ci sont générés selon un algorithme prédictif. Il est donc nécessaire de connaître la structure des requêtes avant de pouvoir implémenter cette méthode.

Une demande spécifiant le flux multimédia pourrait être décrite ainsi :

```
C->S: RECORD rtsp://example.com/media.mp4 RTSP/1.0
      CSeq: 6
      Session: 12345678
```

Figure 41 : Méthode RTSP RECORD

Champ	Valeur	Remarque
Method	RECORD	-
URL	rtsp://example.com/med…	URL du flux de à enregistrer

Selon la RFC, le serveur doit acquitter le message à l'aide d'un message 200 OK au minimum.

```
S->C: RTSP/1.0 200 OK
      CSeq: 6
      Session: 12345678
```

Figure 42 : RTSP Record OK

Méthode SET_PARAMETER

La méthode SET_PARAMETER permet de définir un paramètre de transmission actuellement négocié. Son format est le suivant :

```
C->S: SET_PARAMETER rtsp://example.com/media.mp4 RTSP/1.0
      CSeq: 10
      Content-length: 20
      Content-type: text/parameters

      barparam: barstuff
```

Figure 43 : Méthode RTSP SET_PARAMETER

Champ	Valeur	Remarque
Method	SET_PARAMETER	-
URL	rtsp://example.com/media.mp4	URL d'un flux d'exemple
Content-type	text/parameters	Type de codage pour le contenu
Text content	barparam:barstuff	Attribution de la valeur `barstuff` au paramètre `barparam`

Si le paramètre est accepté, le message est acquitté par une réponse de type 200 OK :

```
S->C: RTSP/1.0 200 OK
      CSeq: 7
```

Figure 44 : RTSP Set_parameter OK

Cependant en cas d'erreur, le message 451 Invalid Parameter suivant est retourné, avec le paramètre en question :

```
S->C: RTSP/1.0 451 Invalid Parameter
      CSeq: 10
      Content-length: 10
      Content-type: text/parameters

      barparam
```

Figure 45 : RTSP Set_parameter NOK

3.2 Outils existants

Avant de concevoir le simulateur de Set-Top Box, une analyse préliminaire doit être réalisée afin de déterminer les outils existants sur le plan matériel et logiciel. Ainsi, des éléments pourront être repris afin de créer le produit final répondant à tous les critères du mandant.

Ce chapitre vise à énumérer puis présenter sous l'aspect pratique les outils existants. Le choix d'utiliser ou non ceux-ci sera fait dans la phase de spécification, au moyen d'un tableau multicritères de décision.

3.2.1 Solutions matérielles

3.2.1.1 Spirent TestCenter Live for IPTV

Spirent est une société multinationale de télécommunications basée en Angleterre comptant aujourd'hui 1'500 employés. Elle créée des éléments réseaux depuis 1936 et s'est entièrement orientée depuis 1997 dans le business des appareils de mesure des réseaux d'opérateurs.

La solution TestCenter Live de Spirent est très complète. Elle permet de réaliser des tests de charge dans un réseau en générant toute sorte de trafic *(TV Live, TV Replay, Voice, Data, etc…)* depuis des éléments hardware simulant des serveurs de contenus multimédia. Conçu pour un réseau à très haute densité, le générateur est capable de provisionner jusqu'à 589'000 abonnés TV Unicast par châssis.

Côté client, des équipements sondes permettent de réceptionner les flux en simulant les Set-Top Box. Dotés d'algorithmes de mesures synchronisés avec les générateurs, ils peuvent retourner des graphiques indiquant l'état de la QoS dans le réseau.

Figure 46 : Générateur Spirent SPT-N11U

Figure 47 : Sonde Spirent TestCenter Live 6500

Toutes les informations de simulation peuvent être visualisées de manière graphique au sein de l'application TestCenter. L'infrastructure Spirent est prévue pour être utilisée en parallèle d'un réseau en production. Il est donc possible de n'utiliser plus que les sondes en exploitation, permettant ainsi de réaliser un monitoring très poussé du réseau ; le générateur étant utilisé que lors des tests de charge.

3.2.1.2 IXIA WaveTest

IXIA est une société américaine de 1'800 employés active depuis 1997 dans la conception d'outils de mesure de sécurité et de performance des réseaux de télécommunications.

L'entreprise ne conçoit pas de produits pour l'IPTV en particulier. Cependant le but premier de ce projet étant de valider la charge induite par *n* Set-Top Box derrière un CPE de net+ FR, un générateur de flux IP Multicast et Unicast permet de réaliser cette opération.

Le châssis WaveTest 90 pour les grands opérateurs respectivement WaveTest 20 pour les plus petits, permet de générer *n* flux IP Unicast ou Multicast en parallèle. Sur le même outil, une interface graphique est disponible afin de consulter le résultat des mesures au travers des valeurs de QoS telles que la gigue, la latence ou encore le taux de perte de paquets.

Figure 48 : Générateur WaveTest 90

Figure 49 : Générateur WaveTest 20

Cette solution matérielle correspond moins aux besoins du mandant, n'étant pas particulièrement orientée flux IPTV. Cependant, elle pourrait être utilisée dans un cadre plus générique pour réaliser des tests de charge des équipements réseaux en terme de volume de trafic, pour le provisionning des connexions Internet par exemple.

3.2.1.3 NextraGen Packet Raptor

Cet outil matériel n'est pas conçu pour réaliser des tests de charge ou du monitoring comme le permettent les solutions présentées précédemment. Cependant, il peut être intéressant pour visualiser les conséquences d'un affaiblissement de la QoS sur des flux IP.

L'appareil dénommé Packet Raptor se branche en bridge le long d'un câble à paires torsadées et permet de faire varier artificiellement des paramètres tels que la gigue, la latence et le taux de perte de paquets. Ces mesures permettraient de simuler divers cas de figures et ainsi déterminer la qualité de service minimale requise pour un service donné.

Figure 50 : NextraGen Packet Raptor

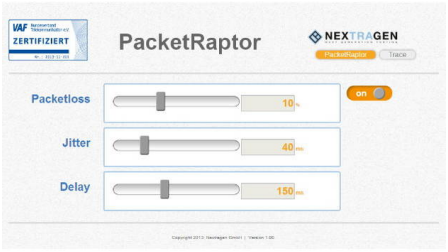


Figure 51 : Interface web de configuration

3.2.2 Solutions logicielles commerciales

3.2.2.1 NextraGen TraceSim

Ce produit de l'entreprise NextraGen est entièrement logiciel. Il est orienté flux multimédias et permet de simuler du trafic audio et vidéo. S'exécutant sur un ordinateur traditionnel, il prend autant le rôle d'un générateur de trafic que d'un client.

Les deux entités client-serveur communiquent entre elles et permettent de mesurer bon nombre de paramètres de QoS traditionnels, mais également quelques-uns plus élaborés comme :

- Taux de paquets n'arrivant pas dans l'ordre
- Taux de paquets dupliqués
- Atténuation de volume le long de la ligne
- Différence du rapport signal sur bruit entre l'émission et la réception

Les paramètres analogiques décrits ci-dessus sont mesurés afin de pouvoir calculer, dans le cas d'un opérateur réalisant des conversions analogiques vers digitales et inversement, la qualité de la ligne.

L'inconvénient de cette jeune solution est que le Multicast n'est pas encore implémenté. Ainsi, seuls les flux IPTV ou VoIP unicast peuvent être simulés.

3.2.2.2 IXIA AppLibrary

Le produit AppLibrary de l'entreprise IXIA est un simulateur logiciel très complet de test de charge d'un réseau. Il est orienté fournisseur d'accès et permet de simuler des flux sur un réseau virtuel.

Son utilisation consiste à définir des sources de trafic *(p.ex. serveurs de contenus multimédia)* ainsi que des clients *(p.ex. Set-Top Box)* disposés au sein du réseau virtuel à la manière du simulateur OPNET.

Cependant le but du projet de semestre étant de tester physiquement la charge d'un CPE, l'étude de cet outil ne va pas être approfondie.

3.2.3 Solutions logicielles libres

3.2.3.1 Scapy

Scapy est un logiciel développé en Python permettant la manipulation de paquets réseaux. Il propose notamment de générer des paquets à partir de la couche 2 *(trames)*, mais également d'en intercepter et de décoder leurs contenus.

Dans le cadre de ce projet, cet outil pourrait permettre de forger des requêtes afin que le simulateur puisse dialoguer avec le serveur multimédia de net+. Des libraires externes permettent de créer simplement des paquets de type Multicast, cependant aucune n'existe à ce jour pour RTSP. Cette problématique pourrait cependant être contournée en modifiant une librairie SIP existante, les deux protocoles étant relativement proches.

L'avantage de Scapy est sa très grande flexibilité dans la forge des paquets. Basé sur le modèle en couches TCP/IP, la création de n'importe quel message est possible. De plus, des méthodes sont disponibles pour calculer automatiquement les champs standardisés des différentes couches comme les checksums par exemple.

De plus, une grande communauté active de développeurs suit les évolutions de Scapy et propose des exemples de réalisations. Le projet étant distribué sous licence GPLv2+, la documentation est ouverte et très riche de par les nombreuses contributions.

Cependant l'outil a un inconvénient de taille : il ne sait pas interpréter les paquets reçus. Dans le cas d'un protocole à états comme RTSP par exemple, cela signifie qu'une machine d'états doit être implémentée du côté du client afin de savoir comment répondre aux messages venant du serveur. Le problème est similaire en multicast, le client doit répondre aux messages IGMP Membership Query émis par le routeur sans quoi la diffusion du flux sera coupée.

Scapy ne fonctionne qu'avec les versions 2.x de Python. Une nouvelle version est actuellement en cours de développement et apportera la compatibilité avec Python 3.x.

A noter également que la licence GPLv2+ de Scapy impose que tout produit final l'intégrant soit lui-même distribué sous licence GPL. Cela signifie que le code source devra être libre et que tout le monde pourra le distribuer et le modifier librement.

3.2.3.2 Twisted

Twisted est un framework Python basé sur un paradigme de programmation événementielle permettant de créer et maintenir une connexion entre deux entités réseaux. Il est notamment utilisé par les clients Dropbox pour gérer les flux de données entre les clients et les serveurs.

Cet outil est complet et convient bien au protocole Multicast ; cependant aucune librairie n'est disponible pour RTSP.

L'inconvénient majeur de ce framework est son jeune âge. Bien que la documentation disponible sur le site officiel soit très riche, il manque une communauté active de développeurs qui documentent leurs problèmes comme on peut trouver avec Scapy.

Cet outil Open-Source est distribué sous la licence MIT.

3.2.3.3 VLC Media Player

VLC Media Player est un lecteur de flux multimédia disponible sur la plupart des systèmes d'exploitation du marché. Utilisé aujourd'hui par plus de 100 millions d'utilisateurs, il s'est rapidement imposé dans son domaine grâce au grand nombre de codecs pris en charge.

Il permet de décoder un flux vidéo depuis de multiples sources *(fichier local, streaming unicast, streaming multicast, etc…)* mais également de pouvoir diffuser sur le réseau un flux de streaming. Cette possibilité est à considérer pour la phase de réalisation de la maquette de tests, permettant de diffuser non seulement sur un groupe multicast à l'aide du protocole RTP mais également en unicast simulant un flux de TV Replay à l'aide du protocole de contrôle RTSP. Ce même outil pourrait remplir la fonction de serveur de contenu multimédia ainsi que de client en simulant une Set-Top Box.

Disponible tout d'abord dans une version GUI, il a par la suite été enrichi d'une interface CLI permettant de réaliser des opérations avancées comme s'abonner à un flux sans le décoder par exemple. Cette possibilité est très intéressante dans le cadre de ce projet, qui permettrait à une machine traditionnelle de s'abonner à *n* flux vidéo sans toutefois que le processeur soit surchargé par le décodage.

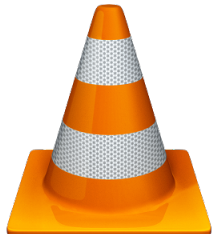


Figure 52 : Logo de VLC

L'application étant distribuée sous licence Open-Source GPLv2+, elle dispose d'une énorme et très active communauté de développeurs. Le web regorge de tutoriels et codes d'exemples afin d'exploiter la plupart des fonctionnalités proposées.

3.2.3.4 vlc.py

Le logiciel VLC étant développé en langages C et C++, il n'est pas très pratique de l'utiliser dans des scripts. Pour pallier à ce problème, les équipes de Video LAN ont développé une passerelle Python appelée *vlc.py* permettant d'interagir directement en Python avec l'API C++ *libvlc* au cœur de VLC.

Tout comme le mode CLI de VLC, la passerelle a l'avantage de pouvoir interagir avec VLC sans charger l'interface graphique. Il est également possible de travailler sur des flux sans devoir y décoder le contenu pour autant *(plan de contrôle uniquement)*.

L'API Python étant relativement récente, elle dispose d'une bonne documentation de type Epydoc mais cependant peu de retours d'expériences et de ressources sur le web.

3.2.3.5 RealPlayer

L'outil RealPlayer est un lecteur multimédia développé par le fournisseur de contenus streaming RealNetworks. Similaire au logiciel VLC Player, il permet de décoder la plupart des codecs libres du marché, ainsi que des codecs propriétaires de la marque tels que RealAudio et RealVideo.

Le logiciel n'est utilisable qu'en version graphique. Une implémentation CLI a été rendue disponible depuis quelques années pour interagir directement avec le framework Helix utilisé au cœur du logiciel et disponible sur la plupart des systèmes d'exploitation du marché.

Figure 53 : Logo de RealPlayer

Helix est actuellement en version 11 et largement soutenu par RealNetworks. L'outil dispose d'une communauté active, mais principalement formée par des développeurs de RealPlayer.

3.2.3.6 Windows Media Player

Le logiciel Windows Media Player est un logiciel propriétaire développé par l'entreprise Microsoft. Principalement optimisé pour le codec WMV *(Windows Media Video)*, il permet de décoder des flux audio et vidéo dans un nombre restreints de codecs libres.

Son code source est fermé et aucune API n'est mise à disposition des développeurs en dehors des technologies propres à Microsoft telles qu'ASP.NET ou VB.NET. L'utilisation du lecteur est restreinte au système d'exploitation Windows, suite à son abandon sur Mac OS en 2008.

L'utilisation du programme en mode pseudo-CLI est limitée à une dizaine de commandes, permettant principalement d'automatiser le lancement du logiciel en mode GUI avec des actions prédéfinies.

Figure 54 : Logo de Windows Media Player

3.2.4 Générateurs de flux et serveurs multimédia

Sur le marché des générateurs de flux et des serveurs multimédia, on trouve beaucoup de produits.

Dans un contexte de test ou de petit réseau ne nécessitant pas une haute disponibilité, on notera la présence des outils suivants :

- **GStreamer :** Framework Open-Source écrit en C, disponible sur de multiples systèmes d'exploitation, permettant de diffuser du contenu multimédia sur un réseau, tant au travers de flux Unicast que Multicast. Très puissant et modulable, il est cependant complexe et nécessite du temps à la prise en main.

- **FFmpeg :** Logiciel Open-Source développé en C, il permet de diffuser du contenu multimédia en Unicast et Multicast. Son utilisation est possible sur la plupart des systèmes d'exploitation du marché, en CLI uniquement.
  Depuis 2010, une segmentation est apparue chez l'éditeur avec les logiciels ffserveur *(produit de streaming original)* et ffplay *(interface C permettant de jouer des flux)* et ffprobe *(outil d'analyse des flux multimédias, sur le plan qualité vidéo)*.

Dans un contexte opérateur, on soulignera la disponibilité des outils suivants :

- **EvoStream :** Solution propriétaire développée par l'entreprise EvoStream, orientée petits et moyens opérateurs permettant la diffusion de flux IPTV Live et Replay. La plupart des protocoles de contrôle traditionnels sont disponibles *(RTSP HTTP, etc…)* ainsi que de nombreux codecs afin de s'adapter à un maximum de plateformes différentes *(mobiles, TVs, navigateurs, etc..)*.
  L'interface d'administration étant réalisée en interface graphique, des fonctions évoluées telles que la répartition de charge et la facturation du contenu par exemple peuvent être mises en place simplement.

- **Anevia Manager :** Plateforme propriétaire également, elle est orientée moyennes et grandes entreprises. Elle permet de monter une infrastructure complexe de diffusion de contenus multimédia en live avec des interfaces de retransmission des signaux satellites, mais également le couplage de baies de stockage pour la gestion des enregistrements et des flux de rediffusion.

  Cette solution est actuellement utilisée par l'opérateur valaisan net+, fournisseur des services IPTV de l'entreprise net+ FR.

3.3 Conclusion

Cette analyse a permis de décrire les principes fondamentaux de la distribution *Live* et *Replay* des flux de télévision dans les différentes couches du réseau.

Les protocoles du plan de contrôle ont ensuite été étudiés depuis des captures provenant directement de la net+ Box. Cela permettra de valider la conformité des messages du simulateur vis-à-vis de la Set-Top Box.

Finalement, l'analyse détaillée des outils disponibles sur le marché permettra de faciliter le choix du langage et de la plateforme de développement en phase de spécifications.

4 Spécifications

Les spécifications ont pour but de décrire le comportement et le fonctionnement du simulateur de Set-Top Box. A l'aide des informations récoltées durant l'analyse, un choix du langage de programmation ainsi que des librairies et outils existants sera réalisé.

4.1 Interface logicielle

4.1.1 Objectif primaire

Dans un premier temps, le simulateur sera développé pour un usage en mode CLI. En appelant le programme un fichier de configuration sera passé en paramètre, contenant toutes les options d'exécution.

4.1.1.1 Interface CLI

Le scénario d'exécution de l'interface CLI est schématisé ainsi :

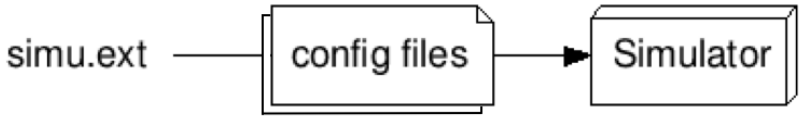


Figure 55 : Schéma de lancement du simulateur

En cours d'exécution, le simulateur affiche simplement dans la console le nombre de flux multicast respectivement unicast acquis auprès du serveur multimédia :

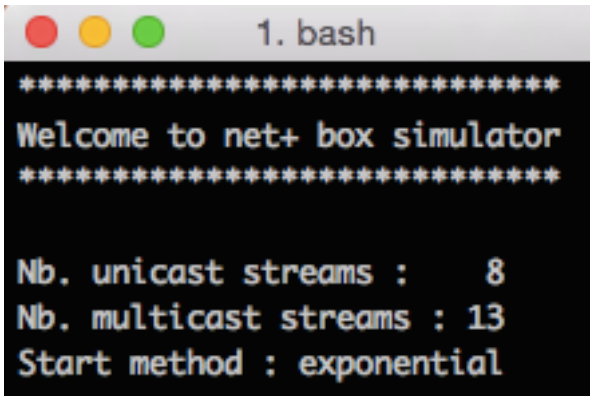


Figure 56 : Schéma d'exécution du simulateur

Pour terminer une simulation en cours, l'utilisateur devra arrêter l'exécution du programme en appuyant simultanément sur les touches CTRL+C.

Un paramètre pourra également être défini pour indiquer une durée de simulation.

Le simulateur devra impérativement terminer proprement toutes les connexions unicast et multicast afin de ne pas laisser de flux fantômes.

4.1.2 Fichiers de configuration

Deux fichiers de configuration seront passés en paramètre au lancement du simulateur. Ce type d'implémentation permet une grande évolutivité et l'ajout aisé de nouvelles options ultérieurement sans se retrouver avec une longue liste de paramètres.

4.1.2.1 Paramètres du fichier général

Ce premier fichier comporte les paramètres globaux de l'application :

Paramètre	Type de valeurs	Description
`list_multicast`	Array[Integer;String]	Liste des flux multicast à disposition avec une pondération de priorité
`list_unicast`	Array[Integer;String]	Liste des contenus replay à disposition avec une pondération de priorité

Tableau 5 : Description du fichier de configuration général

4.1.2.2 Paramètres du fichier de simulation

Ce second fichier de configuration contient les paramètres propres à la simulation en cours :

Paramètre	Type de valeurs	Description
`nb_multicast`	Integer {0;+*n*}	Nombre de flux multicast à s'abonner
`nb_unicast`	Integer {0;+*n*}	Nombre de flux unicast à jouer
`start_type`	Integer {-1;0;+*n*}	Type de modèle pour le lancement des flux : - 1 = Exponentiel 0 = Instantané +*n* = Linéaire d'un delta *n [ms]*
`stop_type`	Integer {-1;0;+*n*}	Type de modèle pour l'arrêt des flux : - 1 = Exponentiel 0 = Instantané +*n* = Linéaire d'un delta *n [ms]*
`simu_duration`	Integer {0 ;+*n*}	Durée de la simulation à partir du moment où tous les flux sont démarrés : 0 = Infini +n = Durée de simulation *n* [sec]
`zapping_enable`	Boolean	Activation de la simulation du zapping sur les threads multicast *(changements de chaînes à intervalles aléatoires)*
`pause_enable`	Boolean	Activation de la simulation de pauses sur les threads unicast *(Pauses et Play des flux à intervalles aléatoires)*
`channel_select`	String	Type de modèle pour la sélection des flux unicast et multicast : Random = Choix aléatoire pondéré Fixed = Choix type liste circulaire

Tableau 6 : Description du fichier de configuration de la simulation

4.1.2.3 Format des fichiers

Les fichiers de configuration seront au format standard Linux, soit des couples paramètre-valeur séparés par des lignes, qui seront parsés avec le langage de programmation :

```
# This is a comment
param1 value
param2 value
```

Figure 57 : Format du fichier de configuration

Le mandant souhaite pouvoir générer les fichiers de configuration à l'aide d'Excel. Une marche à suivre décrivant les opérations d'exportation sera jointe en annexe au rapport final.

4.1.3 Logging des évènements

Tous les composants du simulateur écriront dans un fichier log leurs actions respectives *(abonnement à un groupe Multicast, mise en pause d'un flux Replay, etc…)* liées à une estampille temporelle de type Timestamp. Cela permettra de ne pas surcharger la console mais également de pouvoir garder une trace précise de la simulation.

4.1.4 Options secondaires

4.1.4.1 Interface graphique

Une option qui a été ajoutée en objectif secondaire consiste à réaliser une interface graphique au lieu du fichier de configuration afin de pouvoir exécuter le simulateur en mode GUI.

Une description précise des interfaces sera réalisée si le temps permet son implémentation.

4.1.4.2 Abstraction des Set-Top Box

La modélisation devra pouvoir être adaptée à une abstraction supplémentaire permettant d'appliquer un profil de simulation par Set-Top Box et non plus pour une simulation généralisée.

4.1.4.3 Calcul de la qualité de service

La qualité de service peut être calculée sur la base de paramètres tels que la latence, la gigue et le taux de perte de paquets.

Ce point sera spécifié plus tard au cours du projet en fonction du temps restant et de l'intérêt du mandant vis-à-vis des autres options secondaires.

4.2 Architecture logicielle

Le simulateur devant être générique, une architecture multithread a été définie comme suit :

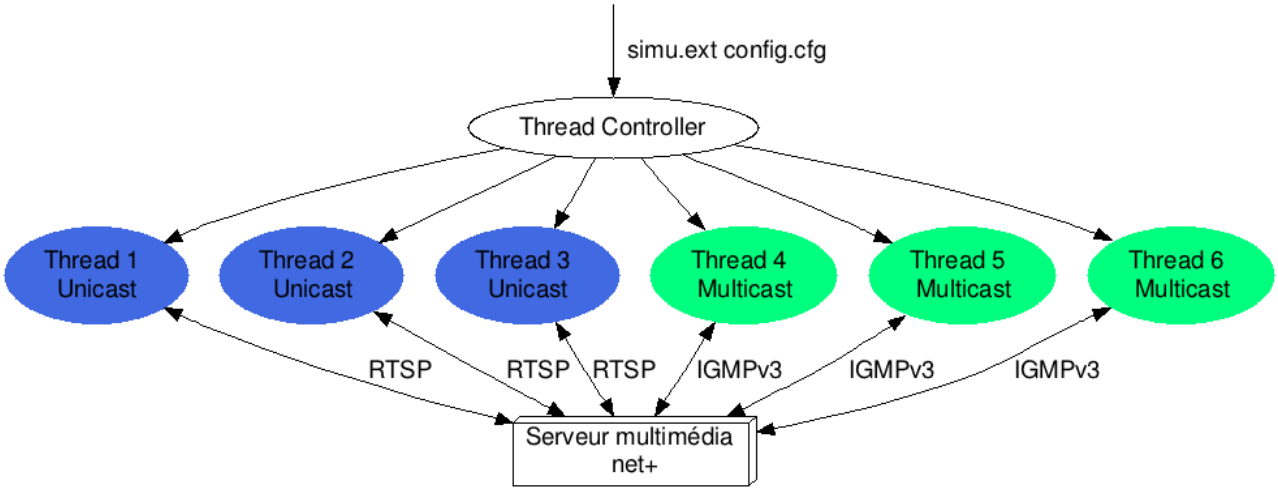


Figure 58 : Architecture multithread

Cette architecture permet au simulateur de bénéficier d'une grande évolutivité. Il est aujourd'hui question de gérer des flux IPTV unicast et multicast mais de nouveaux types de threads pourraient être très facilement implémentés pour d'autres catégories de trafic comme l'Internet Data ou la VoIP par exemple.

Dans ce contexte, un thread désigne l'instance d'une classe exécutée en parallèle à d'autres instances et communiquant avec le serveur multimédia de net+ afin de s'abonner à des flux unicast respectivement multicast.

4.2.1 Types de threads

Thread Controller

Il s'agit du thread principal du simulateur qui pilote l'ensemble du logiciel. Il est parent de tous les threads de simulation et s'occupe des interactions avec ceux-ci, dont notamment leurs créations et leurs destructions tout en fermant les connexions en cours lorsque l'utilisateur souhaite arrêter une simulation.

Thread IPTV Unicast

Ce type de thread a pour but de demander au serveur de contenu multimédia à l'aide du protocole RTSP la diffusion unicast d'un flux de rediffusion ; simulant ainsi la partie TV Replay d'une Set-Top Box. Le flux de données RTP n'est pas décodé ni même traité afin de ne pas surcharger le processeur.

Thread IPTV Multicast

Cet autre type de thread permet de s'abonner à un flux multicast à l'aide du protocole IGMP ; simulant cette fois-ci la partie TV Live d'un Set-Top Box. Tout comme le thread unicast, il ignore le flux de données reçues.

4.3 Comportements

La description du comportement des différents éléments du simulateur au travers de diagrammes de séquences permet de définir le squelette des classes.

4.3.1 Classe Controller

Cette classe est exécutée automatiquement au démarrage du simulateur et effectue les opérations suivantes :

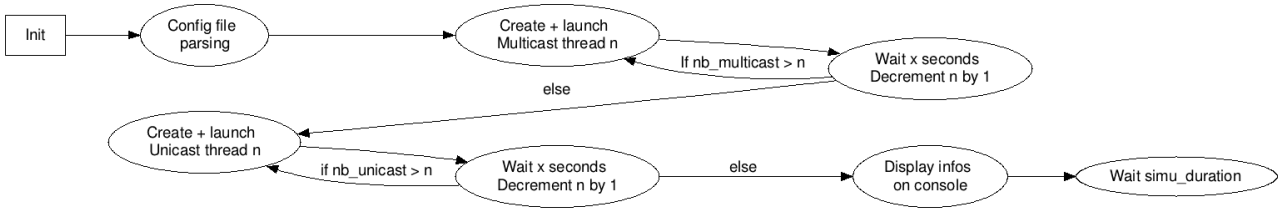


Figure 59 : Comportement de la classe Controller au démarrage

Ensuite, le programme attend la durée de simulation passée en paramètre dans le fichier de configuration avant d'appeler la méthode stop() permettant d'arrêter tous les threads selon l'algorithme spécifié. Cette même méthode est également appelée en cas d'arrêt forcé (CTRL+C).

Le séquencement de la méthode stop() de la classe Controller est le suivant :

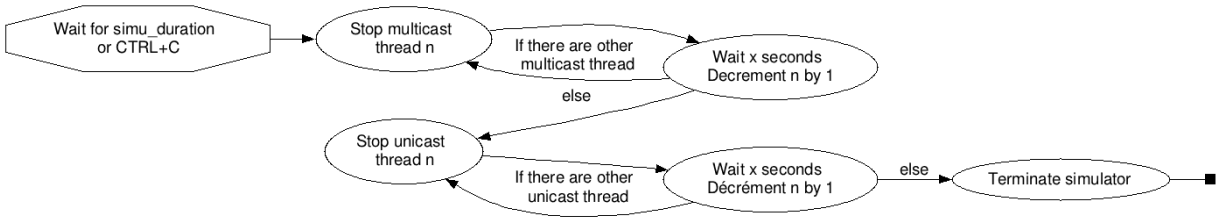


Figure 60 : Phase d'arrêt de la classe Controller

4.3.2 Classe Multicast

Cette classe est instanciée *n* fois par son parent *(Controller)* et exécutée sous forme d'un thread ; *n* étant le nombre de flux multicast à simuler indiqué dans le fichier de configuration.

Son comportement est le suivant :

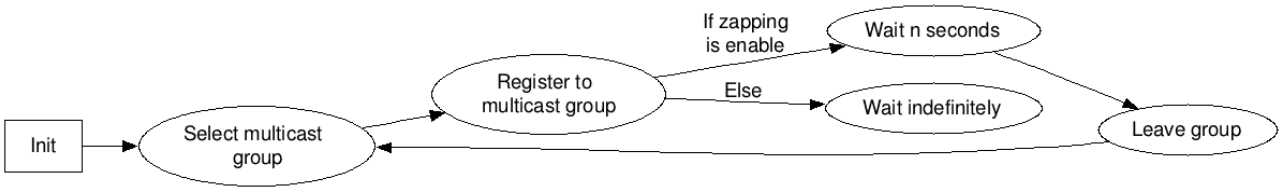


Figure 61 : Comportement de la classe Multicast

Cette routine peut être interrompue à tout moment par l'appel de la méthode stop() depuis la classe parente :

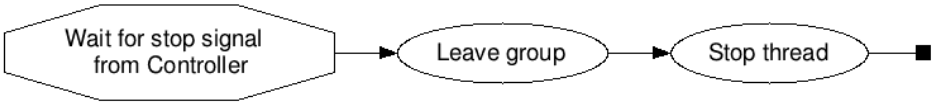


Figure 62 : Méthode d'arrêt de la classe Multicast

4.3.3 Classe Unicast

Cette classe est instanciée *n* fois par la classe parent Controller et exécutée sous forme d'un thread *(n étant le nombre de flux unicast à simuler)*. Son comportement est le suivant :

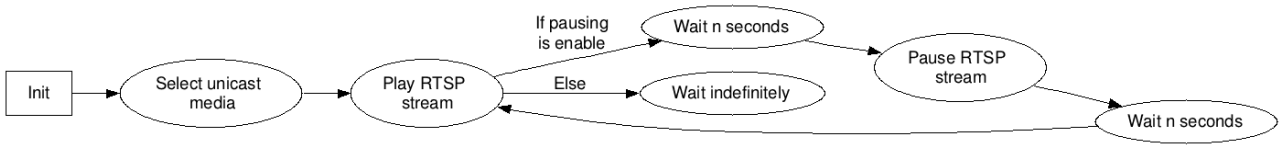


Figure 63 : Comportement de la classe Unicast

Sur le même principe que la classe précédente, le Thread Unicast peut être interrompu à tout moment par l'appel de sa méthode stop() :

Figure 64 : Méthode d'arrêt de la classe Unicast

4.4 Choix du langage et des outils externes

Basé sur le chapitre d'analyse de ce projet, deux tableaux multicritères de décision ont été réalisés pour choisir le langage de développement du simulateur ainsi que les librairies et outils utilisés.

4.4.1 Langage de programmation

Les contraintes du simulateur vis-à-vis du langage de programmation sont les suivantes :

- Disponibilité de librairies et d'outils réseaux tiers
- Possibilité d'exécution CLI et GUI
- Compatibilité avec les systèmes d'exploitation Linux, Windows et OS X
- Facilité de prise en main pour de futures modifications
- Disponibilité d'outils de création d'interfaces graphiques

L'analyse multicritère ci-dessous a permis de choisir le langage le plus adapté au projet :

Critères et éléments de décision		Pdt	PHP		Python		Java		C++	
			Pts	Val	Pts	Val	Pts	Val	Pts	Val
01	Disponibilité de librairies et outils tiers	9	6	54	9	81	9	81	7	63
02	Facilité de prise en main *(concepteur)*	6	7	42	8	48	7	42	4	24
03	Facilité de prise en main *(utilisateur)*	7	6	42	7	49	8	56	7	49
04	Compatibilités systèmes d'exploitation	8	8	64	9	72	9	72	3	24
05	Temps nécessaire à la conception	5	4	20	7	35	6	30	4	20
06	Disponibilité fonctions réseaux	9	2	18	8	72	7	63	6	54
07	Possibilité de générer des GUI	7	9	63	7	49	9	63	8	56
08	Possibilité d'exécution en CLI	9	1	9	8	72	6	54	8	72
09	Ressources communautaires	5	7	35	8	40	7	35	7	35
10	Facilité d'installation et portabilité	4	9	36	8	32	8	32	8	32
11	Aide éditeur	6	7	42	7	42	7	42	7	42
	TOTAL			**425**		**592**		**570**		**471**

Tableau 7 : Analyse multicritères du langage

Description des critères

01 Richesse de librairies et outils tiers disponibles avec le langage pour compléter ses possibilités.

02 Aisance de prise en main du langage, du point de vue du concepteur.

03 Aisance de prise en main du langage, du point de vue de l'utilisateur final qui utilisera le simulateur.

04 Compatibilité du langage avec les différents systèmes d'exploitation *(Linux, Windows et OS X)*.

05 Selon les connaissances, temps de prise en main du langage.

06 Possibilité de réaliser des opérations réseaux orientées IPTV.

07 Possibilité de pouvoir générer simplement des interfaces graphiques avec le langage.

08 Possibilité d'utiliser les programmes générés en mode CLI

09 Communauté de la plateforme active, mettant à disposition des ressources et des exemples.

10 Possibilité de porter le simulateur final simplement d'une machine à une autre.

11 Qualité de l'aide mise à disposition de la part l'éditeur, concernant le langage lui-même.

Tableau 8 : Description des critères d'analyse du langage

Synthèse

Python semble le langage idéal pour la réalisation de ce projet. De nature simple et intuitive tout en bénéficiant de fonctionnalités puissantes, il permettra à de futures personnes l'édition simple et rapide du simulateur afin d'ajouter de nouvelles fonctionnalités.

4.4.2 Librairies et outils existants

Un langage de programmation seul ne suffit pas pour réaliser ce projet. Afin de programmer les diverses classes de manière efficace, il est nécessaire de s'appuyer sur des outils et librairies proposant les avantages suivants :

- Possibilité de gérer des flux multicast avec IGMPv3
- Possibilité de demander le Replay des médias vidéo en unicast avec RTSP
- Evolutivité garantie des outils avec une communauté active de développeurs
- Ouverture maximale des codes sources

Le choix des outils a été réalisé à l'aide de l'analyse multicritère suivante :

Critères et elements de décision		Pdt	RealPlayer Helix		Windows Media		VLC		Scapy		Twisted	
			Pts	Val	Pts	Val	Pts	Val	Pts	Val	Pts	Val
01	Ouverture du code	8	3	24	0	0	8	64	8	64	8	64
02	Facilité de prise en main (concepteur)	7	6	42	9	63	7	49	7	49	4	28
03	Facilité de prise en main (utilisateur)	4	5	20	7	28	8	32	8	32	8	32
04	Temps nécessaire à la conception	7	6	42	3	21	5	35	1	7	1	7
05	Compatibilité multicast	10	7	70	10	100	10	100	10	100	10	100
06	Compatibilité unicast	10	10	100	10	100	10	100	10	100	10	100
07	Compatibilité IGMPv3	10	10	100	10	100	10	100	6	60	8	80
08	Compatibilité RTSP	10	5	50	10	100	10	100	0	0	0	0
09	Ressources communautaires	8	4	32	2	16	8	64	9	72	7	56
10	Support de l'éditeur	6	6	36	1	6	7	42	8	48	8	48
11	Disponibilité d'API Python	8	5	40	0	0	8	64	10	80	10	80
TOTAL				**556**		**534**		**750**		**612**		**595**

Tableau 9 : Analyse multicritères des outils et librairies

Description des critères

01 Ouverture du code source de l'outil *(Licence)*

02 Aisance de prise en main de l'outil, du point de vue du concepteur.

03 Aisance de prise en main de l'outil, du point de vue de l'utilisateur final qui utilisera le simulateur.

04 Selon les connaissances, temps de prise en main de l'outil.

05 Compatibilité et prise en charge par l'outil des flux multicast.

06 Compatibilité et prise en charge par l'outil des flux unicast.

07 Compatibilité et prise en charge par l'outil du protocole IGMPv3

08 Compatibilité et prise en charge par l'outil du protocole RTSP.

09 Communauté de la plateforme active, mettant à disposition des ressources et des exemples.

10 Qualité de l'aide mise à disposition de la part du fabriquant, concernant la librairie ou l'outil lui-même.

11 Possibilité d'interaction de l'outil avec le langage Python.

Tableau 10 : Description des critères d'analyse des outils et librairies

Synthèse

Les outils VLC et Scapy sont idéaux. Le premier possède depuis peu une API Python reprennant la plupart des fonctionnalités disponibles dans VLC. Si toutefois une fonction ne serait pas encore accessible en Python, il serait tout de même possible d'y remédier en lançant une commande Shell pour interagir avec VLC en mode CLI.

Le second outil n'est probablement pas nécessaire dans un premier temps. Cependant si d'autres fonctionnalités optionnelles devaient être implémentées et ne seraient pas prises en charge par VLC, il s'agirait d'une excellente alternative pour générer toutes sortes de paquets.

4.5 Conclusion

Ce chapitre a permis de décrire précisément l'architecture et le fonctionnement du simulateur de Set-Top Box.

Il ne reste maintenant plus qu'à coder séquentiellement les différentes classes pour ensuite les assembler et former le logiciel final.

5 Conception

Le chapitre de conception résulte du développement du simulateur, mettant en avant les démarches et les concepts utilisés. Cela permettra à une personne externe de pouvoir reprendre le code en vue d'y ajouter des fonctionnalités plus ciblées pour de nouveaux cas d'utilisation.

5.1 Modélisation

Dans le chapitre de spécifications, l'architecture conceptuelle a été décrite. Sur cette base, il s'agira ensuite de décrire l'architecture logicielle, faisant intervenir les relations entre les différentes classes, librairies externes et modules Python.

5.1.1 Architecture conceptuelle

La modélisation du simulateur est la suivante :

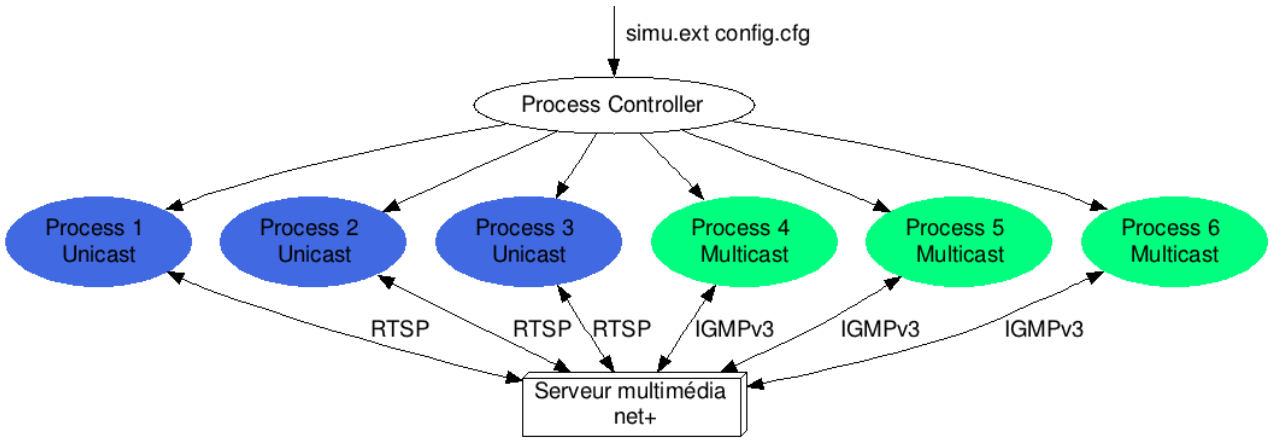


Figure 65 : Modélisation actuelle du simulateur de Set-Top Box

Une Set-Top Box est modélisée soit par un process de type TV Live soit TV Replay, exécutés dans des processus distincts. Il n'est cependant pas possible de passer d'un flux unicast à un flux multicast au sein d'un même processus.

Quant au profil de la simulation, il est appliqué globalement au processus principal dénommé Controller. C'est celui-ci qui va instancier et lancer en parallèle les différentes instances de Set-Top Box.

Chaque processus Unicast ou Multicast communique directement avec le serveur de contenus Multimédia de net+ à l'aide des protocoles RTSP respectivement IGMPv3.

Cette architecture générique confère une grande évolutivité au simulateur. Il sera décrit au terme de ce chapitre les différentes abstractions supplémentaires possibles ainsi qu'évolutions possibles.

5.1.2 Architecture logicielle

Le simulateur étant développé en programmation orientée objet, il fait état de nombreuses dépendances. Le schéma ci-dessous décrit l'architecture logicielle ainsi que les différentes interactions entre les classes, librairies externes et modules Python.

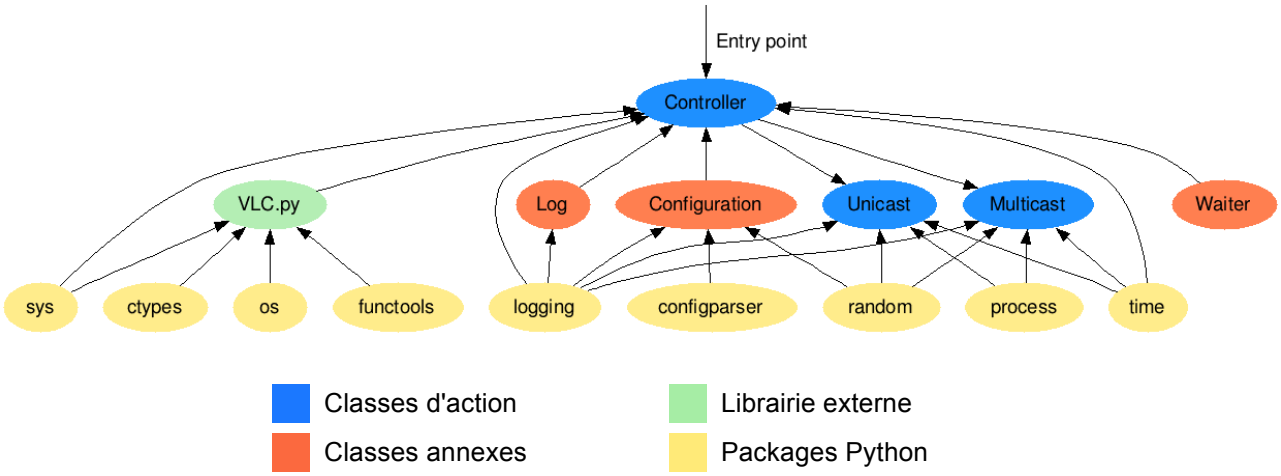


Figure 66 : Diagramme d'interaction des classes

La librairie externe vlc.py étant compatible Python3, le simulateur a été développé sous cette dernière version ; cela permet de lui garantir une longue durée de vie. Si toutefois pour une future évolution il serait nécessaire d'utiliser Python 2.7, seule l'utilisation du package `configparser` nécessite une adaptation *[CON2]*.

5.1.3 Architecture utilisateur

Du point de vue de l'utilisateur du simulateur, l'architecture des fichiers est la suivante :

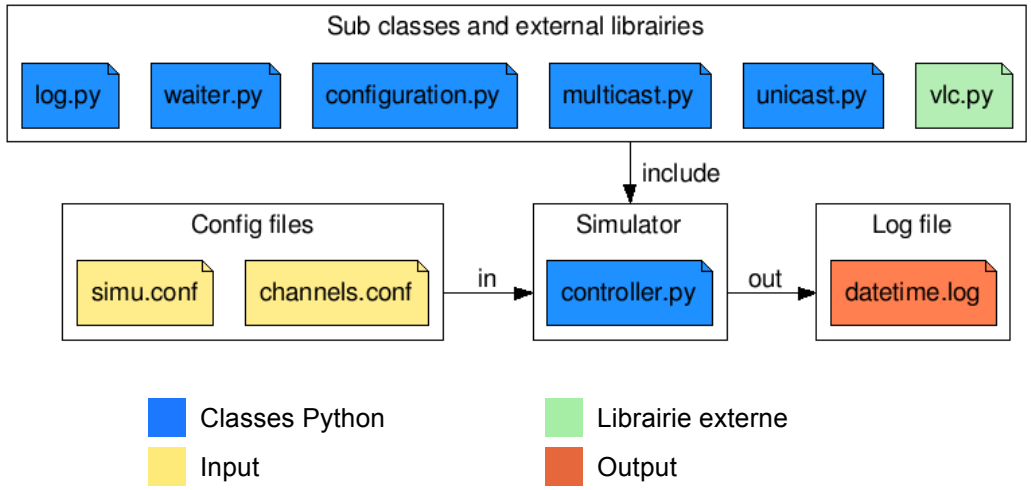


Figure 67 : Diagramme d'interaction des classes

Les fichiers de types classes Python et librairies sont nécessaires à l'exécution du simulateur ; tous doivent être placés à la racine.

L'exécution d'une simulation nécessite de passer en paramètre les deux fichiers de configuration. En les plaçant également à la racine du projet, la commande est la suivante :

```
python3 controller.py simu.conf channels.conf
```

5.2 Dépendances et librairies externes

5.2.1 Multimédia

La librairie multimédia utilisée est vlc.py. Il s'agit de bindings Python permettant d'interagir directement avec le cœur de VLC écrit en C.

Dans le simulateur net+ SimBox, son implémentation est la suivante :

- Inclusion de la librairie vlc.py une seule fois dans la classe Controller
- Création d'une instance VLC dans la classe Controller
- Passage de la référence de l'instance en paramètres aux processus Unicast et Multicast
- Création des média players dans les processus Unicast et Multicast sur la base de l'instance VLC reçue

Afin de ne pas décoder le flux vidéo et ainsi réduire la charge du processeur, l'instance de VLC est créée avec les paramètres suivants : `--vout dummy`

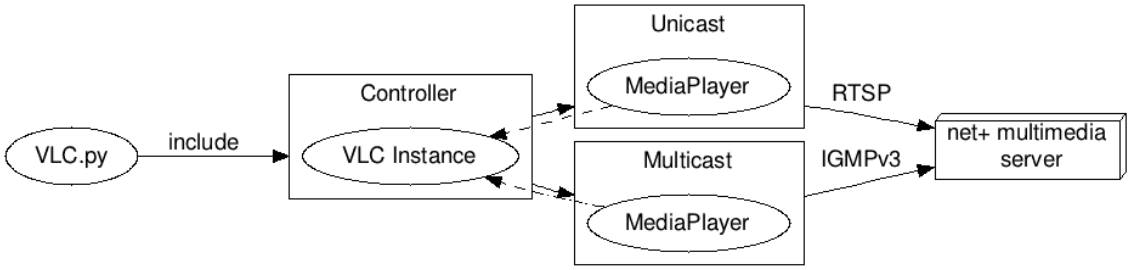


Figure 68 : Architecture d'utilisation de la librairie vlc.py

5.2.2 Logging

Le mandant souhaitant pouvoir logger tout le déroulement de la simulation, la librairie Python Logging a été utilisée de la manière suivante :

- Création de l'instance de la classe Logging avec le nom *simnetbox* dans le constructeur de la classe privée Log. Celle-ci est appelée une fois au lancement de la simulation depuis la classe Controller.
- Chaque classe nécessitant un accès au log peut récupérer l'instance créée à l'aide de la commande suivante : `module_logger = logging.getLogger('netsimbox')`
- Chaque classe peut écrire dans le fichier log à l'aide de la fonction suivante : `module_logger.info("Value_to_log")`

5.2.3 Lecture du fichier de configuration

La lecture du fichier de configuration s'effectue à l'aide de la librairie ConfigParser. Celle-ci a été implémentée de la manière suivante dans le simulateur :

- Une classe privée appelée Configuration a été créée. Elle permet de parser l'ensemble des deux fichiers de configuration dans un dictionnaire à l'appel de son constructeur.
  Cette classe est instanciée une fois par le Controller puis passée en paramètre de chaque process de type Unicast ou Multicast.
- Des accesseurs publics ont été créés dans la classe Configuration permettant de récupérer la valeur des paramètres depuis n'importe quelle classe ayant instancié Configuration.
- D'autres accesseurs ont été créés pour récupérer le nombre d'adresses multicast respectivement le nombre d'URLs unicast renseigné.

5.3 Compatibilité logicielle

La pleine compatibilité du simulateur est garantie pour la configuration logicielle suivante :

OS :	Debian 8.0.0
VLC :	2.2.0
vlc.py	2015-04
Python :	3.4.2
GCC :	4.9.1

Dans le concept du langage Python, la compatibilité des packages utilisés est garantie tout au long d'une version majeure. En l'occurrence, aucun problème de compatibilité ne devrait avoir lieu au fil des mises à jour de Python 3.

L'exécution du simulateur est *possible* sur OS X et Windows moyennant l'installation manuelle de Python 3.4 et VLC.

Quant au serveur de contenus multimédia, celui-ci peut être réalisé indépendamment sur Linux, OS X ou Windows pour les flux Multicast, et Linux et Windows pour les flux Unicast contrôlés à l'aide du protocole RTSP.

5.4 Commentaire et style de codage

Tous les fichiers Python sont certifiés PEP8. Ils respectent ainsi le guide de style de codage universel proposé par Python, permettant ainsi à des tierces personnes de reprendre facilement le code développé.

Les méthodes et différents algorithmes sont ensuite décrits et commentés selon le pattern suivant, afin de permettre de générer éventuellement par la suite une documentation de type Epydoc :

```
"""
Title of the function

Description of the function and the returned values

@type  param_n_name:  Type of nth param
@param param_n_name:  Description of the nth param
"""
```

5.5 Problèmes rencontrés

5.5.1 Threads vs Processus en Python

En Python, tous les threads sont régulés par le Global Interpreter Lock. Initialisé à une valeur de 1, chaque thread désirant se mettre en état *running* doit impérativement acquérir le verrou préalablement. Ce point majeur empêche d'avoir un parallélisme entre les différents threads au profit d'une concurrence, même sur une architecture processeur de type multi-cœur.

Ce problème a été détecté dès le moment où plusieurs flux de type Unicast étaient demandés en concurrence à l'aide de threads Python. A la figure 69 en suivant le flux RTP d'un thread, on peut constater qu'au bout d'un certain temps le serveur s'aperçoit que le client n'écoute plus. Après 5 messages de type *Port Unreachable*, l'émission du flux est suspendue.

108	17.906434006	160.98.31.173	160.98.31.164	RTP	213 PT=DynamicRTP-Type-96, SSRC=0x47E440F, Seq=47598, Time=1439494298
109	17.906435006	160.98.31.173	160.98.31.164	RTP	219 PT=DynamicRTP-Type-96, SSRC=0x47E440F, Seq=47599, Time=1439494298
110	17.906437006	160.98.31.173	160.98.31.164	RTP	245 PT=DynamicRTP-Type-96, SSRC=0x47E440F, Seq=47600, Time=1439494298
111	17.906853006	160.98.31.173	160.98.31.164	RTP	820 PT=DynamicRTP-Type-96, SSRC=0x47E440F, Seq=47601, Time=1439494298
120	17.948665006	160.98.31.173	160.98.31.164	RTP	75 PT=DynamicRTP-Type-96, SSRC=0x47E440F, Seq=47602, Time=1439498049
121	17.948714006	160.98.31.164	160.98.31.173	ICMP	103 Destination unreachable (Port unreachable)
122	17.948820006	160.98.31.173	160.98.31.164	RTP	271 PT=DynamicRTP-Type-96, SSRC=0x47E440F, Seq=47603, Time=1439498049
123	17.948841006	160.98.31.164	160.98.31.173	ICMP	299 Destination unreachable (Port unreachable)
124	17.948919006	160.98.31.173	160.98.31.164	RTP	297 PT=DynamicRTP-Type-96, SSRC=0x47E440F, Seq=47604, Time=1439498049
125	17.948936006	160.98.31.164	160.98.31.173	ICMP	325 Destination unreachable (Port unreachable)
126	17.949068006	160.98.31.173	160.98.31.164	RTP	273 PT=DynamicRTP-Type-96, SSRC=0x47E440F, Seq=47605, Time=1439498049

Figure 69 : Problème de concurence Python

Ce problème a pu être résolu en utilisant des Processus au lieu des Threads. Ils permettent une exécution réellement parallèle distribuée sur tous les cœurs de la machine pour simuler le fait que tous les clients écoutent continuellement les flux émis par le serveur et éviter que ce dernier n'interrompe le flux.

5.5.2 Messages en IGMPv2

Après le démarrage de la machine virtuelle Ubuntu, les messages IGMP étaient envoyés en version 3. A la fin de la simulation après l'envoi du message de type IGMP Membership Request pour se désabonner à un flux Multicast, un équipement réseau du laboratoire envoyait systématiquement en IGMPv2 un message de type IGMP Query. Dès lors, tous les messages émis depuis la machine virtuelle étaient émis en IGMP version 2.

Afin de garantir la compatibilité entre les différentes versions d'IGMP, les systèmes d'exploitation modernes sont configurés pour envoyer par défaut les messages en IGMPv3. Cependant si un quelconque message de type IGMPv2 est reçu sur une interface réseau, l'OS ne communique plus qu'en IGMPv2.

Pour résoudre ce problème l'IGMP a simplement été désactivé sur l'équipement réseau en question, n'ayant aucune utilité pour son fonctionnement.

5.5.3 Non envoi des messages RTSP et IGMPv3

D'après l'implémentation de la librairie VLC, l'envoi des messages d'adhésion aux flux unicast et multicast est déclenché par la méthode `play()` du Média Player. Or cette fonction n'est pas bloquante et prend du temps par rapport à la vitesse du processeur ; il est donc nécessaire d'attendre un certain temps avant de quitter la méthode dans laquelle elle est appelée.

N'ayant pas de règle précise à cet effet, l'implémentation a été la suivante :

- Attente indéfinie après l'appel de la méthode `play()`
- Création d'une méthode `stop()` permettant d'arrêter à tout moment l'attente infinie en quittant le processus

5.5.4 Mise à jour de la librairie vlc.py

Suite à la mise à jour de la librairie vlc.py *(2015-04)*, le comportement de la méthode `play()` a légèrement varié. Pour les flux Multicast, aucun changement n'a été apporté, les échanges étant sans connexion.

Cependant pour les flux Unicast, le mode connecté du protocole RTSP ne permet plus de stopper le flux depuis une autre méthode que celle depuis laquelle il a été créé. Un workaround a donc été trouvé, en n'attendant plus indéfiniment l'appel de la méthode `stop()`, mais en attendant sur un Lock Python qui va déclencher l'arrêt du flux.

5.6 Améliorations futures

5.6.1 Création de scénarios

Une seconde variante possible serait de créer des scénarios de tests, avec plusieurs profils différents.

Grâce à l'implémentation actuelle, ceci pourrait être très simplement réalisé en lançant itérativement *n* fois le simulateur avec des fichiers de configuration différents grâce à une fonction Python telle que `Popen` [CON09].

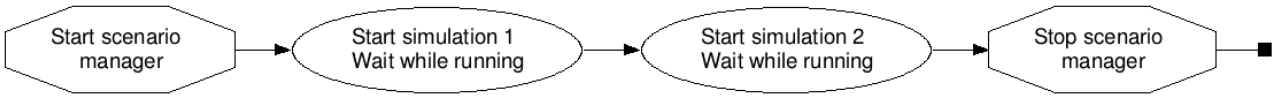


Figure 70 : Illustration du concept de scénarios avec 2 simulations

Une autre variante pour l'implémentation des scénarios consisterait à instancier séquentiellement n fois la classe Controller. Cependant, cette solution requière de surcharger son constructeur afin de pouvoir lui passer les fichiers de configuration en paramètre.

En termes de performances les deux méthodes sont équivalentes, la première étant plus simple à utiliser *(à la façon d'un script séquentiel)* alors que la seconde est dans un état d'esprit orienté objet.

5.6.2 Modélisation des profils des Set-Top Box

Une autre variante pouvait être aisément déployée afin de pouvoir appliquer des profils de simulations différents pour les Set-Top Box serait de réaliser une abstraction supplémentaire. Une Set-Top Box ne serait ainsi plus schématisée par un processus de type Unicast ou Multicast, mais par un processus de type Controller.

Cette implémentation permettrait de simuler des Set-Top Box recevant plusieurs flux en parallèle, comme par exemple l'enregistrement d'une émission simultanément au visionnement d'un flux Replay.

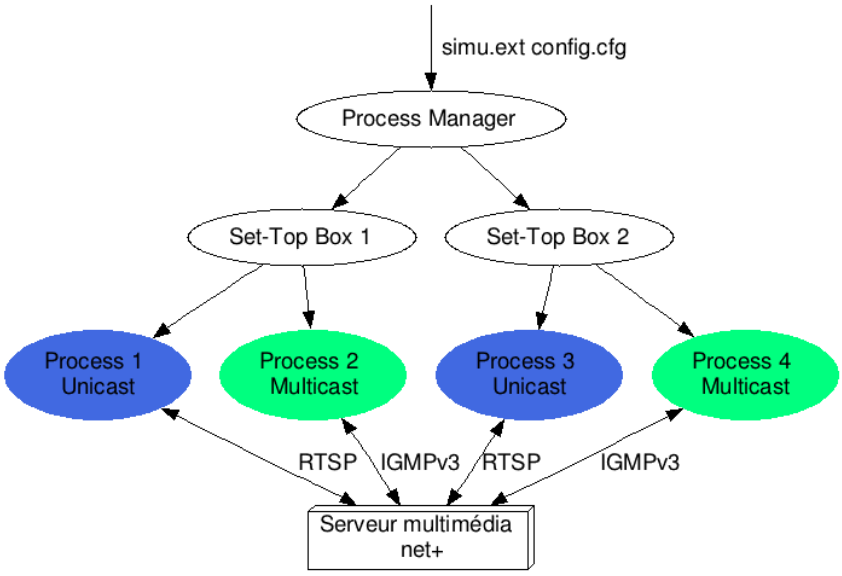


Figure 71 : Variante de modélisation avec des profils différents pour les Set-Top Box

Dans le schéma ci-dessus, seuls deux flux sont illustrés par Set-Top Box. Cependant, il est possible d'en créer *n* de chaque type. Pour relater au plus proche la réalité, il faudrait en cas d'implémentation, définir les limites du système afin de ne pas avoir de simulations totalement irréalistes. L'idéal serait de se baser sur les caractéristiques des net+ box en reprenant le nombre de flux maximal autorisé par client.

5.6.3 Evaluation de la qualité de service

Actuellement le simulateur permet de s'abonner à des flux IPTV Multicast et Unicast. Cependant, rien ne permet encore de déterminer la qualité des flux reçus. Une évolution importante consisterait à étendre les fonctionnalités du simulateur en y ajoutant un calculateur de QoS.

Cette évolution permettrait par exemple de pouvoir déterminer quel est le nombre de flux maximal pouvant être lancé tout en garantissant une qualité correcte pour le client final.

Dans un second temps, il pourrait également être question de logger graphiquement la qualité de service des flux en fonction du temps, afin de visualiser rapidement le seuil critique.

5.6.4 Extraction du niveau de sévérité de logging

Une amélioration mineure serait d'extraire le niveau de sévérité pour le fichier log, défini actuellement en dur dans le code au niveau `INFO`, pour le rendre définissable par l'utilisateur au travers du fichier de configuration.

Il serait également possible d'avoir deux niveaux différents, l'un pour le fichier log et l'autre pour la console Python.

5.6.5 Ajout d'autres types de trafic

Une évolution future du simulateur serait de pouvoir y ajouter d'autres types de trafics. Cette opération pourrait être réalisée très simplement par l'ajout de nouvelles classes à instancier comme processus, au même titre que le Multicast et l'Unicast actuellement.

On notera les deux types de flux suivants :

VoIP :	Flux téléphonique de voix sur IP
Data :	Flux de transmission de données *(upstream et downstream)*

5.6.6 Capture réseau automatique

Pour capturer le trafic réseau, il est nécessaire de lancer une capture Wireshark manuellement et en parallèle à une simulation. Une opportunité future serait d'utiliser une librairie Python telle que *libpcap* [CON10] ou *pypcap* [CON11] afin de capturer et générer automatiquement un fichier `*.pcap` de la simulation.

Cette option pourrait être définie dans le fichier de configuration `simu.conf`, laissant ainsi le choix à l'utilisateur d'activer ou non la capture des flux.

Des options supplémentaires pourraient être également implémentées afin de filtrer directement le trafic désiré et relatif au simulateur.

5.7 Conclusion

Comme précédemment décrit, le simulateur utilise des outils libres et open-sources cités en phase d'analyse. Cette approche confère une plus grande évolutivité au logiciel et une grande facilité d'adaptation par des tierces personnes en vue de l'implémentation de nouvelles fonctionnalités.

La prochaine étape consiste à tester puis valider le simulateur vis-à-vis des spécifications et du cahier des charges initial.

6 Tests et validation

Cette phase vise à valider le bon fonctionnement du simulateur et les résultats obtenus sur la base d'un plan de tests détaillé. Les résultats permettent d'apporter les corrections nécessaires jusqu'à obtenir le comportement désiré tel que défini dans le cahier des charges et la phase de spécifications.

Dans un second temps, des tests de charge seront réalisés afin de définir sur deux configurations matérielles données quelles sont les possibilités et les limites matérielles vis-à-vis du nombre de flux lancé.

6.1 Validation du simulateur

Une liste de tests a été rédigée afin de vérifier le bon fonctionnement du simulateur de Set-Top Box. Une fois les divers tests effectués, les problèmes ont été corrigés et ce de manière récursive jusqu'à l'obtention de la version finale.

6.1.1 Liste des tests

La liste des tests se trouve en annexe. Ceux-ci ont été effectués sur la maquette de simulation décrite en phase de spécifications.

Une marche à suivre décrivant la procédure pour générer des flux unicast et multicast contrôlés par RTSP respectivement IGMP depuis VLC se trouve en annexe.

6.1.2 Résultat des tests

Les tests ont débuté à partir de la version 0.3 du simulateur.

Version	Notes
BETA 3	Tous les tests ont été effectués, des modifications sont nécessaires afin de corriger des bugs.
BETA 4	Corrections des problèmes de la beta 2 à l'exception de l'arrêt des flux Replay. Ajout de fonctionnalités : • Mise en mode silencieux de VLC • Affichage des informations dans la console Python
BETA 5	Migration de la librairie externe vlc.py en version 2015-04. Le problème de l'arrêt des flux Replay est corrigé mais l'implémentation de la fonction de pausing nécessite une modélisation différente *(vlc.play() est devenu une méthode bloquante)*.
1.0	Corrections des problèmes de la beta 5. Cette version est considérée comme délivrable final malgré le retrait de la fonction de pausing.

Tableau 11 : Notes de tests sur les versions

6.2 Références de validation

Les résultats obtenus par le simulateur ont été validés selon plusieurs sources considérées comme références.

6.2.1 RFC

Les machines d'états des différents protocoles ont été validées par les RFC suivantes :

Protocole	N° RFC	Référence
RTSP	2326	[VAL01]
IGMPv3	3376	[VAL02]
IGMPv2	2236	[VAL03]
RTP	3550	[VAL04]
UDP	768	[VAL05]
TCP	793	[VAL06]

Tableau 12 : Liste des RFC de référence

6.2.2 net+ Box

La cohérence des échanges réalisés entre le simulateur et le serveur multimédia de la maquette de tests a été validée à l'aide des données de références générées depuis une net+ Box à Bulle. Ces captures sont disponibles sur la Forge (https://forge.tic.eia-fr.ch/documents/3868).

6.3 Validation des résultats

La validation des résultats sur le plan réseau a été effectuée à l'aide de captures Wireshark depuis le poste simulateur.

En plus des vérifications détaillées décrites dans le plan de tests, plusieurs scénarios de grandes envergures ont été simulés.

6.3.1 Test standard - Deux flux unicast et deux flux multicast

Cette simulation globale consiste à générer deux flux unicast et deux flux multicast dans l'infrastructure de tests puis de s'y abonner avec le client en spécifiant des paramètres de base.

6.3.1.1 Paramètres

Les paramètres de la simulation, renseignés dans le fichier `simu.conf` sont les suivants :

```
[simulation]
nb_multicast:   2
nb_unicast:     2
start_type:     100
stop_type:      200
simu_duration:  20
zapping_enable: False
pause_enable:   False
channel_select: Random
```

Figure 72 : Contenu du fichier de configuration simu.conf

6.3.1.2 Console

La console affiche bien tous les éléments décrits dans le chapitre de spécifications, y compris l'heure de début et de fin de la simulation.

```
*****************************
Welcome to net+ box simulator
*****************************
Simulation start : 2015-04-30 16:06:50
Nb. unicast streams : 2
Nb. multicast streams : 2
Start type : Linear with delta 100 ms
Stop type : Linear with delta 200 ms
Channel select : Random
Simu duration : 20 sec
Simulation stop : 2015-04-30 16:07:11
*****************************
```

Figure 73 : Affichage console

6.3.1.3 Gestion des flux Unicast

On remarque les échanges standards tels que décrits dans le chapitre d'analyse pour la demande du flux Replay. Directement après l'envoi du message RTSP Play par le client, le serveur émet en Unicast le flux de données en RTP.

Dans cette capture, on remarque également qu'il y a deux flux distincts décrits en réponse du message RTSP Describe. Il s'agit de l'audio et de la vidéo, qui transitent séparément sur deux flux et sont agrégés à la destination.

540	4.067045	160.98.31.189	160.98.31.165	RTSP	191	OPTIONS rtsp://160.98.31.165:8554/test1 RTSP/1.0
541	4.088717	160.98.31.165	160.98.31.189	RTSP	190	Reply: RTSP/1.0 200 OK
542	4.088807	160.98.31.189	160.98.31.165	RTSP	217	DESCRIBE rtsp://160.98.31.165:8554/test1 RTSP/1.0
596	4.177058	160.98.31.165	160.98.31.189	RTSP/SDP	764	Reply: RTSP/1.0 200 OK
597	4.189327	160.98.31.189	160.98.31.165	RTSP	252	SETUP rtsp://160.98.31.165:8554/test1/trackID=18 RTSP/1.0 — Audio
598	4.208839	160.98.31.165	160.98.31.189	RTSP	336	Reply: RTSP/1.0 200 OK
599	4.209270	160.98.31.189	160.98.31.165	RTSP	279	SETUP rtsp://160.98.31.165:8554/test1/trackID=19 RTSP/1.0 — Vidéo
613	4.258966	160.98.31.165	160.98.31.189	RTSP	336	Reply: RTSP/1.0 200 OK
618	4.260219	160.98.31.189	160.98.31.165	RTSP	234	PLAY rtsp://160.98.31.165:8554/test1 RTSP/1.0
619	4.288619	160.98.31.165	160.98.31.189	RTP	399	PT=DynamicRTP-Type-96, SSRC=0x7BAEEC03, Seq=2104, Time=48276544, Mark
620	4.303628	160.98.31.165	160.98.31.189	RTSP	416	Reply: RTSP/1.0 200 OK
621	4.308143	160.98.31.165	160.98.31.189	RTP	75	PT=DynamicRTP-Type-96, SSRC=0xAD4BEAC8, Seq=9670, Time=197049060
622	4.308820	160.98.31.165	160.98.31.189	RTP	469	PT=DynamicRTP-Type-96, SSRC=0xAD4BEAC8, Seq=9671, Time=197049060
623	4.308844	160.98.31.165	160.98.31.189	RTP	553	PT=DynamicRTP-Type-96, SSRC=0xAD4BEAC8, Seq=9672, Time=197049060
624	4.308849	160.98.31.165	160.98.31.189	RTP	536	PT=DynamicRTP-Type-96, SSRC=0xAD4BEAC8, Seq=9673, Time=197049060
625	4.308852	160.98.31.165	160.98.31.189	RTP	599	PT=DynamicRTP-Type-96, SSRC=0xAD4BEAC8, Seq=9674, Time=197049060

Figure 74 : Etablissement du flux Unicast

A la fin de la transmission, le serveur arrête l'émission immédiatement après la réception du message RTSP Teardown.

13729	34.143818	160.98.31.165	160.98.31.189	RTP	1442	PT=DynamicRTP-Type-96, SSRC=0xAD4BEAC8, Seq=14335, Time=199734059
13730	34.143819	160.98.31.165	160.98.31.189	RTP	331	PT=DynamicRTP-Type-96, SSRC=0xAD4BEAC8, Seq=14336, Time=199734059, Mark
13731	34.147813	160.98.31.165	160.98.31.189	RTP	432	PT=DynamicRTP-Type-96, SSRC=0x7BAEEC03, Seq=2747, Time=48934976, Mark
13732	34.148285	160.98.31.189	160.98.31.165	RTSP	219	TEARDOWN rtsp://160.98.31.165:8554/test1 RTSP/1.0

Figure 75 : Arrêt de la diffusion Replay

Ce scénario se répète de manière identique pour le second flux Unicast.

6.3.1.4 Gestion des flux Multicast

Les systèmes d'exploitation modernes possèdent un mode de compatibilité IGMPv2 et IGMPv3. Si un paquet IGMP de version 2 est reçu sur l'interface réseau du client, celui-ci enverra automatiquement tous les messages en IGMPv2.

C'est ce cas de figure qui intervient dans la mesure. Cependant, le fonctionnement est strictement identique en IGMPv3, d'autres tests ayant pu le prouver.

Pour l'abonnement au flux TV Live, le client envoie sur le réseau un message de type IGMP Membership Report. Directement après, le serveur émet le contenu multimédia sur ce groupe multicast. Dans le cas où le flux est déjà diffusé sur le réseau pour un autre client, le flux est simplement commuté jusqu'au client à l'aide du processus IGMP Snooping.

A noter que la phase d'IGMP Snooping peut prendre quelques millisecondes à se mettre en place et quelques paquets *(de l'ordre d'une demi-dizaine)* peuvent être droppés ou broadcastés au niveau du switch.

1	0.000000	160.98.31.189	239.1.1.1	IGMPv2	46	Membership Report group 239.1.1.1
2	0.208868	160.98.31.175	239.1.1.1	UDP	1370	Source port: 53235 Destination port: 5004
3	0.208889	160.98.31.175	239.1.1.1	UDP	1370	Source port: 53235 Destination port: 5004
4	0.209616	160.98.31.175	239.1.1.1	UDP	1370	Source port: 53235 Destination port: 5004
5	0.209628	160.98.31.175	239.1.1.1	UDP	1370	Source port: 53235 Destination port: 5004

Figure 76 : Etablissement du flux Multicast

Du fait qu'il n'y ait pas d'échanges pour définir des ports et le type de payload avant la diffusion comme c'est le cas en Unicast, l'analyseur n'est pas capable de décoder les paquets directement en tant que RTP. Dans Wireshark il est possible de le spécifier manuellement *(Clic droit sur un paquet du flux / Decode As / RTP)*.

Pour le désabonnement par le client, l'arrêt de la diffusion par le serveur intervient quelques millisecondes après la réception du message IGMP Leave Group.

Dans le cas où d'autres clients seraient toujours abonnés au flux en question, l'IGMP Snooping désassignerait simplement le flux du port sur lequel est connecté le client.

12522	30.153848	160.98.31.189	224.0.0.2	IGMPv2	46	Leave Group 239.1.1.1
12524	30.159485	160.98.30.2	239.1.1.1	IGMPv2	60	Membership Query, specific for group 239.1.1.1
12539	30.209015	160.98.31.175	239.1.1.1	UDP	1370	Source port: 53235 Destination port: 5004
12540	30.209029	160.98.31.175	239.1.1.1	UDP	1370	Source port: 53235 Destination port: 5004
12541	30.209031	160.98.31.175	239.1.1.1	UDP	1370	Source port: 53235 Destination port: 5004
12542	30.209034	160.98.31.175	239.1.1.1	UDP	1370	Source port: 53235 Destination port: 5004
12543	30.209382	160.98.31.175	239.1.1.1	UDP	1370	Source port: 53235 Destination port: 5004
12544	30.209393	160.98.31.175	239.1.1.1	UDP	1370	Source port: 53235 Destination port: 5004

Figure 77 : Arrêt de la diffusion Live

6.3.1.5 Fichier log

Le fichier log contient toutes les informations relatives à la simulation comme défini dans le chapitre de spécifications.

```
2015-04-29 11:47:26,704 - netsimbox - INFO - Controller : net+ SimBox is started
2015-04-29 11:47:26,705 - netsimbox - INFO - Mcst1 : Init
2015-04-29 11:47:26,744 - netsimbox - INFO - Mcst1 : Join rtp://@239.1.1.1
2015-04-29 11:47:28,739 - netsimbox - INFO - Mcst2 : Init
2015-04-29 11:47:28,750 - netsimbox - INFO - Mcst2 : Join rtp://@239.2.2.2
2015-04-29 11:47:30,777 - netsimbox - INFO - Ucst1 : Init
2015-04-29 11:47:30,809 - netsimbox - INFO - Ucst1 : Play rtsp://160.98.31.165:8554/test1
2015-04-29 11:47:32,802 - netsimbox - INFO - Ucst2 : Init
2015-04-29 11:47:32,807 - netsimbox - INFO - Ucst2 : Play rtsp://160.98.31.136:8554/test2
2015-04-29 11:47:54,878 - netsimbox - INFO - Mcst2 : Stop
2015-04-29 11:47:56,895 - netsimbox - INFO - Mcst1 : Stop
2015-04-29 11:47:58,956 - netsimbox - INFO - Ucst2 : Stop
2015-04-29 11:48:00,907 - netsimbox - INFO - Ucst1 : Stop
2015-04-29 11:48:02,917 - netsimbox - INFO - Controller : net+ SimBox is stopped
```

Figure 78 : Fichier log de la simulation

L'estampille temporelle des évènements est indiquée en date et heure. Il est possible dans Wireshark d'afficher ces mêmes valeurs afin de faire correspondre le log avec la capture *(View / Time Display Format / Time Of Day)*.

6.3.1.6 Débit moyen et pertes

Le flux média de test diffusé par le serveur a une résolution de 800x600px. Sa diffusion sur le réseau occupe une bande passante d'environ 850 Kbps.

Sur la capture ci-dessous on peut constater la bande passante des différents flux *(2 Unicast en vert et 2 Multicast en rouge par rapport au total en noir)*.

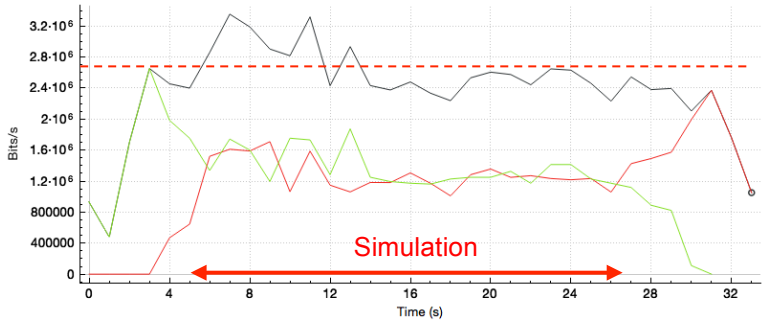


Figure 79 : Occupation de la bande passante

Quant à la qualité des flux, on peut constater qu'aucune perte de paquets n'a été mesurée par l'analyseur. Le client était donc tout à fait capable d'acquérir ces 4 flux multimédia simultanément.

Source Address	Source P	Destination Addre	Destinati	SSRC	Payload	Packet	Lost
160.98.31.165	59984	160.98.31.189	56590	0x7baeec03	RTPType-96	644	0 (0.0%)
160.98.31.165	59985	160.98.31.189	40776	0xad4beac8	RTPType-96	4667	0 (0.0%)
160.98.31.136	49122	160.98.31.189	36126	0x57b9123	RTPType-96	4052	0 (0.0%)
160.98.31.136	59512	160.98.31.189	38972	0xf52978be	RTPType-96	558	0 (0.0%)

Figure 80 : Analyse de la qualité des flux RTP

6.3.2 Test de charge – 40 flux unicast

6.3.2.1 Contexte de la simulation

La prochaine simulation résulte du fichier de configuration suivant :

```
[simulation]
nb_multicast:   0
nb_unicast:     40
start_type:     100
stop_type:      200
simu_duration:  20
zapping_enable: False
pause_enable:   False
channel_select: Fixed
```

Figure 81 : Contexte de la deuxième simulation

6.3.2.2 Gestion des flux

En analysant la capture Wireshark à l'issue de la simulation, cela permet de garantir que tous les 40 flux Unicast demandés par le client ont été correctement arrêtés après l'envoi des messages RTSP Teardown respectifs.

Il n'y a donc pas de flux orphelins à l'issue de la simulation, aspect très important du point de vue du mandant qui utilisera l'outil dans une infrastructure de production.

6.3.2.3 Débit moyen et pertes

Durant les 20 secondes de simulation avec les 40 flux Unicast en parallèle, le débit moyen mesuré sur la carte réseau est d'environ 40 Mbps. Il n'y a donc aucune congestion en ce sens pour l'interface Gigabit *(1000 Mbps)*.

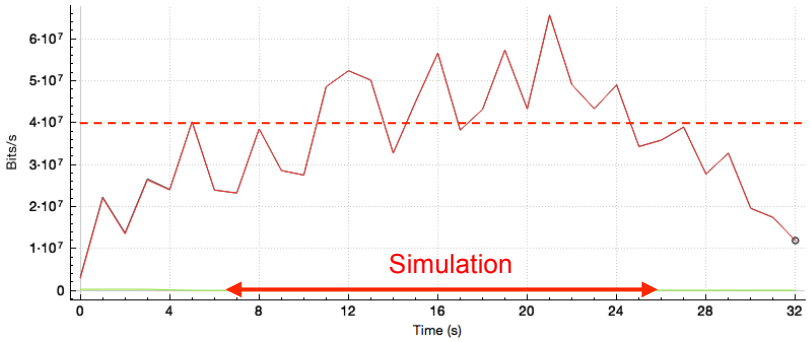


Figure 82 : Débit utilisé par la simulation

Cependant en analysant les flux RTP, on remarque un très grand pourcentage de paquets perdus, ce qui pourrait résulter à une mauvaise qualité audio/vidéo chez le client *(5% de paquets perdus en moyenne par flux)* :

Src addr	Src por	Dst addr	Dst por	SSRC	Payload	Packets	Lost
160.98.31.136	35741	160.98.31.167	38550	0xCDD43EDE	RTPType-96	8011	517 (6.1%)
160.98.31.136	44176	160.98.31.167	35188	0xA3F50817	RTPType-96	1109	38 (3.3%)
160.98.31.136	53258	160.98.31.167	41702	0xCDD43EDE	RTPType-96	8000	516 (6.1%)
160.98.31.136	51607	160.98.31.167	60584	0xA3F50817	RTPType-96	1108	37 (3.2%)
160.98.31.136	48700	160.98.31.167	55452	0xA3F50817	RTPType-96	1102	38 (3.3%)
160.98.31.136	35284	160.98.31.167	37492	0xCDD43EDE	RTPType-96	7970	504 (5.9%)
160.98.31.136	56074	160.98.31.167	59974	0xA3F50817	RTPType-96	1093	39 (3.4%)
160.98.31.136	54143	160.98.31.167	39738	0xCDD43EDE	RTPType-96	7913	507 (6.0%)
160.98.31.136	59749	160.98.31.167	43038	0xCDD43EDE	RTPType-96	7851	515 (6.2%)
160.98.31.136	43605	160.98.31.167	52306	0xA3F50817	RTPType-96	1085	38 (3.4%)
160.98.31.136	58342	160.98.31.167	36114	0xA3F50817	RTPType-96	1077	38 (3.4%)

...

Figure 83 : Pertes de paquets

Après analyse, le graphique suivant montre qu'en moyenne 7'000 paquets par secondes arrivent sur l'interface réseau. Il ne s'agit donc pas d'une surcharge à ce niveau là.

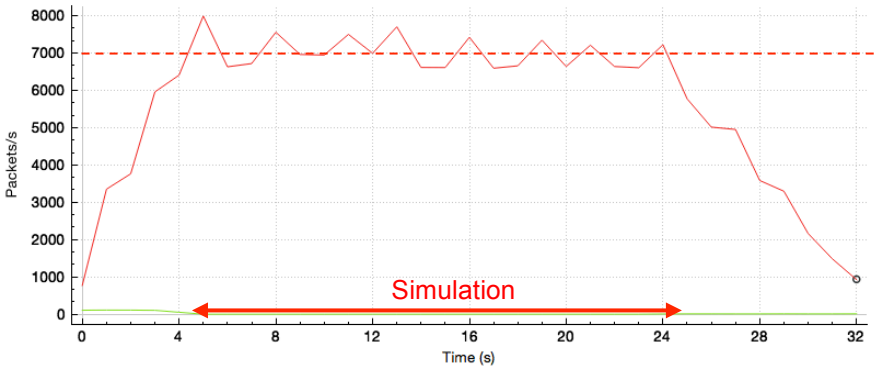


Figure 84 : Nombre de paquets reçus

Pour résoudre ce problème, trois autres points de mesures ont été mis en place sur le réseau :

a) Analyse directement sur le serveur multimédia

Comme attendu la perte est nulle pour tous les flux au niveau de l'interface réseau du serveur multimédia. Il n'y a donc aucune perte à l'émission.

b) Port mirroring sur le switch

Là encore, aucune perte n'a été mesurée sur les flux. Le switch intermédiaire, bien que ne possédant que des ports à 100Mbps, n'est pas la source des pertes.

c) Analyse directement depuis la machine physique où est hébergé le client virtuel

Les premières pertes commencent à apparaître déjà à ce niveau, de l'ordre de 0% à 1% pour environ la moitié des flux. Les autres ne subissent aucune altération de la qualité.

Sachant qu'il s'agit exactement des mêmes paquets, le taux de pertes diffère depuis la machine virtuelle et la machine physique. Ceci est probablement dû à la virtualisation qui induit un temps de latence supplémentaire. Certains paquets n'arrivent donc pas dans l'ordre et sont considérés comme droppés par Wireshark *(l'analyse du flux RTP est faite au niveau applicatif)*.

Ce problème n'est pas majeur pour le projet, les flux de données étant ignorés. Cependant pour un projet ultérieur nécessitant le traitement de ces paquets *(mesure de la QoS par exemple)*, il serait conseillé de prendre les mesures suivantes :

- Exécution du client sur une machine physique directement
- Utilisation de TCPDump ou LibPcap pour capturer les paquets au plus bas niveau

6.3.2.4 Charge CPU

Un processus Unicast du simulateur, consomme environ 1.5% d'un processeur professionnel actuel *(type i7-4770 à 3.4GHz)*. Des pics de consommation se font sentir au lancement et à l'arrêt des flux, nécessitant une gestion plus poussée des processus.

Sur une machine virtuelle possédant les caractéristiques suivantes :

- 4 cœurs i7-4770 à 3.4 GHz chacun
- 8 Go de mémoire RAM

On peut constater le graphique suivant, montrant la consommation des ressources :

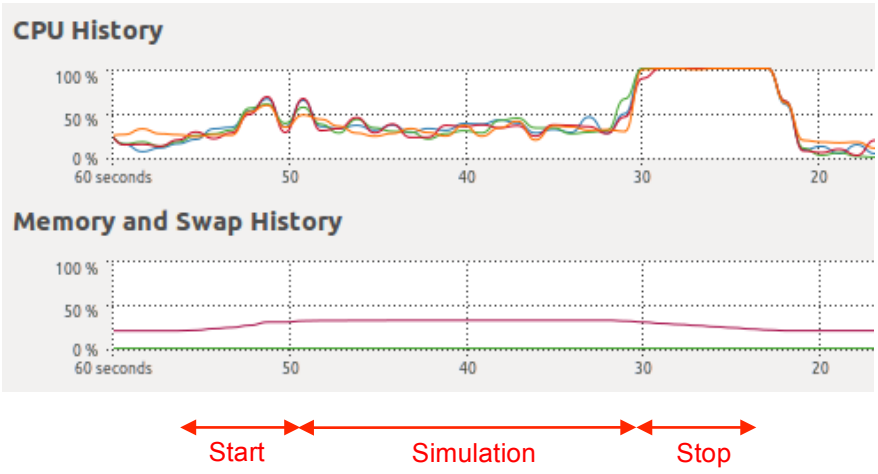


Figure 85 : Consommation en ressources pour 40 flux Unicast

Le pic d'activité pour le processeur se situe à l'arrêt des processus, sans toutefois bloquer la machine dans cette phase.

En revanche la consommation d'un processus Multicast est négligeable en dehors des phases de lancement et d'arrêt d'une simulation. Ceci car il n'y a pas de machine d'états TCP à maintenir et gérer comme pour l'unicast. Un processus Multicast reste donc totalement passif.

6.3.3 Test de charge – 80 flux unicast

Pour ce troisième test, 80 flux ont été demandés depuis le client, avec la machine virtuelle aux caractéristiques suivantes :

- 8 cœurs virtuels à 3.4 GHz
- 8 Go de RAM

On peut une nouvelle fois constater le graphique suivant :

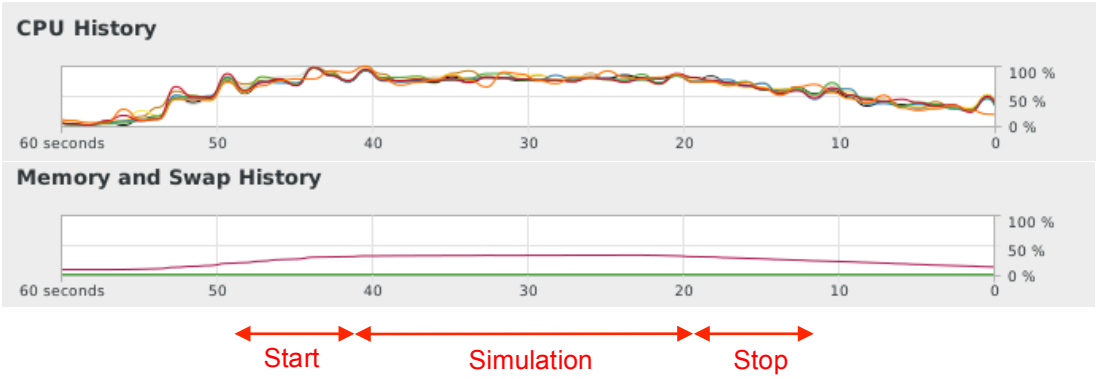


Figure 86 : Consommation des ressources pour 80 flux Unicast

On ne distingue plus le pic de consommation à la fin de la simulation, de par une meilleure distribution des tâches sur les 8 cœurs du processeur.

Du point de vue mémoire on ne note pas de besoin excessif de la part du simulateur. Sa consommation est d'environ 2.5 Go pour 80 flux, ce qui représente environ 30 Mo par processus.

6.3.4 Recommandations logicielles et matérielles

Par ces différents scénarios, la configuration minimale suivante est fortement recommandée pour exécuter le simulateur :

Elément	Description
Système d'exploitation	Debian 8
Mémoire RAM	8 Go
Processeur	Intel i5, i7 ou Xeon
Nombre de cœurs	4x 2.7 GHz minimum
Carte réseau	1000 Mbps

Tableau 13 : Recommandations matérielles

Une telle configuration permet de gérer aisément 30 flux unicast parallèles. Pour aller au-delà de ce nombre, des ressources supplémentaires seront nécessaires au niveau du processeur. Pour des flux multicast par contre, la quasi seule limite est la capacité de l'interface réseau, la charge processeur étant négligeable.

Au cas où une capture Wireshark est lancée en parallèle, il faut également s'assurer que le flux de données peut être capturé correctement pour ne pas introduire d'erreurs dues à un trop gros nombre de paquets comme décrit précédemment.

6.4 Conclusion

Au travers de cette phase, le bon fonctionnement du simulateur a pu être validé au sein de l'infrastructure de test de la HEIA-FR. Il ne reste maintenant plus qu'à intégrer le logiciel au sein du réseau de net+ FR, sur lesquels sont connectés les abonnés.

7 Intégration

La phase d'intégration consiste à déployer le simulateur sur le réseau de production de net+ FR. Il s'agit de réaliser la même batterie de tests fonctionnels qu'opérés dans le laboratoire télécom de la HEIA-FR. Cela confirmera le bon fonctionnement du simulateur dans l'environnement du mandant.

Dans un second temps, une analyse des débits de l'interface WAN du CPE de net+ FR ainsi que de la charge processeur de l'équipement sera réalisée.

7.1 Tests d'intégration

7.1.1 Test standard - Deux flux unicast et deux flux multicast

Tout comme préalablement réalisé dans l'infrastructure de tests, deux flux unicast et deux flux multicast seront demandés aux serveurs multimédia de net+ CH.

7.1.1.1 Gestion des flux unicast

Les flux unicast sont bien démarrés par les serveurs de manière similaire à une net+ Box :

14	09:22:20.275352	192.168.1.223	178.237.87.67	RTSP	231	OPTIONS rtsp://viamanager.netplus.ch:554/pvrreadonly/test_content_vds15.mpg RTSP/1.0
16	09:22:20.279713	178.237.87.67	192.168.1.223	RTSP	286	Reply: RTSP/1.0 200 OK
18	09:22:20.279795	192.168.1.223	178.237.87.67	RTSP	257	DESCRIBE rtsp://viamanager.netplus.ch:554/pvrreadonly/test_content_vds15.mpg RTSP/1.0
19	09:22:20.295163	178.237.87.67	192.168.1.223	RTSP/SDP	499	Reply: RTSP/1.0 200 OK
21	09:22:20.301351	192.168.1.223	178.237.87.67	RTSP	286	SETUP rtsp://viamanager.netplus.ch:554/pvrreadonly/test_content_vds15.mpg/ RTSP/1.0
34	09:22:20.411484	178.237.87.67	192.168.1.223	RTSP	405	Reply: RTSP/1.0 200 OK
35	09:22:20.411740	192.168.1.223	178.237.87.67	RTSP	287	PLAY rtsp://viamanager.netplus.ch:554/pvrreadonly/test_content_vds15.mpg/ RTSP/1.0
42	09:22:20.441046	178.237.87.67	192.168.1.223	MPEG TS	1358	Source port: 1234 Destination port: 47724
43	09:22:20.441389	178.237.87.67	192.168.1.223	MPEG TS	1358	Source port: 1234 Destination port: 47724
44	09:22:20.441409	178.237.87.67	192.168.1.223	MPEG TS	1358	Source port: 1234 Destination port: 47724
45	09:22:20.441414	178.237.87.67	192.168.1.223	MPEG TS	1358	Source port: 1234 Destination port: 47724
46	09:22:20.441419	178.237.87.67	192.168.1.223	MPEG TS	1358	Source port: 1234 Destination port: 47724
47	09:22:20.441517	178.237.87.67	192.168.1.223	MPEG TS	1358	Source port: 1234 Destination port: 47724

Figure 87 : Démarrage d'un flux unicast

Pour l'arrêt également, les données cessent d'être émises par les serveurs immédiatement après l'envoi du message RTSP Teardown :

23648	09:22:40.763518	178.237.87.67	192.168.1.223	MPEG TS	1358	Source port: 1234 Destination port: 47724
23649	09:22:40.765309	178.237.87.67	192.168.1.223	MPEG TS	1358	Source port: 1234 Destination port: 47724
23650	09:22:40.767044	178.237.87.67	192.168.1.223	MPEG TS	1358	Source port: 1234 Destination port: 47724
23651	09:22:40.768597	178.237.87.67	192.168.1.223	MPEG TS	1358	Source port: 1234 Destination port: 47724
23652	09:22:40.770368	178.237.87.67	192.168.1.223	MPEG TS	1358	Source port: 1234 Destination port: 47724
23653	09:22:40.770636	192.168.1.223	178.237.87.67	RTSP	272	TEARDOWN rtsp://viamanager.netplus.ch:554/pvrreadonly/test_content_vds15.mpg/ RTSP/1.0

Figure 88 : Arrêt d'un flux unicast

7.1.1.2 Gestion des flux multicast

Les flux multicast sont également démarrés de manière similaire à un net+ Box, suite à l'émission d'un message IGMPv3 Membership Report demandant à rejoindre le groupe en question :

14	09:13:30.477115	192.168.1.223	224.0.0.22	IGMPv3	54	Membership Report / Join group 239.101.10.41 for any sources
15	09:13:30.495525	10.0.218.9	239.101.10…	MPEG TS	1358	Source port: 49152 Destination port: 1234
16	09:13:30.497539	10.0.218.9	239.101.10…	MPEG TS	1358	Source port: 49152 Destination port: 1234
17	09:13:30.499510	10.0.218.9	239.101.10…	MPEG TS	1358	Source port: 49152 Destination port: 1234
18	09:13:30.501424	10.0.218.9	239.101.10…	MPEG TS	1358	Source port: 49152 Destination port: 1234
19	09:13:30.503454	10.0.218.9	239.101.10…	MPEG TS	1358	Source port: 49152 Destination port: 1234
20	09:13:30.505293	10.0.218.9	239.101.10…	MPEG TS	1358	Source port: 49152 Destination port: 1234
21	09:13:30.507160	10.0.218.9	239.101.10…	MPEG TS	1358	Source port: 49152 Destination port: 1234

Figure 89 : Démarrage d'un flux multicast

Pour l'arrêt, un message IGMPv3 Membership Report demandant à quitter le groupe est émis puis coupe le flux venant des serveurs. Si d'autres membres sont encore abonnés, l'IGMP Snooping du switch désassigne simplement le port du client au groupe multicast.

```
23787 09:13:51.006989  10.0.218.9     239.101.10…  MPEG TS   1358 Source port: 49152  Destination port: 1234
23788 09:13:51.008800  10.0.218.9     239.101.10…  MPEG TS   1358 Source port: 49152  Destination port: 1234
23789 09:13:51.010662  10.0.218.9     239.101.10…  MPEG TS   1358 Source port: 49152  Destination port: 1234
23790 09:13:51.012455  10.0.218.9     239.101.10…  MPEG TS   1358 Source port: 49152  Destination port: 1234
23791 09:13:51.014541  10.0.218.9     239.101.10…  MPEG TS   1358 Source port: 49152  Destination port: 1234
23793 09:13:51.359383  192.168.1.223  224.0.0.22   IGMPv3      54 Membership Report / Leave group 239.101.10.41
```

Figure 90 : Arrêt d'un flux multicast

7.1.1.3 Mesure des débits

Sur le graphique ci-dessous, on distingue clairement les deux flux multicast *(en rouge)* ainsi que les deux flux unicast *(en vert)* par rapport au total *(en noir)*.

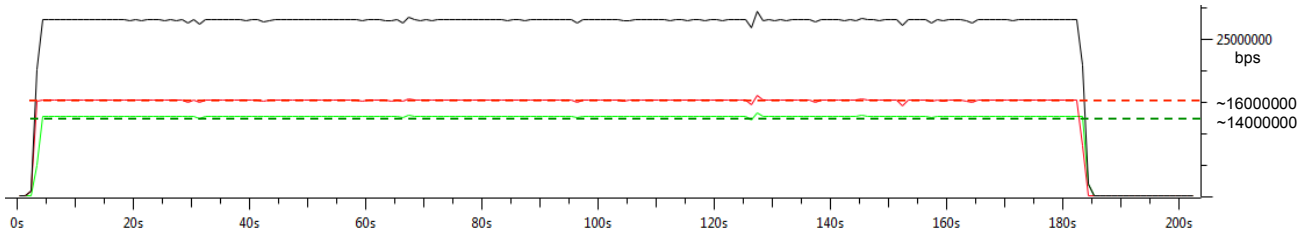


Figure 91 : Bande passante de la simulation 2x2 flux

D'après cette simulation, on peut constater a bande passante utilisée pour chacun des deux types de flux dans le réseau de net+ CH :

- Unicast : ~ 6 Mbps
- Multicast : ~ 7 Mbps *(chaîne HD)*

7.1.2 Test de charge – 15 flux unicast

Ce premier test de charge consiste à analyser le comportement du CPE face à plusieurs flux parallèles. Pour ce faire une simulation demandant simultanément 15 flux unicast a été lancée.

Afin de ne pas surcharger les serveurs multimédia de net+ CH, chaque flux est requêté sur un serveur différent parmi les 20 disponibles.

7.1.2.1 Gestion des flux

Comme il a déjà été démontré lors de la mesure précédente, tous les 15 flux lancés ont correctement été arrêtés à la fin de la simulation à l'aide de messages RTSP Teardown.

7.1.2.2 Mesure de débit

De par la taille importante des flux, le décodage des données MPEG par Wireshark peut prendre *beaucoup* de temps. Ainsi pour éviter de trop attendre, il est conseillé de désactiver le décodage de la couche applicative pour les mesures où il n'y en a aucune nécessité.

Désactivation du décodage MPEG TS dans Wireshark

Dans le menu **Analyze**, cliquer sur **Enabled Procotols**. Dans la liste, sélectionner **MP2T** avant de cliquer sur le bouton **Appliquer**.

Pour les 15 flux en parallèle, la bande passante mesurée est de ~ 85 Mbps.

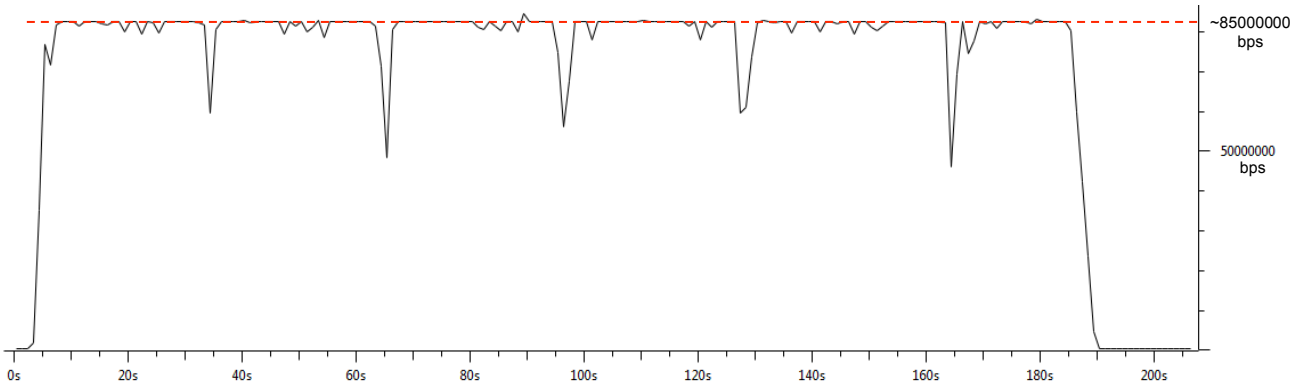


Figure 92 : Bande passante de la simulation 15 flux

En rapportant la bande passante totale à une seule unité, on retrouve l'approximation donnée précédemment pour un flux unicast.

$$\frac{\sim 85\ Mbps}{15\ flux} = \sim 6\ Mbps/flux\ unicast$$

7.1.2.3 Mesures depuis le CPE

Un accès SSH sur le CPE des clients permet à net+ FR d'effectuer différentes mesures à distance. Ainsi, la transmission des 15 flux simultanés a pu être observée et deux informations clés ont été relevées.

La première est la bande passante occupée sur le port WAN *(optique)* du CPE. On retrouve bien les valeurs mesurées, soit ~ 85 Mbps en moyenne :

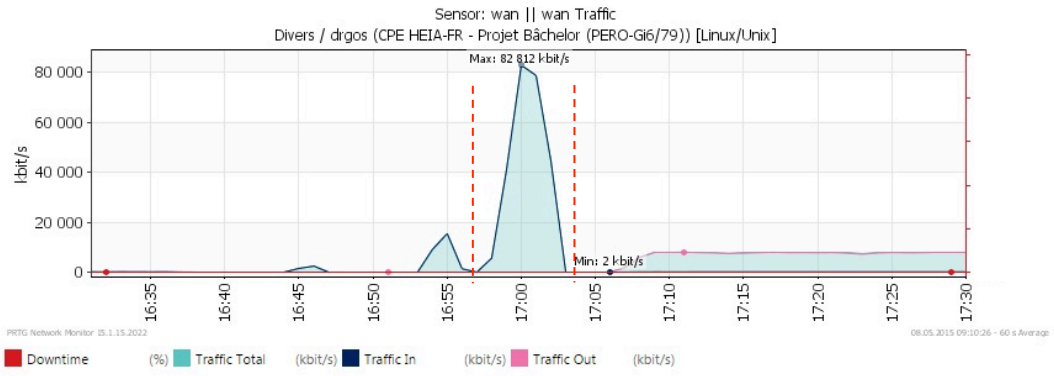


Figure 93 : Bande passante utilisée en WAN lors de la simulation

La seconde information utile est la charge du CPE durant ce même laps de temps, d'environ 20% :

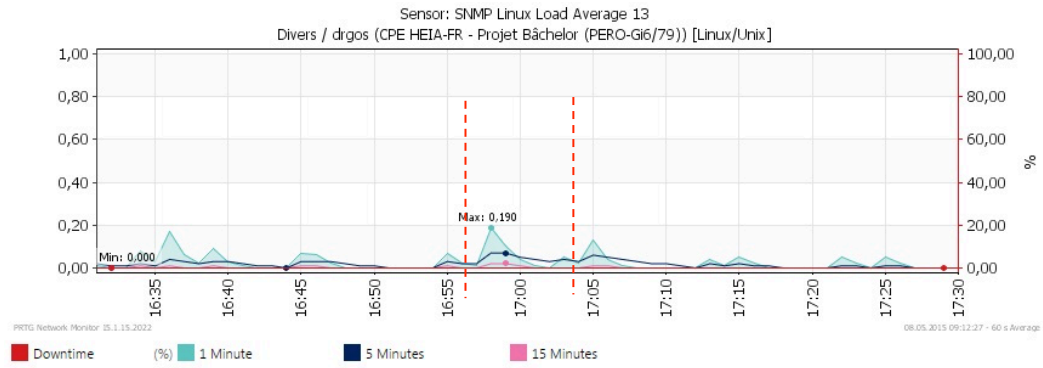


Figure 94 : Charge du CPE lors de la simulation

D'après ces informations, on peut constater que le port WAN optique gigabit est à environ 10% de sa capacité. Cependant, la charge du CPE est reste tout à fait correcte *(environ 20%)*.

Une mesure avec un plus grand nombre de flux est donc nécessaire pour déterminer le comportement du CPE face à une charge plus importante.

7.1.3 Test de charge – multiples flux parallèles

Ce test a été subdivisé en 7 phases distinctes, afin de pouvoir analyser les différences de comportement en fonction du type de flux et de la charge du plan de contrôle *(RTSP ou IGMP)*.

- Simulation 1 – 20 flux unicast
- Simulation 2 – 40 flux unicast
- Simulation 3 – 40 flux multicast sans zapping
- Simulation 4 – 40 flux multicast avec zapping toutes les 0 à 5 secondes
- Simulation 5 – 40 flux multicast avec zapping toutes les 30 à 60 secondes
- Simulation 6 – 40 flux multicast sans zapping + 20 flux unicast
- Simulation 7 – 68 flux multicast sans zapping

Pour ce faire, le simulateur a été lancé 7 fois avec des paramètres différents selon les cas. Ceci avec un intervalle de temps entre les simulations afin de bien les distinguer sur les graphiques.

7.1.3.1 Gestion des flux

Durant ces importantes charges, le simulateur s'est comporté de la manière attendue. Tous les flux ont été correctement démarrés et arrêtés à l'issue de la simulation.

Les captures Wireshark réalisées entre les simulations depuis le poste client prouvent qu'aucun flux orphelin n'est présent sur le réseau.

7.1.3.2 Mesures depuis le CPE

De la même manière que pour le test précédent, des graphiques ont été générés à l'aide d'un outil de monitoring afin de connaître le taux d'utilisation du port WAN ainsi que du processeur sur le CPE.

Pour commencer, on peut constater que le débit utilisé sur le port WAN est conforme aux prédictions, soit la somme des débits de tous les flux cumulés.

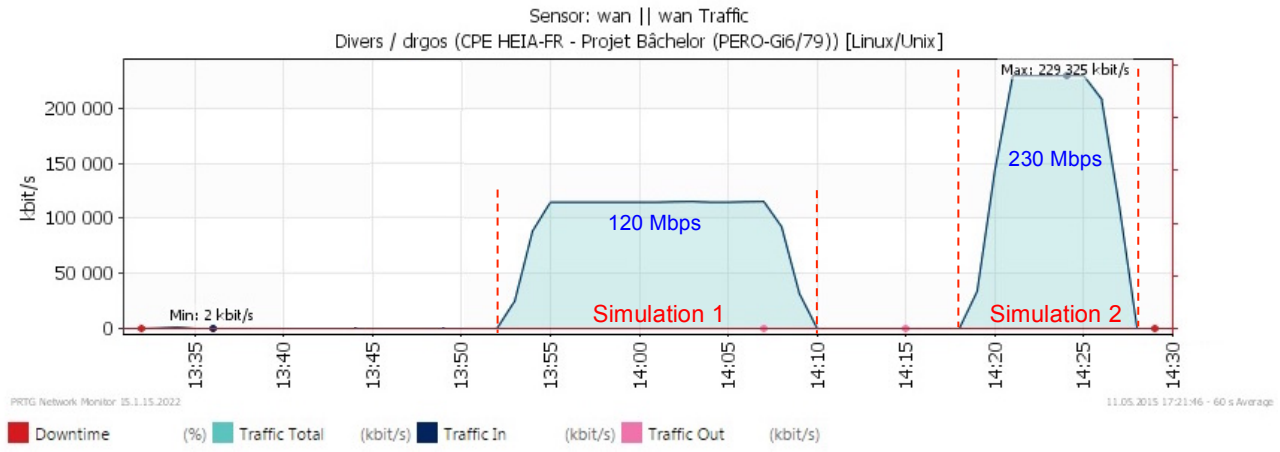


Figure 95 : Utilisation port WAN en simulations multiples 1/3

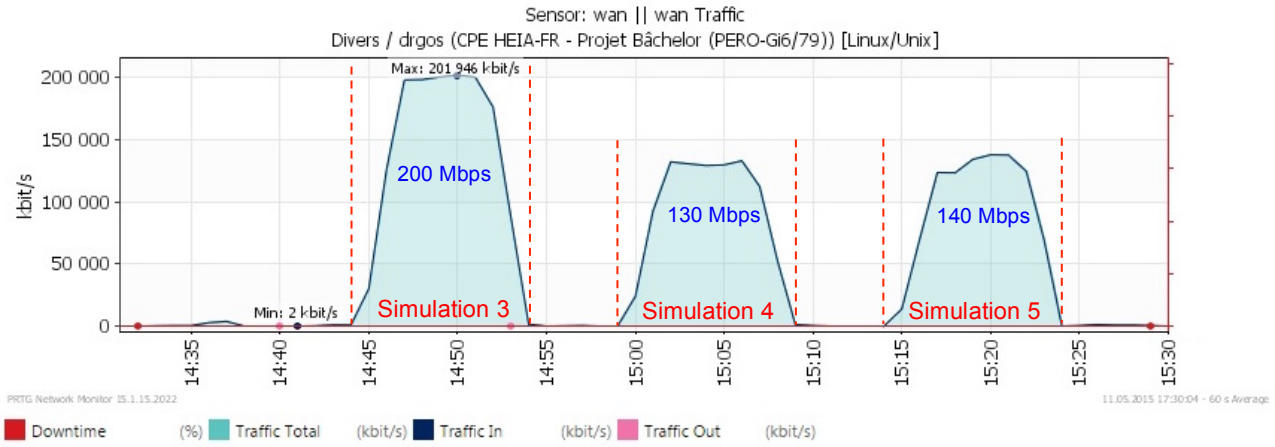


Figure 96 : Utilisation port WAN en simulations multiples 2/3

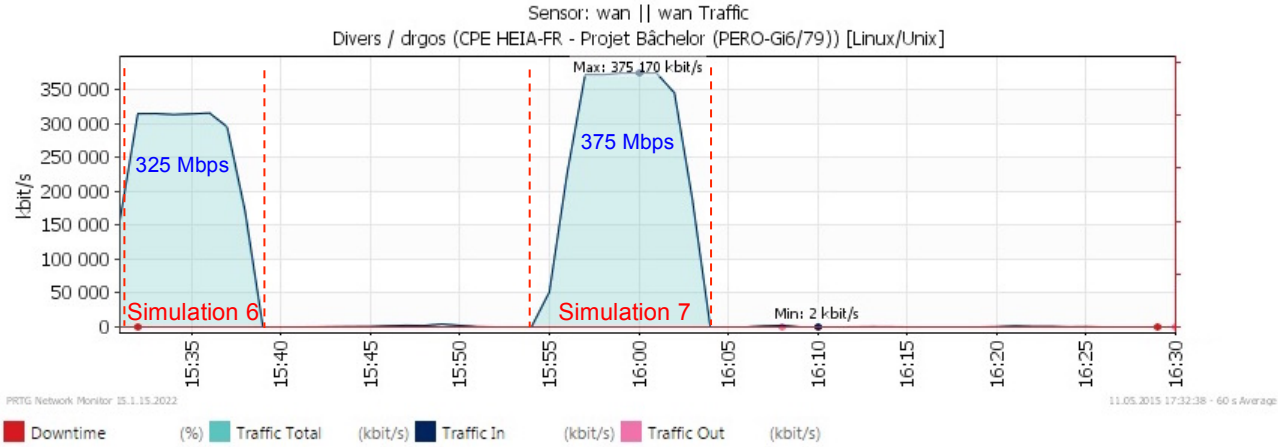


Figure 97 : Utilisation port WAN en simulations multiples 3/3

Les graphiques suivants montrent la charge du processeur utilisée par le CPE aux instants des différentes simulations.

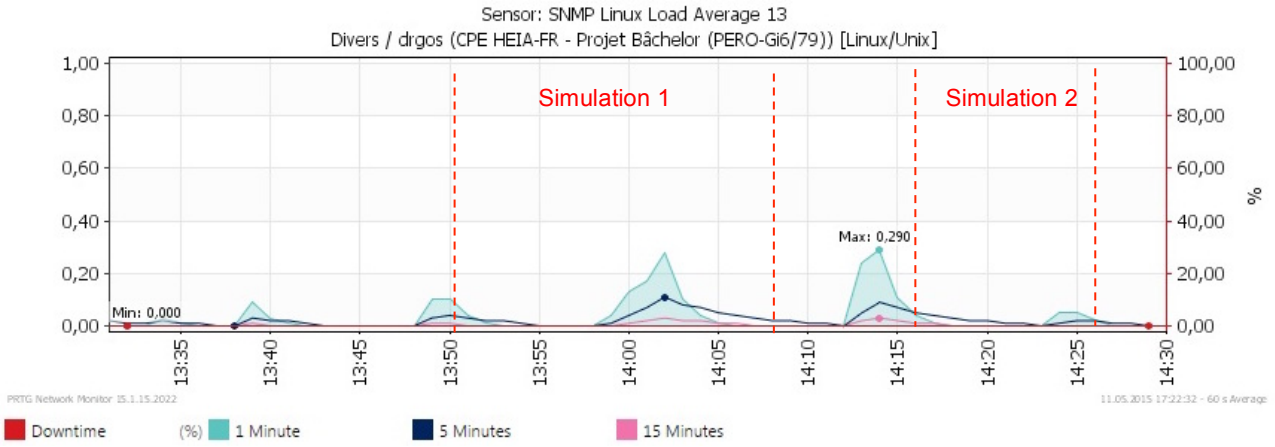


Figure 98 : Utilisation du CPE en simulations multiples 1/3

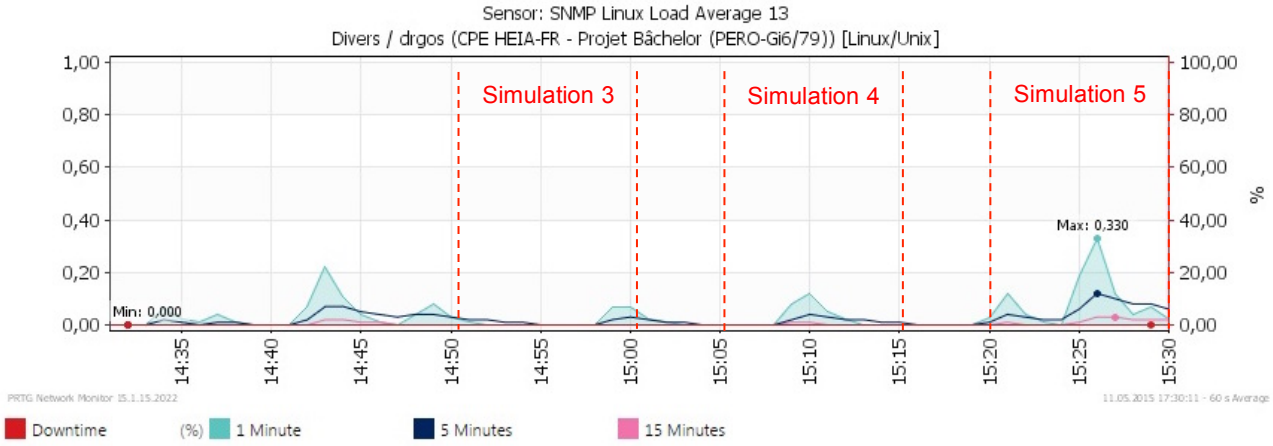


Figure 99 : Utilisation du CPE en simulations multiples 2/3

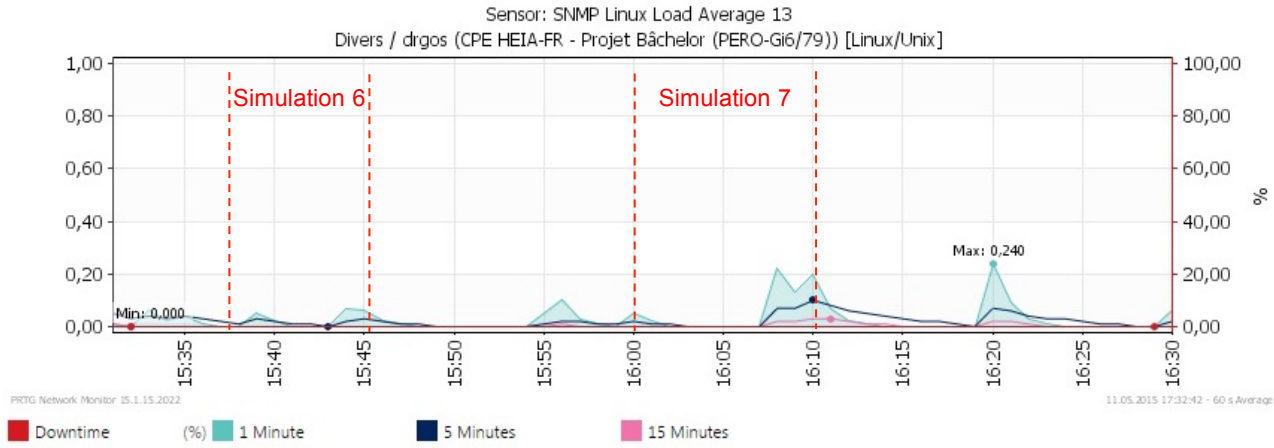


Figure 100 : Utilisation du CPE en simulations multiples 3/3

Au travers de ces différents graphiques, on peut rapidement remarquer que la consommation de CPU n'est pas liée à l'utilisation du port WAN.

Une analogie pourrait être observée sur les simulations 4 et 5. La fonction de zapping étant activée dans ces simulations, elle génère un nombre important de paquets IGMP non cadencés comme les flux de données RTP. C'est d'ailleurs cette même augmentation de la charge qui est observée en début et fin des simulations 3, 6 et 7 faisant intervenir des flux multicast sans zapping.

7.1.3.3 Synthèse

Ces différentes simulations permettent tout d'abord de prouver que le CPE est capable de supporter la charge de 60 Set-Top Box simples *(1 flux par box)*, le port n'ayant pas dépassé 37% de sa capacité de le CPU 33%.

Un second test de charge sera réalisé à l'issue du projet. Celui-ci sera distribué sur plusieurs simulateurs, afin d'atteindre les limites du CPE qui devraient se trouver aux alentours de 150 flux parallèles.

7.2 Impact du simulateur sur le réseau

Sur le réseau de net+ FR, le simulateur peut avoir des conséquences non négligeables. Il est source d'un trafic de données important qui pourrait impacter le réseau de plusieurs manières.

7.2.1 Charge de trafic dans le réseau Métro-Accès

La charge du trafic généré par le simulateur utilisé en connaissance de cause n'est pas plus importante que celle générée par *n* Set-Top Box. Cependant, il n'y a aucune limite définie dans le nombre maximal de flux pouvant être lancé lors d'une simulation. Ainsi il serait par exemple possible de demander 1'000 flux TV Replay aux serveurs.

Bien sûr la carte réseau du client n'aurait jamais la capacité de les absorber, mais c'est également tous les équipements actifs entre les serveurs multimédia et l'abonné qui risqueraient d'être surchargés !

7.2.2 Flooding des serveurs multimédia

Le fait de surcharger des liens du réseau avec un nombre important de demandes tel que décrit précédemment aurait un impact sur le réseau, mais ce que localement pour tous les clients empruntant le même brin que l'abonné en question pour se rendre vers les serveurs multimédia.

Le réseau Core qui transporte les flux multimédia est dimensionné pour soutenir une telle charge, il faut en effet pouvoir assurer tous les abonnés TV puissent regarder un flux Replay simultanément.

Un autre impact que pourrait avoir le simulateur sur les serveurs multimédia serait un risque de flooding par des requêtes jusqu'à les surcharger. Admettons qu'un client demande 1'000'000 de flux unicast immédiatement, à l'aide du logiciel. Que se passe-t-il ? Pour y répondre, il faudrait étudier l'implémentation des serveurs multimédia Via actuellement en production et consulter les tests de charge déjà réalisés. Mais probablement que le serveur de Front-End aurait de la peine à traiter toutes les requêtes simultanément.

7.2.3 Flux orphelins

Le dernier risque identifié concernant le simulateur serait un plantage au cours d'une simulation. Pour cela, deux cas de figure sont à considérer.

7.2.3.1 Flux unicast

Ceux-ci sont liés à une instance de Python et gérés par le système d'exploitation. Ainsi, si le processus venait à se planter, il suffirait de le terminer de force pour que la session TCP/RTSP établie soit automatiquement fermée par l'envoi d'un message TCP RST *(Reset)* de la part du client. Suite à cet événement, le serveur arrête immédiatement le flux RTP relatif à la session RTSP terminée.

```
25538 09:27:49.278617000 192.168.1.223  178.237.87.67   TCP          54 58691→554 [RST] Seq=1209 Win=0 Len=0
25547 09:27:49.279964000 192.168.1.223  178.237.87.67   TCP          54 58691→554 [RST] Seq=1210 Win=0 Len=0
25549 09:27:49.280091000 192.168.1.223  178.237.87.67   TCP          54 58691→554 [RST] Seq=1210 Win=0 Len=0
25552 09:27:49.281962000 192.168.1.223  178.237.87.67   TCP          54 58691→554 [RST] Seq=1210 Win=0 Len=0
25560 09:27:49.282742000 192.168.1.223  178.237.87.67   TCP          54 58691→554 [RST] Seq=1210 Win=0 Len=0
```

Figure 101 : Arrêt de la session TCP

Une autre possibilité serait que le client ne puisse pas ou n'ait pas le temps d'arrêter les flux, lors d'une coupure d'électricité par exemple. Ce cas est géré par le flux RTP directement au travers du protocole RTCP. Ainsi en communication standard, des échanges réguliers de type Receiver et Sender Report ont lieu. Ceux-ci décrivent la qualité de la transmission et pourraient être utiles dans une étude de la QoS.

A la fin de la communication, un message de type Goodbye est envoyé pour signaler la fin du flux.

6549	82.188201000	160.98.31.160	160.98.31.166	RTCP	106	Sender Report	Source description	
6556	82.193311000	160.98.31.160	160.98.31.166	RTCP	106	Sender Report	Source description	
6631	82.627603000	160.98.31.166	160.98.31.160	RTCP	98	Receiver Report	Source description	
7310	86.037819000	160.98.31.166	160.98.31.160	RTCP	98	Receiver Report	Source description	
7525	87.183460000	160.98.31.160	160.98.31.166	RTCP	106	Sender Report	Source description	
7532	87.213513000	160.98.31.160	160.98.31.166	RTCP	106	Sender Report	Source description	
7768	88.577629000	160.98.31.166	160.98.31.160	RTCP	98	Receiver Report	Source description	
8434	91.666686000	160.98.31.166	160.98.31.160	RTCP	98	Receiver Report	Source description	
8483	91.927482000	160.98.31.166	160.98.31.160	RTCP	98	Receiver Report	Source description	
8540	92.188253000	160.98.31.160	160.98.31.166	RTCP	106	Sender Report	Source description	
8553	92.228952000	160.98.31.160	160.98.31.166	RTCP	106	Sender Report	Source description	
9071	94.957669000	160.98.31.166	160.98.31.160	RTCP	98	Receiver Report	Source description	
9271	96.107703000	160.98.31.166	160.98.31.160	RTCP	98	Receiver Report	Source description	
9439	97.003744000	160.98.31.160	160.98.31.166	RTCP	114	Sender Report	Source description	Goodbye
9440	97.003745000	160.98.31.160	160.98.31.166	RTCP	114	Sender Report	Source description	Goodbye
9442	97.037312000	160.98.31.166	160.98.31.160	RTCP	82	Receiver Report	Goodbye	
9443	97.037312000	160.98.31.166	160.98.31.160	RTCP	82	Receiver Report	Goodbye	

Figure 102 : Echanges RTSP normaux

En cas de disparition brutale du client, aucun message ne sera reçu en retour par la source. Après un certain temps, elle pourra donc en déduire par elle-même que le client a disparu et par conséquence arrêter la diffusion du flux.

891	11.761980000	160.98.31.166	160.98.31.160	RTCP	98	Receiver Report	Source description
973	12.222084000	160.98.31.166	160.98.31.160	RTCP	98	Receiver Report	Source description
1305	13.955374000	160.98.31.160	160.98.31.166	RTCP	106	Sender Report	Source description
1307	13.975098000	160.98.31.160	160.98.31.166	RTCP	106	Sender Report	Source description
2317	18.972208000	160.98.31.160	160.98.31.166	RTCP	106	Sender Report	Source description
2325	18.972970000	160.98.31.160	160.98.31.166	RTCP	106	Sender Report	Source description
3373	23.942679000	160.98.31.160	160.98.31.166	RTCP	106	Sender Report	Source description
3375	23.972393000	160.98.31.160	160.98.31.166	RTCP	106	Sender Report	Source description
4322	28.954522000	160.98.31.160	160.98.31.166	RTCP	106	Sender Report	Source description
4324	28.974286000	160.98.31.160	160.98.31.166	RTCP	106	Sender Report	Source description
5283	33.969594000	160.98.31.160	160.98.31.166	RTCP	106	Sender Report	Source description
5285	33.974695000	160.98.31.160	160.98.31.166	RTCP	106	Sender Report	Source description
6230	38.941259000	160.98.31.160	160.98.31.166	RTCP	106	Sender Report	Source description
6232	38.971581000	160.98.31.160	160.98.31.166	RTCP	106	Sender Report	Source description
7148	43.953786000	160.98.31.160	160.98.31.166	RTCP	106	Sender Report	Source description
7151	43.973820000	160.98.31.160	160.98.31.166	RTCP	106	Sender Report	Source description

Figure 103 : Messages RTCP sans réponse

Ainsi un plantage du simulateur avec des flux unicast n'aurait pas d'impact majeur sur le réseau de net+. Cela générerait des flux orphelins durant un certain laps de temps, mais l'émetteur est par lui-même capable de les couper après une certaine période. C'est d'ailleurs le même procédé qui entrerait en action si une coupure d'électricité survenait dans une région alors que des clients sont en train de visionner du contenu de TV Replay.

7.2.3.2 Flux multicast

Les flux multicast étant gérés par IGMP, lui-même basé sur UDP au plan de contrôle, il n'y a pas de notions de sessions comme en TCP. Du fait, les serveurs multimédia n'ont aucun moyen de savoir si le client est toujours présent ou non sur le réseau. C'est pour cela que le switch derrière lequel se trouve le client potentiellement abonné à un flux multicast envoie périodiquement des messages IGMP Membership Query. La réponse du client permet de confirmer sa présence à la manière d'un Keep-Alive.

Ainsi si le client ne se désabonnait pas explicitement à un flux multicast, celui-ci serait automatiquement arrêté au bout d'un certain temps *(selon les paramètres définis dans les équipements)*, aucune réponse n'étant émise suite aux messages IGMP Membership Query.

7.3 Conclusion

Cette dernière étape du projet a permis de valider le bon fonctionnement du simulateur au sein du réseau de net+ FR. Le logiciel peut ainsi être remis au mandant dans les temps définis selon la planification initiale.

Cela a également permis de prouver que tous les points clés du cahier des charges ont été respectés à l'implémentation et que le comportement du simulateur est correct face aux différents cas d'utilisation dans son environnement de production.

8 Conclusion

8.1 Débriefing

Le premier objectif consistait à énumérer, analyser et documenter les différentes solutions software et hardware de génération de paquets ainsi que les architectures de diffusion de contenus multimédia de type IPTV. Cette analyse complète a permis de proposer des solutions adéquates pour la spécification du simulateur et sa mise en œuvre avec les différents scénarios de tests.

Pour atteindre le second objectif, une maquette de tests a été mise en place dans le laboratoire télécom de la HEIA-FR. Celle-ci a permis de tester le logiciel final dans un environnement sécurisé, en dehors du réseau productif de net+ FR. Elle a en outre réduit au maximum les problèmes d'intégration dans l'environnement du mandant, en utilisant les mêmes protocoles.

L'objectif final qui consistait à développer le simulateur de Set-Top Box a été réalisé avec succès sur la base des différents éléments analysés en début de projet. Tous les points définis au cours de la phase de spécifications ont pu être implémentés avec succès à l'exception d'une petite fonctionnalité secondaire.

Les trois objectifs définis dans le cahier des charges ont été atteints avec succès à la fin du projet selon les délais imposés.

8.2 Difficultés rencontrées

Durant la phase de spécifications, la première difficulté rencontrée consistait à trouver une modélisation simple mais très évolutive permettant l'extension aisée du projet. N'étant que la base de futurs projets, il était très important de pouvoir rajouter facilement de nouvelles fonctionnalités au logiciel sans avoir à retravailler le code de base. C'est ainsi que la modélisation par processus a été proposée puis mise en place.

La seconde difficulté est apparue au cours de la mise en place de l'infrastructure IPTV de tests. Il est relativement complexe sur la base de captures Wireshark de recréer un environnement de simulation au plus proche de celui implémenté chez le mandant. Le réseau de la HEIA-FR comporte beaucoup d'éléments actifs de vendeurs différents, générant du trafic avec des protocoles parfois de versions non désirées. La solution a été trouvée en mettant en place un réseau isolé pour les tests.

Finalement, le jeune âge de la librairie VLC pour Python a posé quelques difficultés. Sa grande communauté de développeurs propose très régulièrement des mises à jour pour amener certaines corrections et l'implémentation des fonctions principales *peut* varier d'une version à une autre. C'est d'ailleurs un changement de version qui a empêché de mettre en place la fonctionnalité de *pausing* telle que décrite dans les spécifications.

8.3 Perspectives futures

Le développement du simulateur de Set-Top Box a permis de relever un bon nombre de perspectives en vue de son amélioration par l'ajout de nouvelles fonctionnalités ou l'extension des concepts déjà implémentés.

Comme décrit dans le chapitre de Conception, les principaux points d'extension sont :

- Création de scénarios de simulation
- Modélisation de profils de trafics pour chaque Set-Top Box
- Analyse de la qualité de service des flux reçus
- Ajout d'autres types de trafic
- Capture automatique du trafic au lancement d'une simulation

8.4 Conclusion personnelle

Durant ce travail de semestre la partie que j'ai préférée est l'Analyse. Elle m'a permis d'étudier les architectures et technologies utilisées dans les réseaux IPTV des grands opérateurs de télécommunications, thème sur lequel je n'ai jamais travaillé au cours de mes études. Elle m'a également permis de consolider mes connaissances dans le domaine de la distribution de flux en Multicast.

L'analyse des architectures IPTV m'a ensuite aidé pour la conception des scénarios de tests en vue de valider le bon fonctionnement du simulateur.

Ensuite par la conception du logiciel, j'ai pu acquérir de l'expérience avec le langage de programmation Python tout en mettant en pratique les différents aspects théoriques fraîchement appris.

Bien plus que sur le plan académique, j'ai pu acquérir au travers de ce projet des compétences professionnelles telles que la gestion de projet, la tenue de séances, la rédaction de procès-verbaux, la planification des tâches et bien plus encore. Je suis persuadé qu'il s'agit là d'une bonne expérience que je pourrai appliquer dans le monde professionnel.

Finalement, le travail en collaboration avec l'entreprise net+ FR fut très enrichissant. Cela m'a permis de bien comprendre les problématiques et contraintes de la réalisation d'un tel projet pour un réseau de production avec plusieurs milliers de clients connectés. C'est également très motivant de pouvoir développer un logiciel qui sera sans doute utilisé et étendu avec de nouvelles fonctionnalités dans le futur.

8.5 Remerciements

Je tiens tout particulièrement à remercier les personnes qui, de près ou de loin, m'ont offert de leur temps afin de m'encadrer, renseigner et partager quelques points de vue durant ce projet :

MM. François Buntschu et Patrick Gaillet, Professeurs en Télécommunications à la Haute Ecole d'Ingénierie et d'Architecture de Fribourg, qui m'ont suivi et supervisé durant tout le travail et auprès desquels j'ai pu bénéficier d'une grande expérience dans le domaine des réseaux IP.

MM. Alain Schmoutz et Patrick Gaudin, mandants du projet, Ingénieur chef de projets respectivement Responsable Infrastructure et Services chez net+ FR, qui m'ont transmis leurs expériences et connaissances dans les réseaux de distribution IPTV.

8.6 Contenu du CD

- 01 - Administration
  - 01 - Directives et règlements du projet
  - 02 - Donnée du projet
- 02 - Gestion de projet
  - 01 - Invitations aux séances
  - 02 - Procès-verbaux des séances
  - 03 - Planification
  - 04 - Journal de travail
- 03 - Simulateur
  - 01 - Simulateur
  - 02 - Plan de tests de fonctionnement
  - 03 - Simulations de validation
- 04 - Documentation
  - 01 - Rapport final
  - 02 - Présentation
- 05 - Ressources
  - 01 - Données de références électroniques

8.7 Déclaration d'honneur

Je, soussigné, Romain Froidevaux, déclare sur l'honneur que le travail rendu est le fruit d'un travail personnel. Je certifie ne pas avoir eu recours au plagiat ou à toutes autres formes de fraudes. Toutes les sources d'information utilisées et les citations d'auteur ont été clairement mentionnées.

Romain Froidevaux

9 Références

9.1 Analyse

[UMC01] Communauté, "Multicast", *Wikipedia*, 2015. [En ligne]. Disponible: http://en.wikipedia.org/wiki/Multicast. [Consulté le 10 mars 2015].

[UMC02] White Paper, "Guidelines for Enterprise IP Multicast", *Cisco*, 2015. [En ligne]. Disponible: http://www.cisco.com/c/dam/en/us/support/docs/ip/ip-multicast/ipmlt_wp.pdf. [Consulté le 10 mars 2015].

[UMC03] F. Buntschu, Haute Ecole d'Ingénierie et d'Architecture de Fribourg, "Multicast", 2012.

[UMC04] IANA, "IP Multicast Address Space Registry", *IANA*, 2014. [En ligne]. Disponible: http://www.iana.org/assignments/multicast-addresses/multicast-addresses.xhtml. [Consulté le 10 mars 2015].

[UMC05] IANA, "IP Multicast Address Space Registry", *IANA*, 2014. [En ligne]. Disponible: http://www.iana.org/assignments/multicast-addresses/multicast-addresses.xhtml. [Consulté le 10 mars 2015].

[UMC06] Technical Team, "IP Multicast", *H3C*, 2014. [En ligne]. Disponible: http://www.h3c.com/portal/Products___Solutions/Technology/IP_Multicast/. [Consulté le 10 mars 2015].

[UMC07] IETF Community, "RFC 3376 - IGMPv3", *IETF*, 2002. [En ligne]. Disponible: https://tools.ietf.org/html/rfc3376. [Consulté le 10 mars 2015].

[UMC08] IETF Community, "RFC 2236 – IGMPv2", *IETF*, 1997. [En ligne]. Disponible: https://tools.ietf.org/html/rfc2236. [Consulté le 10 mars 2015].

[UMC09] IETF Community, "RFC 2326 – RTSP", *IETF*, 1998. [En ligne]. Disponible: https://tools.ietf.org/html/rfc2326. [Consulté le 10 mars 2015].

[UMC10] N. Meneceur, Réseau Académique Parisien, "Quelques mots sur la technologie multicast", 2012.

[UMC11] N. Meneceur, Réseau Académique Parisien, "Quelques mots sur la technologie de streaming", 2008.

[UMC12] B. Rapacchi et B. Tuy, CNRS/UREC, "Multicast protocoles de routage", 2007.

[UMC13] T. Eckert, Cisco, "IP Multicast for IPTV and beyond", 2006.

[UMC14] Cisco Engineering, Cisco, "Introduction to IP Multicast", 2006.

[OEX01] "TestCenter Live", *Spirent*, 2015. [En ligne]. Disponible: http://www.spirent.com/Products/Spirent-TestCenter_Live. [Consulté le 12 mars 2015].

[OEX02] "SPT-N11U", *Spirent*, 2015. [En ligne]. Disponible: http://www.spirent.com/Ethernet_Testing/platforms/11u_chassis. [Consulté le 12 mars 2015].

[OEX03] Communication Service, Spirent, "Spirent TestCenter IPTV Test Package", 2010.

[OEX04] W. Agoura, IXIA, "Multicast Test Plan Primer", 2005.

[OEX05] Technical Team, IXIA, "Multicast: Conformance and Performance Testing", 2005.

[OEX06] Technical Team, IXIA, "WaveTest Multi-client Traffic Generator / Performance Analyzer", 2015.

[OEX07] Technical Team, NextraGen, "TraceSim Call generator for VoIP and Video", 2015.

[OEX08] Technical Team, IXIA, "Multi-Application Traffic Generation with Ixia AppLibrary", 2013.

[OEX09] Communauté, "Scapy", *Wikipedia*, 2015. [En ligne]. Disponible: http://www.secdev.org/projects/scapy/. [Consulté le 12 mars 2015].

[OEX10] Communication Service, "Twisted", *Twisted Matrix*, 2015. [En ligne]. Disponible: https://twistedmatrix.com/trac/. [Consulté le 12 mars 2015].

[OEX11] Communication Service, "VLC Media Player", *Video LAN*, 2015. [En ligne]. Disponible: http://www.videolan.org/vlc/index.html. [Consulté le 12 mars 2015].

[OEX12] Communauté, "VLC Command Line", *Video LAN*, 2015. [En ligne]. Disponible: https://wiki.videolan.org/VLC_command-line_helphttp://www.videolan.org/vlc/index.html. [Consulté le 12 mars 2015].

[OEX13] Communauté, "VLC Python Bindings", *Video LAN*, 2015. [En ligne]. Disponible: https://wiki.videolan.org/Python_bindings. [Consulté le 12 mars 2015].

[OEX14] Developers, "VLC Python GIT", *Video LAN*, 2015. [En ligne]. Disponible: http://git.videolan.org/?p=vlc/bindings/python.git. [Consulté le 12 mars 2015].

[OEX15] Helix Community, "Helix Player", *Helix*, 2015. [En ligne]. Disponible: https://player.helixcommunity.org. [Consulté le 18 mars 2015].

[OEX16] Microsoft Support, "Command Line Options Available for Windows Media Player", *Microsoft*, 2015. [En ligne]. Disponible: https://support.microsoft.com/en-us/kb/241422. [Consulté le 18 mars 2015].

[OEX17] GStreamer, "News and presentation", *GStreamer*, 2015. [En ligne]. Disponible: http://gstreamer.freedesktop.org. [Consulté le 18 mars 2015].

[OEX18] EvoStream, "Products", *EvoStream*, 2015. [En ligne]. Disponible: https://evostream.com/products/on-servers/. [Consulté le 18 mars 2015].

[OEX19] FFmpeg, "Documentation", *FFmpeg*, 2015. [En ligne]. Disponible: https://www.ffmpeg.org/documentation.html. [Consulté le 18 mars 2015].

[OEX19] Anevia, "Produits et services", *Anevia Group*, 2015. [En ligne]. Disponible: http://www.anevia-group.com/produits-services/. [Consulté le 18 mars 2015].

9.2 Conception

[CON01] "ConfigParser for Python3", *Python*, 2015. [En ligne]. Disponible: https://docs.python.org/3.4/library/configparser.html. [Consulté le 08 avril 2015].

[CON02] "ConfigParser for Python 2.7", *Python*, 2013. [En ligne]. Disponible: https://docs.python.org/2/library/configparser.html. [Consulté le 19 avril 2015].

[CON03] D. Beazley, PyCon, "Understanding the Python GIL", 2010.

[CON04] J. Noller, PyWorks, "Getting started with concurrency", 2008.

[CON05] D. M. Beazley, O'Reilly Open Source Conference, "Advanced Python programming", 2000.

[CON06] "PEP8 - Python Styling guide", *Python*, 2015. [En ligne]. Disponible: https://www.python.org/dev/peps/pep-0008. [Consulté le 08 avril 2015].

[CON07] "Logging", *Python*, 2015. [En ligne]. Disponible: https://docs.python.org/3.4/library/logging.html. [Consulté le 08 avril 2015].

[CON08] "Python 3.4", *Python*, 2015. [En ligne]. Disponible: https://www.python.org. [Consulté le 08 avril 2015].

[CON09] "Subprocess management", *Python*, 2015. [En ligne]. Disponible: https://docs.python.org/3/library/subprocess.html. [Consulté le 20 avril 2015].

[CON10] "libpcap", *TCP Dump*, 2015. [En ligne]. Disponible: http://www.tcpdump.org. [Consulté le 29 avril 2015].

[CON11] "pypcap", *Python*, 2015. [En ligne]. Disponible: https://pypi.python.org/pypi/pypcap. [Consulté le 29 avril 2015].

9.3 Tests et validation

[VAL01] IETF Community, "RFC 2326 - RTSP", *IETF*, 1998. [En ligne]. Disponible: https://www.ietf.org/html/rfc2326. [Consulté le 27 avril 2015].

[VAL02] IETF Community, "RFC 3376 - IGMPv3", *IETF*, 2002. [En ligne]. Disponible: https://tools.ietf.org/html/rfc3376. [Consulté le 27 avril 2015].

[VAL03] IETF Community, "RFC 2236 - IGMPv2", *IETF*, 1997. [En ligne]. Disponible: https://tools.ietf.org/html/rfc2236. [Consulté le 27 avril 2015].

[VAL04] IETF Community, "RFC 3550 - RTP", *IETF*, 2003. [En ligne]. Disponible: https://www.ietf.org/html/rfc3550. [Consulté le 27 avril 2015].

[VAL05] IETF Community, "RFC 768 - UDP", *IETF*, 1980. [En ligne]. Disponible: https://www.ietf.org/html/rfc768. [Consulté le 27 avril 2015].

[VAL06] IETF Community, "RFC 793 - TCP", *IETF*, 1981. [En ligne]. Disponible: https://www.ietf.org/html/rfc793. [Consulté le 27 avril 2015].

[VAL07] Gnome, "System Monitor", *GNOME*, 2015. [En ligne]. Disponible: https://help.gnome.org/users/gnome-system-monitor. [Consulté le 01 mai 2015].

[VAL08] Wireshark, "Wireshark Display Filtering", *Wireshark*, 2015. [En ligne]. Disponible: https://wiki.wireshark.org/DisplayFilters. [Consulté le 01 mai 2015].

[VAL09] Wireshark, "Wireshark RTP Analysis", *Wireshark*, 2015. [En ligne]. Disponible: https://wiki.wireshark.org/RTP. [Consulté le 01 mai 2015].

10 Annexes

1. Donnée du projet
2. Planification
3. Liste des tests
4. Manuel utilisateur du simulateur
5. Manuel de diffusion de flux IPTV avec VLC

11 Table des illustrations

12 Lexique

ACK	Acknowledge, message validant la réception d'un élément
AS	Autonomous System, entité d'Internet constituée de plusieurs réseaux
ASIC	Application-Specific Integrated Circuit, hardware dédié à une opération spécifique
B2B	Business to Business, fourniture de services d'entreprise à entreprise
B2C	Business to Consumer, fourniture de services d'une entreprise à des clients privés
BNG	Broadband Network Gateway, routeur d'agrégation des réseaux Core et Métro
Broadcast	Diffusion de Point à Tous les points
CAM	Content Addressable Memory, dénomination Cisco pour la table des adresses MAC
CGMP	Protocole propriétaire Cisco, pendant d'IGMP
CLI	Console Line Interface, interface en ligne de commande
CPE	Customer Premises Equipment, équipement chez le client le raccordant à l'opérateur
CPU	Processeur d'un élément actif *(routeur, switch, ordinateur, etc…)*
DOCSIS	Technologie d'accès basée sur une infrastructure CATV
DR	Rôle Designated Router d'OSPF
DUT	Device Under Test, dénomination d'un équipement en cours de test
DVMPR	Distance Vector Multicast Routing Protocol
Epydoc	Outil de génération de documentation pour les API
FF	Fast-Forward, avance rapide sur les systèmes VCR
FTTH	Fiber To The Home, accès à fibre optique de l'opérateur jusque chez le client
GLOP	Méthode d'adressage multicast IPv4 basé sur le numéro d'AS
GPL	GNU Public License, type de licence Open-Source
GUI	Graphical User Interface, interface graphique
HEIA-FR	Haute Ecole d'Ingénierie et d'Architecture de Fribourg (www.heia-fr.ch)
HES-SO	Haute Ecole Spécialisée de Suisse Occidentale (www.hes-so.ch)
IANA	Autorité de régulation des adresses et noms de domaine sur Internet
IETF	Internet Exchange Task Force, organisation de certification de l'Internet
IGMP	Internet Group Management Protocol, protocole de gestion des flux multicast en couche 2
IP	Internet Protocol, protocole de transport de couche 3
IPTV	Télévision sur IP
Live	Diffusion en direct
MIT	Massachusetts Institute of Technology, type de licence Open-Source et établissement scolaire
MOSPF	Multicast OSPF
MPEG	Codec vidéo adaptatif
Multicast	Diffusion de Point à Multipoints
NACK	Negative Acknowledge, message indiquant la non réception d'un élément
net+ FR	Fournisseur d'accès fribourgeois (www.netplusfr.ch)
OCBT	Ordered Code Based Tree, ancien protocole de routage multicast
PC	Personal Computer
PGM	Pragmatic General Multicast, protocole multicast implémentant un concept de fiabilité
PIM	Protocol Independant Multicast, protocole de routage multicast
PIM-DM	Protocole de routage multicast PIM en mode dense
PIM-SM	Protocole de routage multicast PIM en mode épars

QoS	Quality of Service, qualité de service
QoSMIC	Quality of Service Sensitive Multicast Internet Protocol
RDT	Real Data Transport, analogue au RTP mais propriétaire de RealNetworks
Replay	Contenu rediffusé en Unicast, destiné à un client précis
REW	Rewind, retour en arrière sur les systèmes VCR
RFC	Request For Comment, publication de l'IETF
RP	Routeur Rendez-vous Point
RPF	Reverse Path Forwarding, algorithme de détection du chemin multicast optimal
RTCP	RTP Control Protocol, couche de contrôle du protocole RTP *(Port UDP RTP + 1)*
RTP	Real Time Protocol, basé sur UDP et optimisé pour les flux live
SIP	Session Initiation Protocol, plan de contrôle de la voix sur IP entre-autre
Slow Start	Algorithme TCP de régulation des flux pour éviter la congestion
Snooping	Fait d'observer du contenu sans l'altérer
SNR	Signal to Noise Ratio, rapport signal sur bruit
SPT	Shortest-Path Tree, algorithme PIM pour trouver le chemin multicast optimal
STB	Set-Top Box, élément client recevant les flux IPTV pour les retransmettre à la TV
STP	Spanning Tree Protocol, algorithme de couche 2 permettant d'éviter les boucles de réseau
TCP	Transfer Control Protocol, protocole de couche 4 orienté connexion
Triple Play	Fourniture multi-services IP *(Téléphonie, Internet et Télévision)* sur un même accès
TTL	Time To Live, durée de vie d'un paquet
TV	Télévision
UDP	User Datagram Protocol, protocole de couche 4 orienté sans connexion
Unicast	Diffusion de Point à Point
URI	Uniform Resource Identifier, identificateur d'une ressource
VCR	Video Cassette Recording
VLC	Lecteur vidéo multimédia universel
VM	Virtual Machine, hôte virtuel s’exécutant sur une machine physique
VoD	Video on Demand, contenu multimédia disponible à la demande
VoIP	Voice over IP, téléphonie sur le réseau IP
xDSL	Technologie d'accès basée sur une infrastructure téléphonique traditionnelle

Annexe 1 : Donnée du projet

netplusFR, actif sur le canton de Fribourg, est un fournisseur de services de télécommunications autant pour les particuliers que pour les entreprises. L'entreprise est née en janvier 2013 de la fusion des activités de télécommunications de Groupe E, Gruyère Energie et IB Murten. netplusFR fournit un grand nombre de services à ses clients résidentiels et business sur différentes technologies d'accès : CATV, VDSL et FTTH. L'entreprise offre des services B2C triple-play (Internet, téléphonie et TV) and B2B tels que VPN, E-LINEs et Internet à haut dé-bit.

Afin d'optimiser son réseau de transport de services et donc la qualité des services offerts à ses clients, netplusFR recherche des moyens de faire des tests de charge sur ses équipements (Backbone, Access et modem destinés aux clients finaux), de mesurer la qualité de la transmission de flux TV (Live et replay) sur son réseau, de tester la QoS mise en place, …

netplusFR souhaite dans un premier temps avoir un outil client simple qui permet de faire des requêtes vers des sources pour établir des flux multicast (TV live) et unicast (TV replay).
Il doit être possible de définir le nombre de flux multicast et unicast à démarrer afin de pou-voir tester la charge des équipements réseaux.
Dans un 2ème temps, toutes sortes d'autres fonctionnalités de mesures de la QoS, qualité de réception, … peuvent être développées.

Dans ce but, SCAPY est un exemple de logiciel libre de manipulation de paquets réseau écrit en Python. Il est capable d'intercepter le trafic sur un segment réseau, de générer des paquets dans un nombre important de protocoles, d'analyser le réseau informatique, … (Source wikipedia).

Objectifs :

- Analyser les solutions software de génération de paquets réseau
- Spécifier une architecture software
- Développement logiciel d'un client simple de génération de requêtes de flux multicast (TV live) et unicast (TV replay)
- Mise en œuvre d'une maquette de démonstration
- Tests et démonstrations

Proposé par	NetPlus Fribourg
Nombre d'étudiants	1
Etudiants inscrits	Romain Froidevaux
Mots clés	TV live/replay, multicast, unicast, RTSP, FTTH, tests de charges, QoS, SCAPY, Python
Responsables internes	François Buntschu Patrick Gaillet
Responsables externes	Alain Schmoutz Patrick Gaudin
Liens	Scapy

Annexe 2 : Planification du projet

	P1 - 16.02.2015							P2 - 23.03.2015							P3 - 02.03.2015							P4 - 09.03.2015						
Nom	Lu	Ma	Me	Je	Ve	Sa	Di	Lu	Ma	Me	Je	Ve	Sa	Di	Lu	Ma	Me	Je	Ve	Sa	Di	Lu	Ma	Me	Je	Ve	Sa	Di
Début du projet			◆																									
Cahier des charges et planification																												
Rédaction des drafts																												
Présentation des drafts										◆																		
Rédaction des CDC et Planification finaux																												
Rendu du cahier des charges et planification à 16h00																	◆											
Analyse																												
Analyse technique orientée flux TV																												
Analyse des outils existants																												
Rendu du document d'analyse à 18h00																												
Spécifications																												
Définition des spécifications																												
Choix de la plateforme / language																												
Rendu du documents des spécifications à 18h00																												
Mise en place maquette de tests																												
Montage maquette labo HEIA-FR																												
Tests de la maquette																												
Conception																												
Réalisation du client pour le labo HEIA-FR																												
Rendu de la pré-version du client labo HEIA-FR à 18h00																												
Tests et validation																												
Préparation de la procédure de test																												
Réalisation des tests et adaptations																												
Rendu de la version finale du client labo HEIA-FR																												
Intégration net+																												
Adaptation du client																												
Tests et validation																												
Rapport final																												
Finalisation du rapport final																												
Rendu du rapport final à 16h00																												
Présentation finale																												
Préparation de la défense																												
Défense orale																												
Fin du projet																												

	P5 - 16.03.2015							P6 - 23.03.2015							P7 - 30.03.2015							P8 - 13.04.2015						
Nom	Lu	Ma	Me	Je	Ve	Sa	Di	Lu	Ma	Me	Je	Ve	Sa	Di	Lu	Ma	Me	Je	Ve	Sa	Di	Lu	Ma	Me	Je	Ve	Sa	Di
Début du projet																												
Cahier des charges et planification																												
Rédaction des drafts																												
Présentation des drafts																												
Rédaction des CDC et Planification finaux																												
Rendu du cahier des charges et planification à 16h00																												
Analyse																												
Analyse technique orientée flux TV																												
Analyse des outils existants																												
Rendu du document d'analyse à 18h00																												
Spécifications																												
Définition des spécifications																												
Choix de la plateforme / language																												
Rendu du documents des spécifications à 18h00																												
Mise en place maquette de tests																												
Montage maquette labo HEIA-FR																												
Tests de la maquette																												
Conception																												
Réalisation du client pour le labo HEIA-FR																												
Rendu de la pré-version du client labo HEIA-FR à 18h00																												
Tests et validation																												
Préparation de la procédure de test																												
Réalisation des tests et adaptations																												
Rendu de la version finale du client labo HEIA-FR																												
Intégration net+																												
Adaptation du client																												
Tests et validation																												
Rapport final																												
Finalisation du rapport final																												
Rendu du rapport final à 16h00																												
Présentation finale																												
Préparation de la défense																												
Défense orale																												
Fin du projet																												

	P9 - 20.04.2015							P10 - 27.04.2015							P11 - 04.05.2015							P12 - 11.05.2015						
Nom	Lu	Ma	Me	Je	Ve	Sa	Di	Lu	Ma	Me	Je	Ve	Sa	Di	Lu	Ma	Me	Je	Ve	Sa	Di	Lu	Ma	Me	Je	Ve	Sa	Di
Début du projet																								◆				
Cahier des charges et planification																												
Rédaction des drafts																												
Présentation des drafts																												
Rédaction des CDC et Planification finaux																												
Rendu du cahier des charges et planification à 16h00																												
Analyse																												
Analyse technique orientée flux TV																												
Analyse des outils existants																												
Rendu du document d'analyse à 18h00																												
Spécifications																												
Définition des spécifications																												
Choix de la plateforme / language																												
Rendu du documents des spécifications à 18h00																												
Mise en place maquette de tests																												
Montage maquette labo HEIA-FR																												
Tests de la maquette																												
Conception								◆																				
Réalisation du client pour le labo HEIA-FR																												
Rendu de la pré-version du client labo HEIA-FR à 18h00								◆																				
Tests et validation															◆													
Préparation de la procédure de test																												
Réalisation des tests et adaptations																												
Rendu de la version finale du client labo HEIA-FR															◆													
Intégration net+																												
Adaptation du client																												
Tests et validation																												
Rapport final																												
Finalisation du rapport final																												
Rendu du rapport final à 16h00																								◆				
Présentation finale																												
Préparation de la défense																												
Défense orale																												
Fin du projet																												

	P13 - 18.05.2015						
Nom	Lu	Ma	Me	Je	Ve	Sa	Di
Début du projet			◆				
Cahier des charges et planification							
Rédaction des drafts							
Présentation des drafts							
Rédaction des CDC et Planification finaux							
Rendu du cahier des charges et planification à 16h00							
Analyse							
Analyse technique orientée flux TV							
Analyse des outils existants							
Rendu du document d'analyse à 18h00							
Spécifications							
Définition des spécifications							
Choix de la plateforme / language							
Rendu du documents des spécifications à 18h00							
Mise en place maquette de tests							
Montage maquette labo HEIA-FR							
Tests de la maquette							
Conception							
Réalisation du client pour le labo HEIA-FR							
Rendu de la pré-version du client labo HEIA-FR à 18h00							
Tests et validation							
Préparation de la procédure de test							
Réalisation des tests et adaptations							
Rendu de la version finale du client labo HEIA-FR							
Intégration net+							
Adaptation du client							
Tests et validation							
Rapport final							
Finalisation du rapport final							
Rendu du rapport final à 16h00							
Présentation finale							
Préparation de la défense							
Défense orale			◆				
Fin du projet			◆				

Légende:

- Temps planifié
- Marge
- Délai fixe
- Délai planifié

Annexe 03 : Liste des tests

N°	Description	Résultat attentu	Status
100	**Interface CLI et sorties dans la console**		**100%**
101	Nombre de flux Unicast	Affichage du nombre de flux Unicast défini dans le fichier de configuration	OK
102	Nombre de flux Multicast	Affichage du nombre de flux Multicast défini dans le fichier de configuration	OK
103	Heure de début de simulation	Affichage de l'heure de début de la simulation	OK
104	Heure de fin de simulation	Affichage de l'heure de fin de la simulation	OK
105	Type de départ de simulation	Affichage du type de départ pour le lancement des flux *(Instant, Linear ou Exponential)*	OK
106	Type de fin de simulation	Affichage du type de fin pour l'arrêt des flux *(Instant, Linear ou Exponential)*	OK
107	Type de choix des channels	Affichage du choix de sélection des canaux *(Fixes ou Random)*	OK
200	**Flux Multicast**		**100%**
201	Plan de contrôle	Le plan de contrôle est IGMPv3 *(IGMPv2 si l'OS est en mode compatibilité)*	OK
202	Plan utilisateur	Le plan utilisateur est RTP	OK
203	Le flux de données n'est pas décodé	Les flux de données arrivant sur la carte réseau du simulateur ne sont pas décodés par VLC	OK
204	Départ instantané	Les flux Multicast sont lancés instantanément	OK
205	Départ linéaire	Les flux Multicast sont lancés de manière linéaire, d'un delta en ms selon le paramètre du fichier de configuration	OK
206	Départ exponentiel	Les flux Multicast sont lancés de manière exponentielle	OK
207	Sélection aléatoire des groupes Mutlicast	Les groupes Multicast sont sélectionnés dans la liste de manière aléatoire pondérée selon le poids attribué	OK
208	Sélection fixe des groupes Multicast	Les groupes Mutlicsat sont sélectionnés dans la liste de manière fixe *(dans l'ordre)*	OK
209	Overflow de la sélection des groupes Multicast	En cas d'overflow *(plus de process que de channels)*, application de l'algorithme du tampon circulaire	OK
210	Fonction de zapping	La fonction de Zapping fait changer le processus Multicast de groupe périodiquement	OK
211	Arrêt du flux de données au zapping	Le flux de données du channel précédent est correctement arrêté au moment du zapping	OK
300	**Flux Unicast**		**82%**
301	Plan de contrôle	Le plan de contrôle est RTSP	OK
302	Plan utilisateur	Le plan utilisateur est RTP	OK
303	Le flux de données n'est pas décodé	Les flux de données arrivant sur la carte réseau du simulateur ne sont pas décodés par VLC	OK
304	Départ instantané	Les flux Unicast sont lancés instantanément	OK
305	Départ linéaire	Les flux Unicast sont lancés de manière linéaire, d'un delta en ms selon le paramètre du fichier de configuration	OK
306	Départ exponentiel	Les flux Unicast sont lancés de manière exponentielle	OK
307	Sélection aléatoire des médias	Les médias sont sélectionnés de la liste de manière aléatoire pondérée selon le poids attribué	OK
308	Sélection fixe des médias	Les médias sont sélectionnés de la liste de manière fixe *(dans l'ordre)*	OK
309	Overflow de la sélection des médias	En cas d'overflow *(plus de process que de channels)*, application de l'algorithme du tampon circulaire	OK
310	Fonction de pausing	La fonction de Pausing met en pause les flux Unicast périodiquement	NOK
311	Arrêt du flux de données au pausing	Le flux de données est bien arrêté lors de la mise en pause du flux	N/A
400	**Fichier log**		**100%**
401	Niveau de sévérité	Le niveau de sévérité des logs est à INFO *(toutes les informations)*	OK
402	Format de fichier	Le fichier log généré est au format texte, avec une extension .log	OK
403	Emplacement des logs	Les logs sont placés à la racine du simulateur, dans un dossier ./logs	OK

N°	Description	Résultat attentu	Status
404	Création des fichiers logs	A chaque simulation, un nouveau fichier log est créé	OK
405	Nom des fichiers log	Les fichiers logs sont nommés avec la date et l'heure	OK
406	En-tête des logs	Chaque entrée du fichier log contient la date et l'heure *(avec millisecondes)* et le niveau de sévérité	OK
500	**Fichiers de configuration**		**100%**
501	Format de fichier	Les fichiers de configuration sont au format texte, avec une extension .conf	OK
502	Modification avec Excel	Les fichiers de configuration sont modifiables avec Excel	OK
503	Erreur dans le fichier channels.conf	Une erreur est retournée *(dans le log)* en cas d'erreur dans le fichier channels.conf et la simulation est arrêtée	OK
504	Erreur dans le fichier simu.conf	Une erreur est retournée *(dans le log)* en cas d'erreur dans le fichier simu.conf et la simulation est arrêtée	OK
505	Pas de paramètre au lancement	Une erreur est retournée *(dans le log)* au cas où les fichiers de configuration ne sont pas passés en paramètre	OK
600	**Général**		**100%**
601	Compatible Python 2.7	Le simulateur est compatible avec Python 2.7	OK
602	Compatible Python 3.4	Le simulateur est compatible avec Python 3.4	OK
603	Durée de simulation défini	La simulation dure le nombre de secondes défini dans le fichier de configuration	OK
604	Durée de simulation indéfini	La simulation dure indéfiniement *(jusqu'à un CTRL+C)*	OK
605	Arrêt forcé de la simulation	La commande CTRL+C arrête proprement tous les flux en cours et stoppe la simulation	OK
606	Nombre de flux Multicast	Le nombre de flux Multicast indiqué dans le fichier de configuration correspond au nombre lancé	OK
607	Nombre de flux Unicast	Le nombre de flux Unicast indiqué dans le fichier de configuration correspond au nombre lancé	OK
		Résultats des tests	**98%**

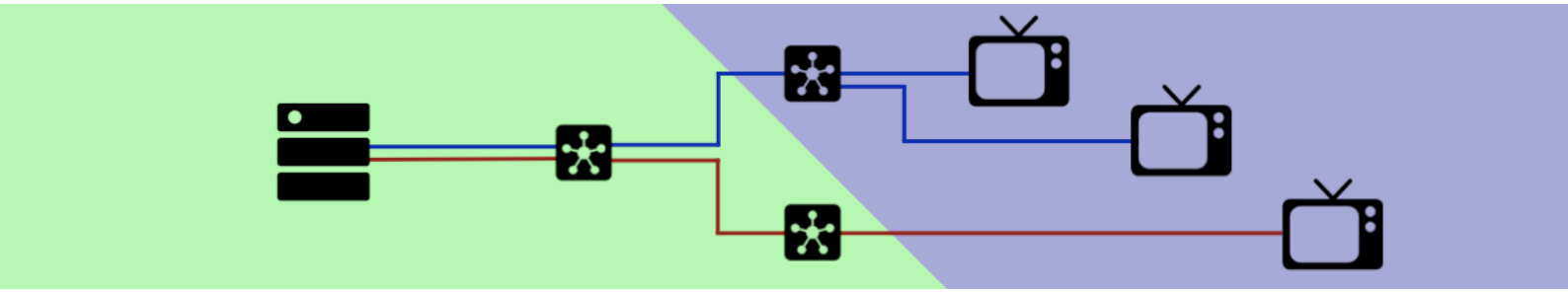
Haute école d'ingénierie et d'architecture Fribourg
Hochschule für Technik und Architektur Freiburg

Bd de Pérolles 80
case postale 32
CH-1705 Fribourg
t. +41 (0)26 429 66 11
f. +41 (0)26 429 66 00
www.heia-fr.ch

net+ SimBox

Projet de Semestre 6

ANNEXE 4 : MANUEL UTILISATEUR



Titre du projet	Simulateur de Set-Top Box	
Filière	Télécommunications orientation Réseaux et Sécurité	
Etudiant	Romain Froidevaux	<romain.froidevaux@edu.hefr.ch>
Professeurs	François Buntschu Patrick Gaillet	<francois.buntschu@hefr.ch> <patrick.gaillet@hefr.ch>
Mandants	Alain Schmoutz Patrick Gaudin	<alain.schmoutz@netplusfr.ch> <patrick.gaudin@netplusfr.ch>
Classe	T-3a	
Date	06.05.2015	
Version	1.1	

Hes·so FRIBOURG FREIBURG
Haute Ecole Spécialisée de Suisse occidentale
Fachhochschule Westschweiz
University of Applied Sciences and Arts Western Switzerland

HISTORIQUE DU DOCUMENT

Version	Auteur	Description des modifications	Date
0.1	Romain Froidevaux	Création du document, « DRAFT 1 »	04.05.2015
1.0	Romain Froidevaux	Validation du document	06.05.2015
1.1	Romain Froidevaux	Corrections orthographiques	06.05.2015

TABLE DES MATIERES

net+ SimBox

1 INTRODUCTION

Le manuel d'utilisation a pour but de décrire le fonctionnement du simulateur du point de vue utilisateur. Il détaillera également l'environnement dans lequel il peut être exécuté et la procédure d'installation.

Cette documentation s'adresse aux techniciens et ingénieurs, disposant des notions de base en IPTV et devant utiliser le simulateur.

2 DESCRIPTION DES FONCTIONNALITES

Le simulateur net+ SimBox a été développé dans le but de réaliser un test de charge sur un CPE dans un environnement IPTV. Il permet de s'abonner à des flux TV Live et Replay au travers d'une interface CLI développée en Python.

Ses fonctionnalités principales sont les suivantes :

Fonctionnalité	Description
Réception Unicast	Demande de flux Unicast, simulant la demande et réception d'un contenu TV Replay.
Abonnement Multicast	Abonnement à des groupes Multicast, simulant le visionnement d'un chaîne TV en Live.
Types de lancements et arrêts	Possibilité de plusieurs types de lancements et d'arrêt des flux : • Linéaire : Séparation de *n* millisecondes entre flux • Immédiat : Enchaînement immédiat • Exponentiel : Séparation d'une valeur exponentielle
Durée de simulation	Définition d'une durée de la simulation, à partir de laquelle tous les flux sont lancés *(peut être définie ou illimitée).*
Arrêt de la simulation	La simulation en cours peut être arrêtée à tout moment au moyen de la combinaison CTRL+C. Ainsi, tous les flux sont stoppés correctement.
Mode zapping	Possibilité d'activer le mode zapping aux flux multicast, simulant un client passant d'une chaîne à un autre périodiquement.
Sélection des canaux	Algorithme de sélection des canaux, soit fixe soit aléatoire.
Pondération des médias	Attribution d'une pondération aux médias, permettant de simuler des demandes variables sur certaines chaînes ou contenus Replay.

Tableau 1 : Description des fonctionnalités

3 ENVIRONNEMENT D'EXECUTION

Il est fortement conseillé d'exécuter le simulateur sur le système d'exploitation Debian. Cette distribution de Linux est extrêmement légère et permet d'allouer un maximum de ressources à l'exécution du simulateur.

Sur la base d'une installation neuve, il est **recommandé** d'installer les outils suivants :

1. Sudo

```
aptitude install sudo
sudo adduser <username> sudo    #Remplacer <username> avec votre login courant
sudo reboot
```

Pour tous les points suivants, il est considéré que le package Sudo est installé. Si ce n'est pas le cas, il est nécessaire d'utiliser aptitude pour l'installation des packages.

2. Wireshark

```
sudo apt-get update
sudo apt-get install wireshark
sudo dpkg-reconfigure wireshark-common
sudo usermod -a -G wireshark $USER
sudo reboot
```

3. Gnome System Monitor

```
sudo apt-get update
sudo pt-get install gnome-system-monitor
```

Il est ensuite **nécessaire** d'installer les prérequis suivants pour l'exécution du simulateur :

1. Python 3

```
sudo apt-get update
sudo apt-get install python3
```

2. VLC Media Player

```
sudo apt-get update
sudo apt-get install vlc
```

Pour éditer le code source Python du simulateur, il est **recommandé** d'installer les outils suivants :

1. IDE PyCharm Community Edition :

```
sudo apt-get update
sudo apt-get install openjdk-7-jre
```

Télécharger PyCharm Community Edition : https://www.jetbrains.com/pycharm/download/
Décompresser l'archive
Depuis un Terminal se rendre dans le dossier décompressé avec la commande `cd`
Lancer PyCharm en tapant `./bin/pycharm.sh`

4 CREATION DES FICHIERS DE CONFIGURATION

Les différents paramètres de la simulation sont définis dans deux fichiers de configuration :

Fichier	Description
channels.conf	Liste des groupes multicast et des médias Replay disponibles
simu.conf	Enumération des différents paramètres relatifs à la simulation

Tableau 2 : Liste des fichiers de configuration

Des exemples de ces deux fichiers sont fournis avec le simulateur dans le dossier `./examples`. Il est conseillé de copier des deux références à la racine du simulateur avant de les éditer.

4.1 FICHIER CHANNELS.CONF

Le fichier est segmenté en deux blocs, définissant les médias Replay et les groupes Multicast disponibles. Par exemple :

```
[multicast]
mcst1: 239.1.1.1 1
mcst2: 239.2.2.2 10
mcst2: 239.3.3.3 1

[unicast]
ucst1: rtsp://10.0.0.8:8554/stream_1 1
ucst2: rtsp://10.0.0.1:8554/stream_2 5
ucst3: rtsp://10.0.0.2:8554/stream_3 1
```

Figure 1 : Exemple du fichier channels.conf

Pour chaque entrée dans les groupes, la structure est la suivante, les données en rouge étant à adapter :

Multicast : `mcstx: multicast_ip priority`
Unicast : `ucstx: rtsp://server_ip:port/stream_path priority`

L'indice des flux *(x)* doit être incrémenté de 0 à n pour chaque entrée du bloc.

La priorité doit être une valeur entière supérieure à 1, qui influence le choix du média en cas de simulation aléatoire pondérée. Ainsi, une entrée avec une priorité de *n* a *n* fois plus de chance d'être choisie qu'une entrée avec une priorité de 1.

4.2 FICHIER SIMU.CONF

Ce fichier ne contient qu'un bloc dans lequel sont décrits tous les paramètres relatifs à la simulation.

```
[simulation]
nb_multicast:   10
nb_unicast:     0
start_type:     100
stop_type:      200
simu_duration:  20
zapping_enable: False
pause_enable:   False
channel_select: Random
```

Figure 2 : Exemple du fichier simu.conf

Les différents paramètres à renseigner sont les suivants :

Paramètre	Type	Description
`nb_multicast`	`int 0+`	Nombre de flux multicast *(chaînes TV Live)* à s'abonner
`nb_unicast`	`int 0+`	Nombre de flux unicast *(contenus TV Replay)* à jouer
`start_type`	`-1 | 0 | int 1+`	Type de lancement des flux : • -1 = exponentiel • 0 = immédiat • n = séparés de *n* millisecondes
`stop_type`	`-1 | 0 | int 1+`	Type d'arrêt des flux : • -1 = exponentiel • 0 = immédiat • n = séparés de *n* millisecondes
`simu_duration`	`int 0+`	Durée de simulation, débutant lorsque tous les flux sont lancés et expirant juste avant l'arrêt des flux.
`zapping_enable`	`True | False`	Activation du mode zapping *(changement périodique de groupe multicast, simulant le changement de chaînes)*
`pausing_enable`	`True | False`	Activation du mode pausing *(mise en pause et replay périodique des flux unicast TV Replay).*
`channel_select`	`Random | Fixed`	Définition du type de choix des médias. • Fixed : Choix linéaire dans l'ordre de la liste • Random : Choix aléatoire pondéré selon la priorité définie pour le média

Tableau 3 : Description des paramètres du fichier simu.conf

Au besoin, les fichiers de configuration peuvent être édités depuis le logiciel Microsoft Excel. Pour ce faire, il suffit de les importer dans le programme au format CSV avec le caractère d'espacement comme délimiteur.

5 LANCEMENT DE LA SIMULATION

Le lancement d'une simulation s'effectue en mode CLI depuis le terminal de l'OS. Pour des raisons de simplicité, il est préférable de se rendre à la racine du simulateur à l'aide de la commande `cd`.

Ensuite, lancer la simulation à l'aide de la commande suivante :

```
python3 controller.py simu.conf channels.conf
```

Dans le cas où les fichiers de configurations n'ont pas été placés à la racine du simulateur, il est nécessaire de renseigner le chemin absolu ou relatif depuis le répertoire courant.

Il est possible pour l'utilisateur d'interrompre à tout moment la simulation au moyen de la combinaison de touches CTRL+C.

6 ANALYSE DES LOGS

Pour chacune des simulations lancées, un fichier log est automatiquement créé dans le répertoire `./logs/`. Le fichier est nommé avec la date et l'heure du lancement.

Pour les simulations sur une longue durée, il est possible d'afficher le contenu du log en direct à l'aide de la commande suivante depuis la racine du simulateur :

```
tail -f logs/file_name.log
```

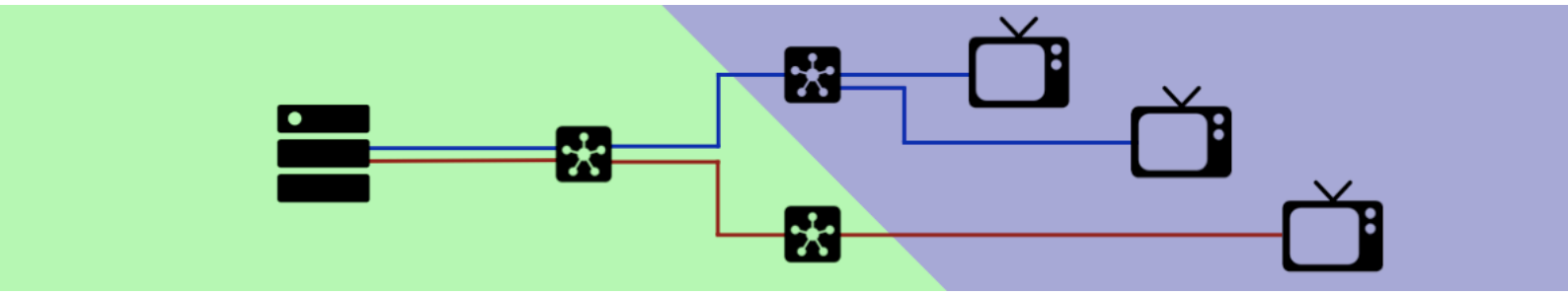
Haute école d'ingénierie et d'architecture Fribourg
Hochschule für Technik und Architektur Freiburg

Bd de Pérolles 80
case postale 32
CH-1705 Fribourg
t. +41 (0)26 429 66 11
f. +41 (0)26 429 66 00
www.heia-fr.ch

net+ SimBox

Projet de Semestre 6

ANNEXE 5 : DIFFUSION MULTIMÉDIA AVEC VLC



Titre du projet	Simulateur de Set-Top Box	
Filière	Télécommunications orientation Réseaux et Sécurité	
Etudiant	Romain Froidevaux	<romain.froidevaux@edu.hefr.ch>
Professeurs	François Buntschu	<francois.buntschu@hefr.ch>
	Patrick Gaillet	<patrick.gaillet@hefr.ch>
Mandants	Alain Schmoutz	<alain.schmoutz@netplusfr.ch>
	Patrick Gaudin	<patrick.gaudin@netplusfr.ch>
Classe	T-3a	
Date	29.04.2015	
Version	1.0	

Hes·so FRIBOURG FREIBURG
Haute Ecole Spécialisée de Suisse occidentale
Fachhochschule Westschweiz
University of Applied Sciences and Arts Western Switzerland

HISTORIQUE DU DOCUMENT

Version	Auteur	Description des modifications	Date
1.0	Romain Froidevaux	Création du document	29.04.2015

TABLE DES MATIERES

net+ SimBox

1 INTRODUCTION

Ce document annexe décrit la procédure pour diffuser des contenus multimédias en Unicast et Multicast avec VLC, pilotables à l'aide des protocoles RTSP et IGMP.

2 PREREQUIS

La procédure nécessite un système d'exploitation Linux ou Windows, avec le logiciel VLC 2.2 minimum installé.

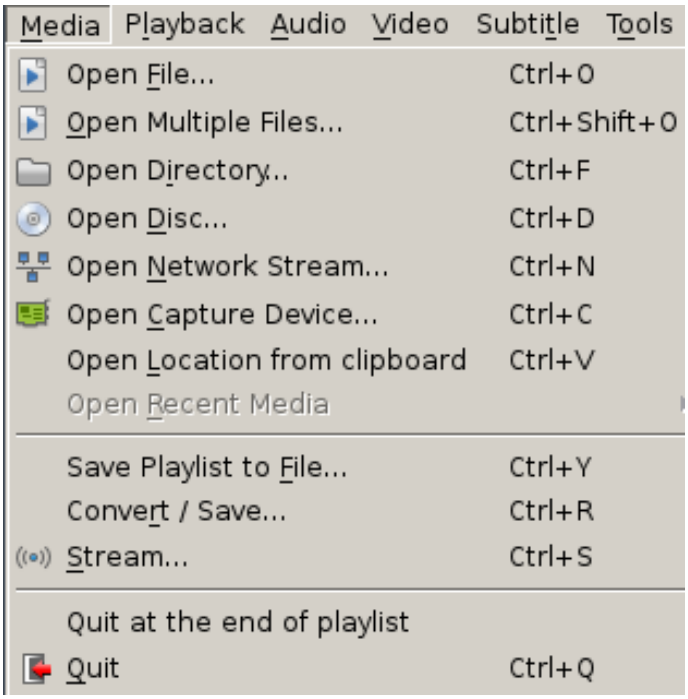
La diffusion en Multicast est possible sur OS X, mais l'Unicast piloté par RTSP n'a pas encore été implémenté pour ce système.

3 FLUX UNICAST

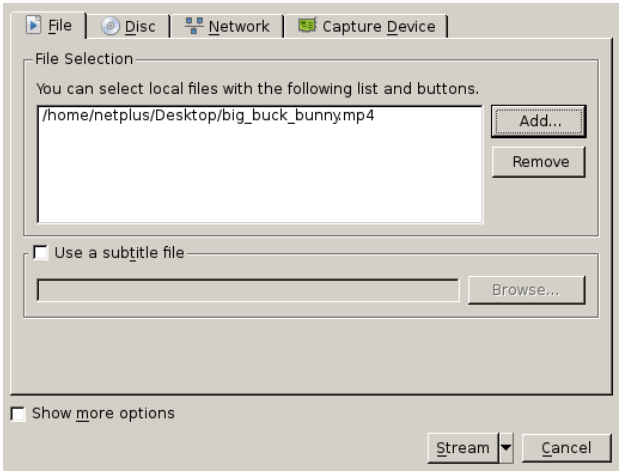
Les flux de type Unicast permettent de simuler la diffusion d'un contenu TV Replay. Il est polité par le protocole RTSP.

3.1 DIFFUSION DU FLUX UNICAST

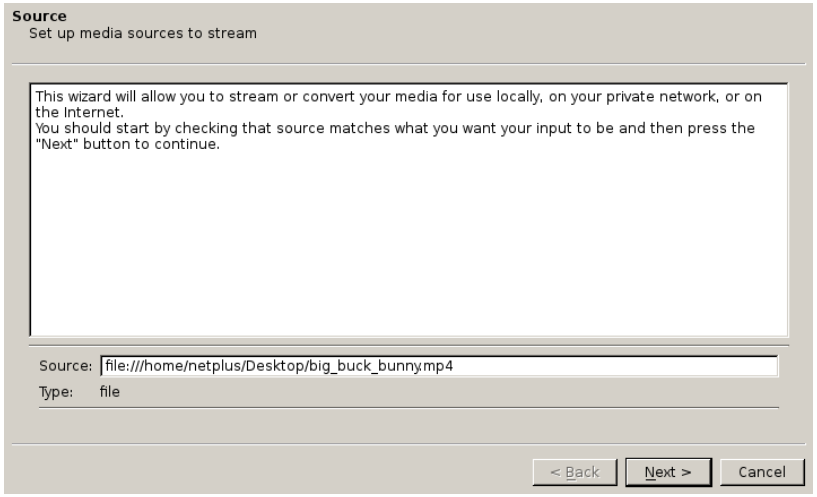
1. Dans le menu **Media**, cliquer sur **Stream …**



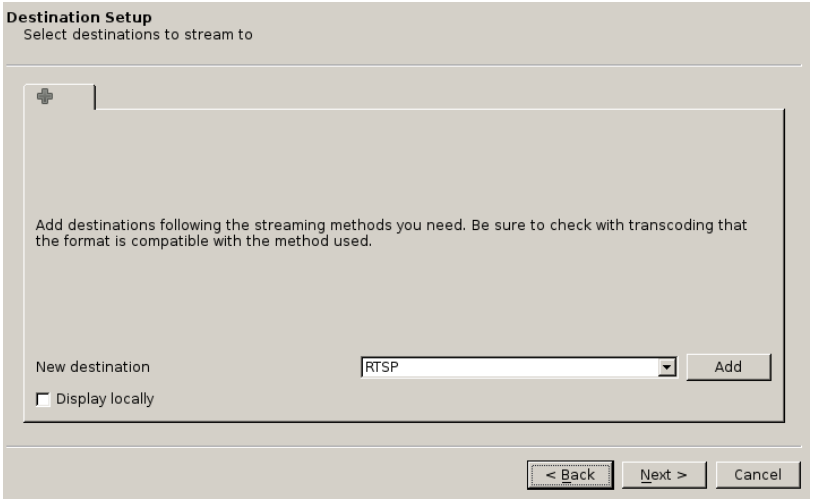
2. Cliquer sur le bouton **Add** afin d'ajouter un média vidéo, avant de cliquer sur le bouton **Stream**



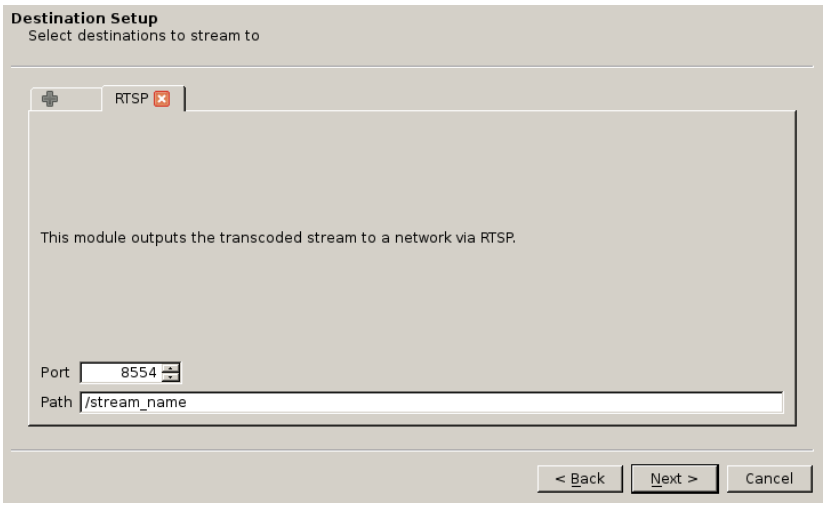
3. Cliquer sur le bouton **Next**



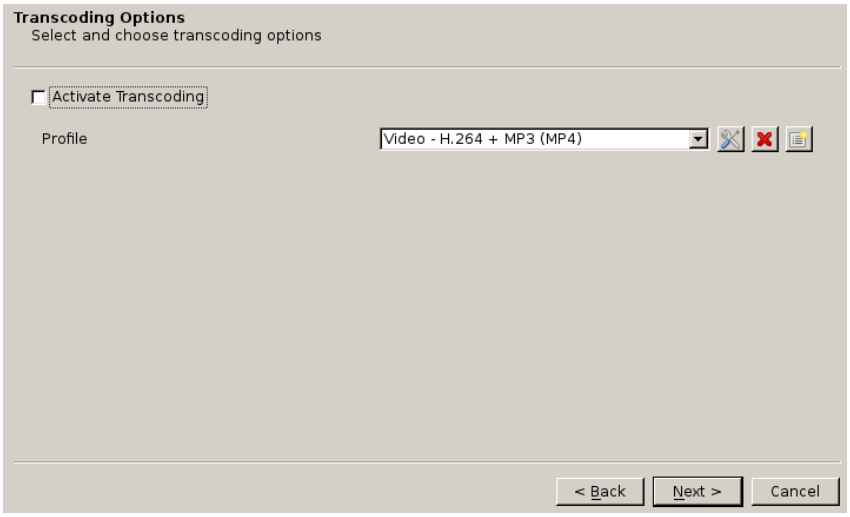
4. Sélectionner **RTSP** dans la liste déroulante puis cliquer sur le bouton **Add**



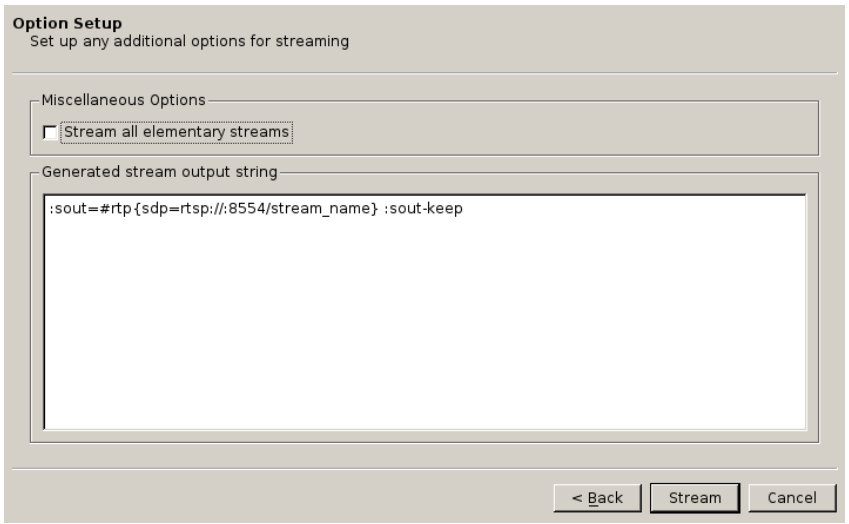
5. Définir un **nom** au stream puis cliquer sur **Next**



6. **Désactiver** le transcodage vidéo avant de cliquer sur le bouton **Next**

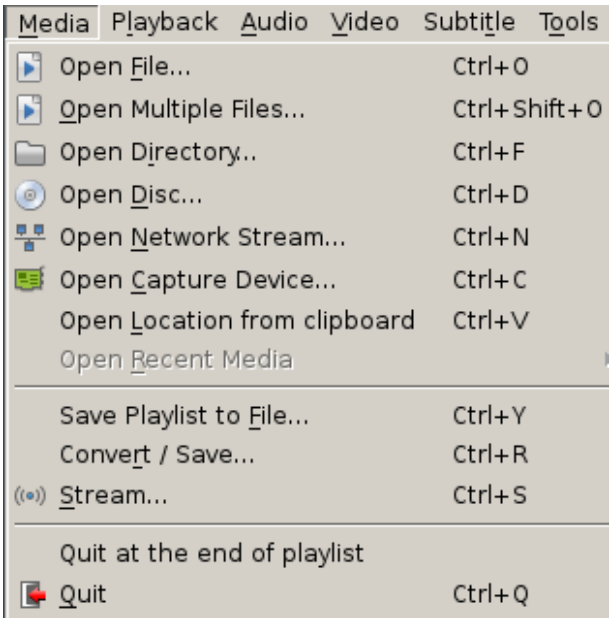


7. Valider en cliquant sur le bouton **Stream**



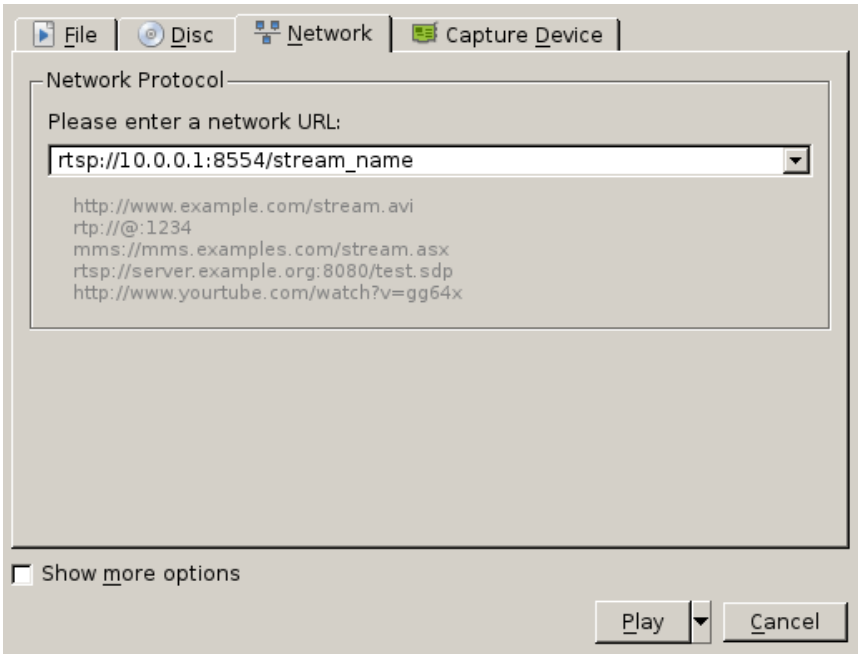
3.2 ABONNEMENT AU FLUX UNICAST

1. Dans le menu **Media**, cliquer sur **Open Network Stream …**



2. Dans le champ de l'URL, renseigner le flux comme suit avant de cliquer sur le bouton **Play** :

`rtsp://`*`ip_address`*`:8554/`*`stream_name`*

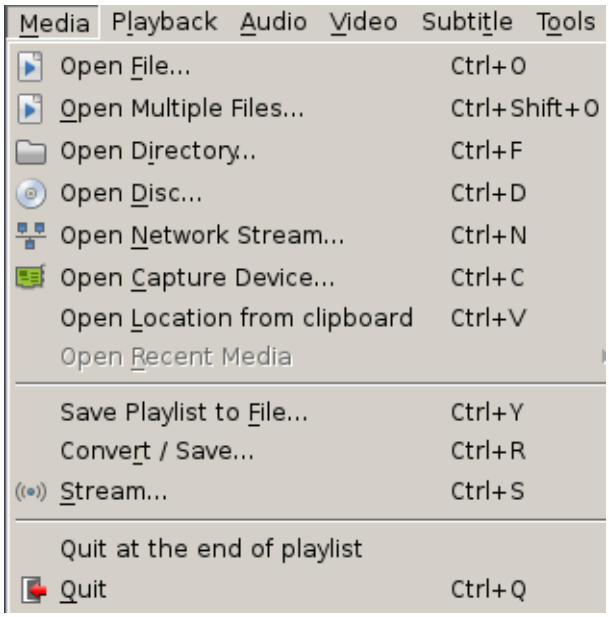


4 FLUX MULTICAST

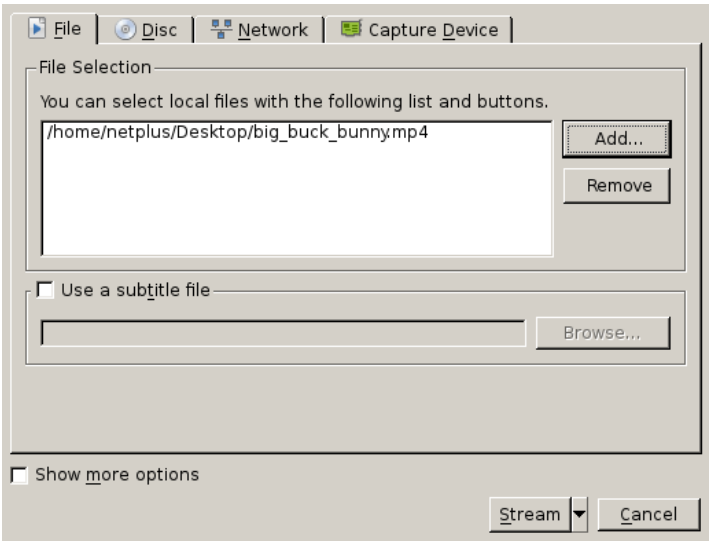
Les flux de type Multicast permettent de simuler la diffusion d'une chaîne de télévision live. Il est piloté par le protocole IGMP.

4.1 DIFFUSION DU FLUX MULTICAST

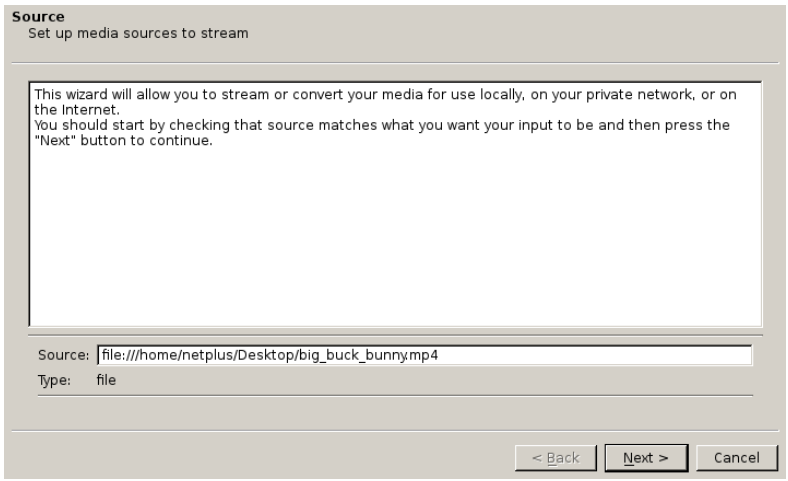
1. Dans le menu **Media**, cliquer sur **Stream …**



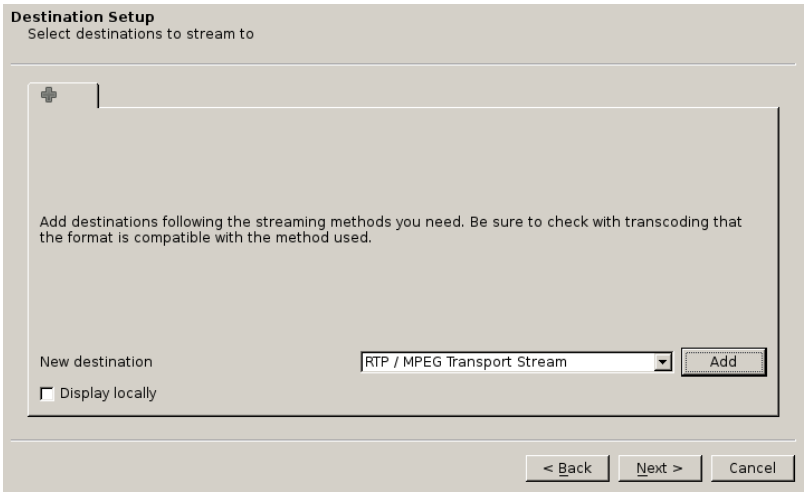
2. Cliquer sur le bouton **Add** afin d'ajouter un média vidéo, avant de cliquer sur le bouton **Stream**



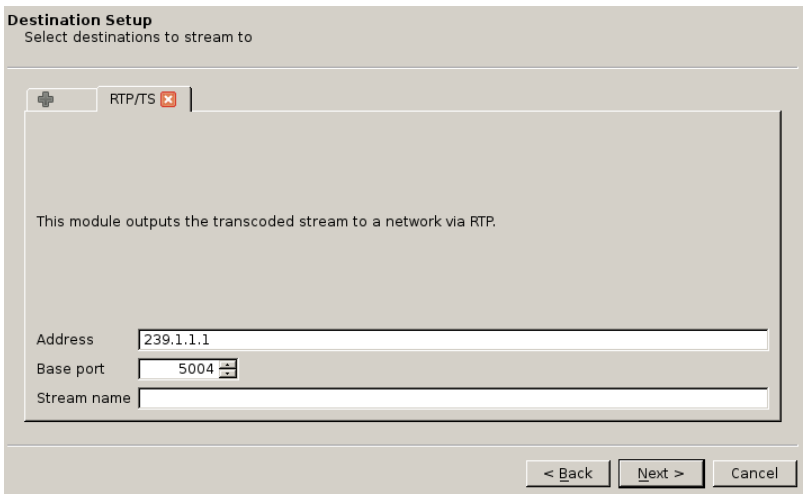
3. Cliquer sur le bouton **Next**



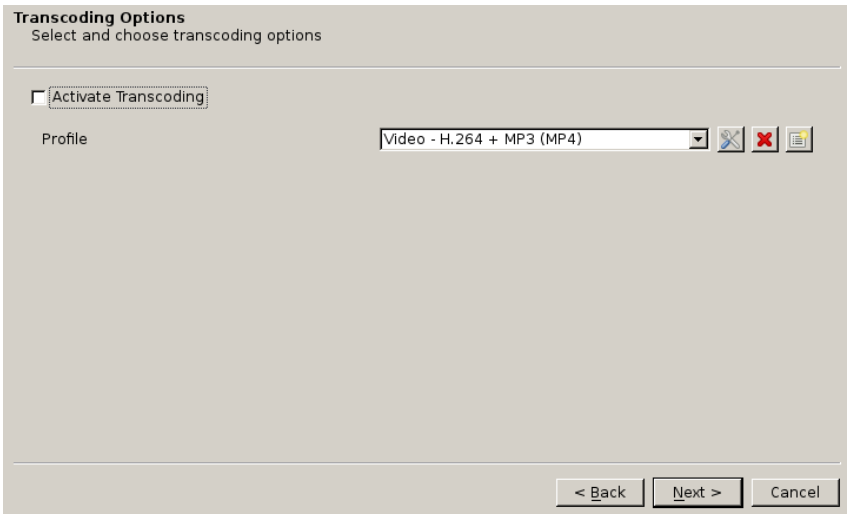
4. Sélectionner **RTP / MPEG Transport Stream** dans la liste déroulante puis cliquer sur le bouton **Add**



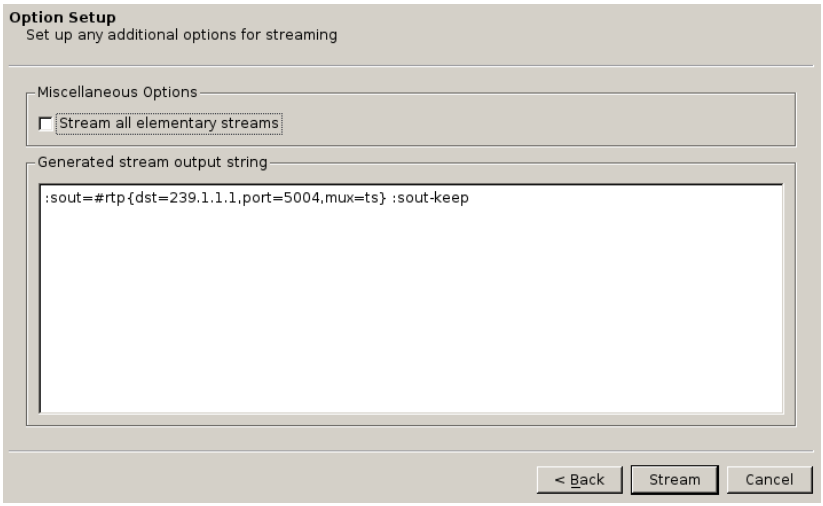
5. Dans le champ **Address**, entrer l'adresse IP du groupe Multicast de diffusion avant de cliquer sur **Next**



6. **Désactiver** le transcodage vidéo avant de cliquer sur le bouton **Next**

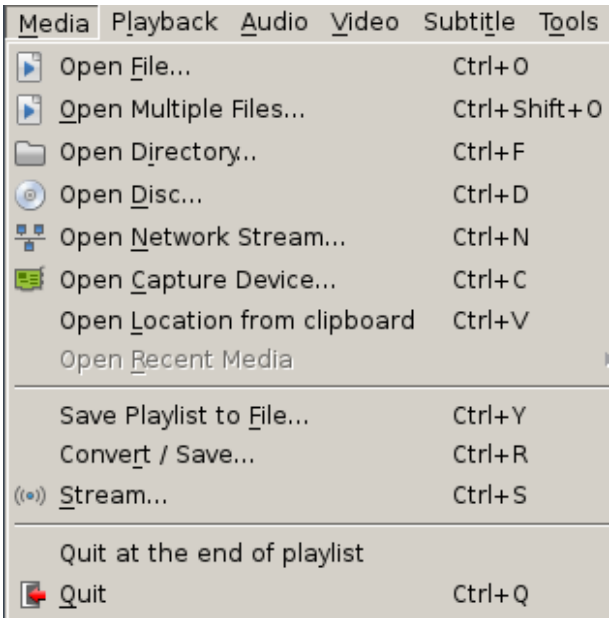


7. Valider en cliquant sur le bouton **Stream**



4.2 ABONNEMENT AU FLUX MULTICAST

1. Dans le menu **Media**, cliquer sur **Open Network Stream …**



2. Dans le champ de l'URL, renseigner le flux comme suit avant de cliquer sur le bouton **Play** :

`rtp://@group_address`