Mémoire de fin d'études
ENS Louis-Lumière

Étude des possibilités d'utilisation d'un réseau sans-fil pour le transport d'un réseau audionumérique

Léo Rossi–Roth

Directeur interne : Jean Rouchouse
Directeur externe : Benoît Quiniou
Rapporteur : Franck Jouanny

Session Juin 2014

Résumé

L'utilisation grandissante depuis quelques années des technologies des réseaux informatiques appliqué à l'audiovisuel d'une part, avec des technologies comme Dante *et* Ravenna*, et des réseaux informatiques sans-fil d'autre part nous pousse à nous interroger sur les possibilités d'utiliser conjointement ces deux aspects pour proposer une nouvelle solution de transfert de l'audio dans les installations professionnelles, offrant de nouvelles possibilités de routage entre les divers appareil audio tout en supprimant la contrainte du câblage.*

Pour cela, ce mémoire étudie le cas d'un réseau utilisant la technologie Ravenna, et discute en particulier de la problématique du transfert d'un signal de synchronisation à travers un réseau sans-fil au regard de différents standards de l'IEEE et de l'AES. En parallèle, des tests sont effectués pour valider ou infirmer les hypothèses émises.

Mots-clés

Réseaux audionumériques, IEEE 802.11, Wi-Fi, Réseaux sans-fil, AES67, Ravenna, IEEE 1588, PTP, AVB, IEEE 802.1AS, gPTP

Abstract

Over the last few years, technology of audio-over-IP has been developed and grown to become commonly used for many sound reinforcement applications, providing less cabling and more flexibility for the routing of audio channels between audio devices, using protocols like Dante *and* Ravenna*. At the same time, performances of Wi-Fi have reached throughput which in theory allow the transmission of a great number of audio channels through a wireless link. This Masters dissertation studies the possibility of using a WLAN to link professional audio devices together, to offer new possibilities of sound installations through an improved flexibility.*

To achieve this result, this Masters dissertation studies the case of a Ravenna network and discusses in particular of the matter of the synchronization through a WLAN, in regards of several standards of the IEEE and the AES. In parallel, tests are run to valid or invalid hypothesis.

Keywords

Audio networking, IEEE 802.11, Wi-Fi, Wireless networking, AES67, Ravenna, IEEE 1588, PTP, AVB, IEEE 802.1AS, gPTP

Remerciements

Je tiens d'abord à remercier tout particulièrement William Leveugle *pour son implication personnelle tout au long de ce projet, ainsi que pour ses conseils et relectures avisés.*

Je voulais ensuite remercier Benoît Quiniou *pour m'avoir aidé à initier ce travail, puis pour l'avoir soutenu tout du long, ainsi que pour m'avoir mis en contact avec les équipes de* Lawo *et d'*ALC NetworX*. Je remercie d'ailleurs Stephan* Türkay *et Andreas* Hildebrand *pour avoir répondu à mes interrogations et m'avoir fourni les licenses JADE.*

Merci aussi à Jean Rouchouse *d'avoir suivi et conseillé ce travail et d'avoir donné quelques coups de pouces bienvenus.*

Je souhaite aussi adresser des remerciements à la société 44.1 et en particulier à Olivier Montagnon *pour avoir mis à ma disposition à plusieurs reprises une interface* Horus*. Je tiens de même à remercier Jean-Nicolas* Valdenaire *et la société KaliSon pour le prêt d'une autre* Horus *au pied levé.*

Merci aussi à Paul Payen de La Garanderie *pour m'avoir fourni les ordinateurs dont j'avais besoin pour mener mes tests.*

Un merci tout particulier à mes parents pour leur implication dans ce projet par leurs relectures appliquées, mais bien plus généralement pour leur soutien infaillible à chacun de mes projets.

Merci à Hanna et Thibaut d'avoir accepté l'intrusion du WiFi dans l'appartement, promis un jour il y aura moins de câbles !

Merci enfin à tous mes camarades de l'école Louis-Lumière, ainsi qu'aux enseignants et au personnel de l'école pour ces trois excellentes années, en espérant que nos chemins puissent se croiser à nouveau.

Table des matières

Liste des acronymes

AES Audio Engineering Society

API Application Programming Interface

ASIO

AVB Audio Video Bridge

BMCA Best Master Clock Algorithm

CIDR Classless Inter-Domain Routing

GM Grandmaster

GPS Global Positioning System

gPTP generalized Precision Time Protocol

IEC Commission électrotechnique internationale

IEEE Institute of Electrical and Electronics Enginners

IP Internet Protocol

ISO Organisation internationale de normalisation

LAN Local Area Network

MAC Media Access Control

MADI Multichannel Audio Digital Interface

MCS Modulation and Coding Scheme

MIMO Multiple Inputs Multiple Outputs (6.7.2, [17])

MLME MAC Sublayer Manager Entity

NTP Network Time Protocol

OSI Open Systems Interconnection

PLL Phase-Locked Loop

PTP Precision Time Protocol

RTP Real-time Transport Protocol

SSID Service Set Identifier

UDP

WAN Wide Area Network

WDM

WLAN Wireless LAN

Introduction

La transmission des signaux audio n'a eu de cesse d'évoluer avec les mutations qu'ont connues tout d'abord l'électronique puis plus récemment l'informatique. Si avec la transmission filaire analogique, deux conducteurs physiques au minimum sont nécessaires pour l'acheminement d'un signal sonore, l'arrivée du numérique a permis le multiplexage des signaux. On peut alors aujourd'hui avec une technologie comme le MADI [1] transférer jusqu'à 64 canaux audio à travers un simple câble coaxial, composé lui aussi de deux conducteurs physiques. Les conséquences directes de ces évolutions sont une réduction significative du coût d'achat, un gain de temps lors des différentes interventions au sein d'une installation, ainsi que l'ouverture à de nouvelles possibilités d'exploitation.

Ces dernières années, le transport de ces signaux a connu une nouvelle mutation majeure avec l'arrivée des technologies des réseaux informatiques dédiées à l'audio numérique. En plus du transport de plusieurs centaines de canaux audio à travers un unique câble, ces technologies amènent une flexibilité de connexions totalement nouvelle. Car là où les technologies numériques traditionnelles se limitent à proposer des liaisons point-à-point unidirectionnelles à travers un câble, l'utilisation d'un réseau informatique permet à tous les appareils audionumériques d'une installation connectés à ce réseau de communiquer directement entre eux. On obtient alors nativement et sans matériel supplémentaire la flexibilité permise autrefois uniquement grâce à l'utilisa-

tion de patchs et le branchement/débranchement d'un nombre conséquent de câbles.

Les gains supplémentaires obtenus par l'utilisation de ces technologies en terme de coût d'achat et de temps d'installation, ainsi que la multiplication des réseaux informatiques, et leur présence quasi-systématique dans les nouvelles installations nous incitent à croire en un développement important et rapide de ces technologies dans un avenir proche.

D'autre part aujourd'hui dans notre société d'information à grande vitesse, la recherche constante d'une mobilité et d'une flexibilité toujours plus grandes conduit à l'essor du développement des technologies de communication sans fil. Même si l'audio sans fil existe depuis les années 1920 et l'apparition de la radiodiffusion, et est aujourd'hui largement développé [1], les solutions proposées dans le monde de l'audiovisuel professionnel restent aujourd'hui bornées à des liaisons unidirectionnelles point-à-point, et ne permettent le passage d'un nombre conséquent de canaux (quelques dizaines) qu'au prix de la multiplication de matériel dédié, dans la limite de la disponibilité des bandes de fréquence allouées à ces activités.

Petit à petit, ce sont les réseaux informatiques aussi qui succombent aujourd'hui au sans-fil, comme on peut le constater à travers le développement fulgurant qu'ont connu les technologies des WLAN [2], communément regroupées sous le terme de « Wi-Fi » [3], ces dernières années. On peut alors imaginer la prochaine étape dans l'évolution de la transmission des signaux audio, si à la flexibilité permise aujourd'hui par l'utilisation des réseaux audionumériques, on ajoute l'extrême légèreté de mise en œuvre et la mobilité d'une liaison sans fil. La question qui se pose alors est la suivante : Est-il possible aujourd'hui de

1. On peut penser notamment à l'omniprésence de la téléphonie mobile dans notre paysage contemporain

2. *Wireless Local Area Network* : Réseau local sans fil

3. On évitera par la suite l'utilisation de ce terme, la dénomination « Wi-Fi » n'étant qu'une appellation commerciale de l'implémentation des différentes normes IEEE 802.11

proposer un système de transport sans fil d'un réseau audio à destination des professionnels de l'audiovisuel à travers l'utilisation d'un WLAN ?

Cette question a déjà été abordée il y a 2 ans, lors du mémoire de fin d'études de l'ENS Louis Lumière de William Leveugle [17] , intitulé *Des possibilités d'évolution vers le sans fil des réseaux audionumérique pour la sonorisation – cas des Wireless LAN*. Le but de notre travail ici sera donc de continuer les travaux entrepris par William Leveugle et d'essayer d'accomplir un pas supplémentaire vers la réussite de cette entreprise. Le préambule de ce mémoire sera d'ailleurs consacré à faire le lien entre ces deux travaux, en explicitant notamment les résultats des précédentes recherches servant de point de départ aux nôtres et en justifiant les différences d'approche qui pourront apparaître entre ces deux travaux.

Le développement de ce mémoire suivra ensuite l'angle d'attaque suivant : afin de pouvoir tester en pratique la transmission effective ou non de signaux audio à travers un WLAN, il a été décidé de construire une plate-forme informatique d'expérimentation, de manière à reproduire une situation d'installation classique. La première partie de ce mémoire s'attellera à la description de cette plate-forme et de ses différents composants, matériels et logiciels, ainsi qu'à la justification du choix de ces différents composants. Elle s'achèvera sur une série de tests préliminaires, en filaire d'abord, puis en testant la liaison sans fil de manière « naïve ». Ces tests nous permettront de conclure sur les premiers problèmes rencontrés. Parmi ces problèmes, on distinguera en particulier celui de la transmission d'un signal de synchronisation à travers un réseau sans fil, qui s'était déjà révélé comme étant le premier problème à traiter lors du mémoire de William Leveugle. La seconde partie de ce mémoire proposera donc de tenter de résoudre ce problème. Elle commencera par exposer le fonctionnement théorique de la synchronisation à travers un réseau informatique, avant de proposer les différents outils à notre disposition pour adapter un tel

signal à un réseau sans fil. Enfin on confrontera ces différents outils à leur mise en œuvre pratique. La troisième et dernière partie de ce mémoire tentera d'une part de s'affranchir des problèmes de la transmission du signal de synchronisation pour confronter les autres paramètres de la transmission aux conditions du sans fil, et conclura sur les différentes options à envisager afin de continuer l'exploration du domaine des WLAN appliqués aux réseaux audionumériques.

Avant-propos

Ce mémoire se proposant d'être dans la continuité des travaux initiés sur le sujet par William Leveugle en 2012, nous nous proposons dans ce préambule de faire le lien entre les résultats précédemment obtenus et le développement de ce mémoire.

Cadre de ce mémoire

Avant toute chose, il nous paraît nécessaire de préciser quelque peu notre pensée lorsque nous parlons de transmission de canaux audio à travers un WLAN. En effet aujourd'hui, n'importe quel internaute connecté à travers une liaison Wi-Fi peut écouter depuis plusieurs années déjà la radio sur Internet ou de la musique sur un site de *streaming* : il reçoit bien alors des données audio à travers un WLAN. Il peut aussi émettre des données sonores sur le réseau en passant par exemple un appel à travers un logiciel de visioconférence. On peut aussi remarquer ces dernières années le développement d'applications de streaming local pour le grand public, permettant de transférer du son, éventuellement à travers un réseau sans fil, d'un appareil domestique à l'autre (technologie AirPlay d'Apple ou appareils compatibles DLNA).

On sait donc transférer du son à travers un WLAN, et les applications existent déjà partout autour de nous. Quel est donc alors le but de ce mémoire ? Et bien si toutes les applications citées précédemment fonctionnent effectivement aujourd'hui, aucune ne satisfait aux exigences actuelles des ap-

plications audiovisuelles professionnelles. En effet, si l'un ou l'autre des critères de la liste suivante peut parfois être respecté, aucune ne les respecte tous. Pour pouvoir être utilisé dans une installation et des applications audiovisuelles professionnelles aujourd'hui, une liaison numérique doit permettre :

- Le passage en temps réel de plusieurs canaux d'audio-numérique non compressé
- Une latence faible
- Une synchronisation de tous les appareils sur une même horloge maître, avec une gigue la plus réduite possible
- Une gestion des erreurs de transmission

Si selon les applications, les valeurs de ces paramètres peuvent être différentes, on pourra néanmoins se référer au cahier des charges établi par William Leveugle pour son mémoire ([17], 10.6). L'objectif de ce mémoire est donc de permettre l'emploi d'une liaison audionumérique ayant comme support un WLAN, et satisfaisant à tous ces paramètres. Ainsi nous aurions une solution susceptible d'être utilisée dans les domaines de l'audiovisuel professionnel.

Par ailleurs ce cahier des charges avait été établi afin de pouvoir répondre aux applications du domaine de la sonorisation live, cadre privilégié du mémoire précité. Même si nous gardons l'objectif de pouvoir appliquer nos résultats à la sonorisation, en particulier car c'est l'une des applications les plus gourmandes en terme de nombres de canaux audio utilisés et des plus sensibles à la latence, nous ne nous fixerons pas de cadre *a priori*, et nous pourrons, en fonction des résultats, envisager dans un premier temps de privilégier l'utilisation d'une transmission audio numérique sur WLAN pour certaines applications professionnelles en particulier.

Résultats acquis

Le mémoire de William Leveugle comportait une importante partie consacrée à l'explication des fondamentaux des réseaux informatiques, et des réseaux sans fil en particulier. La suite de ce mémoire s'appuie sur ces connaissances, et nous n'hésiterons pas à y référer directement si besoin est, mais aussi nous serons amenés à compléter certains points, en partant des développements du précédent mémoire.

Les tests menés sur l'installation sans fil précédente ont déjà permis de valider plusieurs points permettant à un WLAN de pouvoir prétendre être une solution viable pour le transfert de données audionumériques. En particulier la distance de transmission permise (jusqu'à 1,5km avec le matériel utilisé alors, *cf* [17], 12.1.2) est largement suffisante pour couvrir la grande majorité des applications, et offre ainsi même des performances dont seules les liaisons par fibre optique sont aujourd'hui capables. Nous ne nous attacherons donc pas forcément à confronter la distance maximale d'utilisation de notre plateforme à des applications concrètes, étant entendu que l'utilisation d'antennes directives adéquates serait la solution si les performances n'étaient pas à la hauteur.

Un des autres problèmes soulevé par William Leveugle est la forte utilisation de la bande ISM dite des 2.4 GHz par un nombre toujours plus grand d'applications dans notre environnement (présence d'autre WLAN, appareil Bluetooth, transmission des caméras de vidéo-surveillance, etc). Il peut alors être difficile d'assurer une bonne qualité de transmission du signal sans fil si cette bande est localement trop polluée. Pour contourner ce problème, William Leveugle proposait l'utilisation de la bande des 5 GHz, encore assez peu utilisée aujourd'hui. C'est ce que nous nous attacherons à reproduire lors de nos expériences.

Choix du protocole et cadre normatif

Les travaux précédemment menés présentaient un comparatif des différents réseaux audionumériques, avant de s'attarder sur le cas du réseau Dante, utilisé en particulier dans la partie pratique. Nous avons choisi pour ce mémoire de ne pas utiliser Dante, mais d'utiliser à la place le protocole Ravenna, et ceci pour plusieurs raisons que nous allons expliquer ici. À cette fin nous allons passer rapidement sur quelques caractéristiques de ce protocole, sur lesquelles nous reviendrons lors d'une présentation plus détaillée dans la première partie (section 1.3).

Tout d'abord, il faut remarquer que les protocoles Ravenna et Dante se ressemblent beaucoup. Ils utilisent tous les deux un réseau IP pour le transport des informations, le protocole RTP pour la transmission des données audio, et les différents appareils du réseau sont synchronisés à travers le protocole PTP.

Quels avantages tirons-nous alors de l'utilisation du protocole Ravenna ? En premier lieu, contrairement à l'implémentation propriétaire de Dante, Ravenna est un protocole ouvert, et on peut donc avoir accès à toutes ses spécifications, rendant la recherche d'informations en vue d'expérimentation et d'explications plus aisée, voire dans certains cas, tout simplement possible. De plus, Benoît Quiniou, directeur externe de ce mémoire est employé de la société Lawo, maison mère d'ALC NetworX qui a créé et maintient Ravenna. À ce titre il nous a permis d'entrer en contact directement avec les développeurs du protocole, facilitant d'autant la recherche d'informations parfois impossibles à obtenir autrement. Il est également à noter que William Leveugle a fait référence à ce protocole dans son mémoire, mais n'avait à l'époque pas pu l'utiliser lors de ses expériences car aucun produit commercialisé ne l'implémentait alors.

Enfin, l'AES a publié en juin 2013 le standard AES67 [2], inti-

tulé *High-performance streaming audio-over-IP interoperability*[4], définissant un cadre standard pour la transmission de données audionumériques sur un réseau IP dans le cadre d'applications professionnelles, afin de permettre à terme une cohabitation et une compatibilité de différents protocoles. RAVENNA, au contraire de DANTE, est d'ores et déjà compatible avec ce standard, ce qui signifie qu'une solution pour RAVENNA aujourd'hui, sera de fait compatible avec n'importe quel protocole futur respectant ce standard, ce qui permettra de proposer à terme aux professionnels de l'audiovisuel une solution la plus souple possible afin de pouvoir répondre à une plus grande diversité d'applications. Nous aurons l'occasion d'aborder plus en détail certains points de l'AES67 dans la suite de ce mémoire, mais nous pouvons dès à présent noter que même si toutes les solutions commercialisées aujourd'hui utilisent uniquement l'Ethernet comme support de transmission des données audionumériques, le standard n'empêche pas *a priori* l'utilisation d'un autre support de transport. Il suffit que celui-ci soit compatible avec le protocole IP (*in* [2], section 6, p.13). L'AES67 précise juste : *« It is assumed that best practices in transport of IP over the network technologies in question are employed.*[5] *»*

4. Interopérabilité des transmissions audio sur IP à haute performance

5. Il est présumé que les meilleures pratiques de transport de l'IP à travers les technologies réseau en question sont employées

Chapitre 1

Mise en place d'une plate-forme d'expérimentation et tests préliminaires

1.1 Objectifs de la plate-forme d'expérimentation

Les recherches préliminaires effectuées pour ce mémoire et les résultats déjà obtenus par William Leveugle nous permettent d'espérer que le transport d'un réseau audionumérique à travers un WLAN sera possible à plus ou moins long terme. Idéalement, nous souhaiterions que la conclusion de ce mémoire soit une solution clé en main accomplissant cette tâche à proposer à des professionnels de l'audiovisuel. Et c'est dans cette optique que nous avons choisi de commencer ces recherches sous un angle pratique, en mettant en œuvre un réseau audionumérique qui servira de support à nos futures expériences.

Afin de pouvoir transposer les résultats à des situations concrètes,

on cherchera donc à rendre le réseau audionumérique ainsi constitué le plus généraliste possible, tout en restant dans les contraintes budgétaires inhérentes à ce mémoire.

1.2 Structure retenue pour le réseau

La structure retenue pour le réseau est la suivante : deux ordinateurs personnels [1] chacun relié par un câble Ethernet à un routeur sans-fil, soit la structure la plus simple permettant théoriquement une liaison audionumérique bidirectionnelle à travers un WLAN. Les deux ordinateurs seront équipés de logiciels permettant la diffusion et la réception de flux audio à travers leur interface réseau. Ils s'apparenteront à n'importe quel appareil audio prévu pour fonctionner sur un réseau audionumérique.

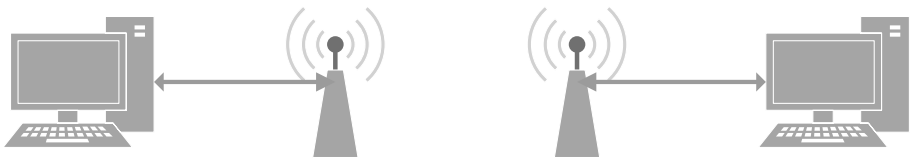

FIGURE 1.1 – Topologie de la plateforme d'expérimentation

Reste le choix d'utiliser deux routeurs sans-fil externes, et non des interfaces sans-fil intégrées aux ordinateurs. Cette option résulte de notre volonté de pouvoir adapter aussi simplement que possible ce réseau à une situation concrète. Or aujourd'hui, et quel que soit le protocole utilisé, les appareils commercialisés pouvant se connecter à un réseau audionumérique disposent

1. on utilise ici la dénomination d'ordinateur personnel, par opposition à un produit dédié pouvant se connecter à un réseau audionumérique, comme par exemple une console de mixage ou un boîtier de scène, et disposant à ce titre d'un système embarqué répondant aussi à la définition d'ordinateur

exclusivement de connexions filaires par câble Ethernet. La solution de l'utilisation de routeurs sans-fil externes permet donc d'envisager d'ores et déjà la connexion d'appareils dédiés à l'audionumérique à notre réseau, et non uniquement d'ordinateurs personnels.

1.3 Présentation de Ravenna

Comme justifié dans l'avant-propos, notre choix de la technologie pour transférer l'audio à travers notre réseau informatique s'est porté sur Ravenna. L'implémentation de Ravenna est ouverte et s'appuie sur différents protocoles et technologies standardisés. Toutes ces technologies ont en commun l'utilisation de l'IP comme support de transmission, et sont donc toutes *a priori* compatibles avec un WLAN IEEE 802.11. D'ailleurs le document officiel de présentation de Ravenna [21] précise : *« IP is in general infrastructure-agnostic*[2] *and can be used on virtually any network technology and topology »*[3]. En particulier les protocoles qui nous intéresseront par la suite sont le RTP pour la transmission des données audio à proprement parler, et le PTP pour la synchronisation des différents éléments du réseau. On pourra se référer à la présentation plus détaillée faite par William Leveugle ([17], 5.2.3). Le RTP étant un protocole très versatile pouvant servir de transport à de la vidéo ou à de l'audio, compressé ou non, le formatage des données utilisé par Ravenna en fonction entre autre de la fréquence d'échantillonnage des données, de la profondeur de codage des données et du nombre de canaux à transférer est décrit dans [2], section 7, *Encoding and streaming*.

2. il faut comprendre ici *agnostic* commme *« Denoting or relating to* hardware *or* software *that is compatible with many types of platform or operating system »*, (Oxford Dictionnary), c'est-à-dire qui peut fonctionner sur un nombre important de supports différents

3. L'IP est en général agnostique vis-à-vis des infrastructures et peut être utilisé sur quasiment n'importe quelle technologie et topologie de réseau

Concernant le PTP, une étude détaillée de ce protocole sera faite dans la deuxième partie de ce mémoire (sous-section 2.2.1), mais on peut d'ores et déjà noter que, contrairement à Dante qui utilise la première version du PTP définie par la norme IEEE 1588-2002, Ravenna utilise la deuxième version du protocole, définie par l'IEEE 1588-2008 [6], à laquelle il est généralement fait référence sous le terme PTPv2.

1.4 Routeur sans-fil

1.4.1 Cahier des charges

Les conclusions du mémoire de [17] nous ont donné les bases des caractéristiques du routeur à choisir. Celui-ci doit être compatible avec la bande de fréquence des 5 GHz, et pour offrir un débit suffisant, être conforme aux spécifications IEEE 802.11n ou IEEE 802.11ac.

Concernant la partie commutateur Ethernet du routeur, on privilégiera un routeur compatible avec le Gigabit Ethernet. Même si, contrairement à Dante, Ravenna ne contraint pas à l'utilisation exclusive de commutateurs Gigabit, ils sont suffisamment répandus aujourd'hui pour que cette contrainte ne soit pas très restrictive, et ce choix nous permettra à terme le passage d'un plus grand nombre de canaux et des performances globalement supérieures. De plus, si la transmission de données Ravenna à travers un WLAN se voit couronner de succès, il serait intéressant de pouvoir adapter la solution à Dante, et ce type de matériel pourra nous faciliter cette transition.

L'autre contrainte qui a été définie pour le choix de ce routeur est sa compatibilité OpenWrt. OpenWrt est un projet qui a pour mission de proposer un micrologiciel alternatif et libre, basé sur un noyau GNU/Linux, visant à remplacer le micrologiciel fourni par défaut avec les routeurs sans-fil vendus dans le commerce. Nous aurons l'occasion de revenir sur certains détails

d'OpenWrt au cours du développement de ce mémoire, mais notre volonté d'acquérir un tel routeur résulte d'abord des possibilités offertes par le modèle *open source*, qui nous donne un accès total au code source du micrologiciel, ce qui nous permettra d'étudier facilement et en détail le comportement de celui-ci. De plus, OpenWrt est pensé pour être modulaire, et ainsi permettre de rajouter des fonctionnalités logicielles directement à l'intérieur du routeur. On peut alors imaginer, une fois les solutions à nos problèmes trouvées, les implémenter directement à l'intérieur d'un routeur OpenWrt, et ainsi fournir une solution clé en main contenue dans un unique produit.

Malgré les nombreux avantages évidents à utiliser un routeur compatible OpenWrt, ces modèles sont peu nombreux, et l'installation d'OpenWrt peut parfois obliger à un peu de bricolage et de soudure, ce que nous souhaiterions éviter en vue d'une viabilité à long terme. Cette contrainte sera donc l'une des plus fortes et la liste des routeurs compatibles avec OpenWrt le point de départ de notre quête du routeur idéal.

1.4.2 Caractéristiques du modèle retenu

Ce cahier des charges associé à la contrainte budgétaire du mémoire nous a permis de déterminer deux candidats potentiels : le modèle WNDR3700 du constructeur Netgear, et le modèle TL-WDR4300 du constructeur TP-Link. La facilité d'installation et la meilleure compatibilité de ce second modèle avec OpenWrt a fait pencher la balance en sa faveur. Nous allons donc ici brièvement présenter ses caractéristiques.

Caractéristiques WLAN

Le TL-WDR4300 est équipé de 2 circuits intégrés distincts afin de pouvoir gérer simultanément un WLAN dans la bande des 2.4 GHz (circuit Atheros AR9341) et un WLAN dans la bande des 5 GHz (circuit Atheros

AR9580). C'est évidemment cette dernière fonctionnalité qui nous intéressera le plus. Le constructeur annonce des débits pouvant aller jusqu'à 450 Mbps, à travers un MIMO 3x3, mettant en œuvre 3 antennes externes ([17], 6.7.2). Les antennes 5 GHz se connectant au routeur via des connecteurs RP-SMA, il sera de plus facile d'éventuellement les remplacer par des antennes directives ultérieurement.

Caractéristiques Ethernet

Les ports Ethernet du routeur sont séparés en deux catégories : un port Ethernet Gigabit *WAN*, et quatre ports Ethernet Gigabit *LAN*. Le port *WAN* est prévu pour recevoir une connexion Internet (depuis un modem ADSL par exemple), tandis que les quatre ports *LAN* sont destinés à la connexion des appareils informatiques du réseau. Si ces ports sont identiques physiquement et techniquement, ils permettent en outre l'application de règles de routage et de filtrage spécifiques selon l'utilisation pratique des ports.

Caractéristiques diverses

En plus des connectiques dédiées au réseau, ce routeur dispose de deux ports USB 2.0, autorisant la connexion et le partage d'un support de stockage ou d'une imprimante sur le réseau. Bien que ces fonctions ne concernent *a priori* pas notre problématique et ne seront pas utilisées dans ce mémoire, elles peuvent ouvrir à terme d'autres possibilités, la documentation d'OpenWrt proposant par exemple de connecter une carte son sur un routeur, possibilité qui pourrait être utilisée dans notre cas à des fins de monitoring des flux audio circulant sur le réseau par exemple.

FIGURE 1.2 – Connecteurs et antennes du TP-Link WDR4300 *(source : tp-link.com)*

1.5 Solutions Ravenna logicielles

Pour des raisons de coût et de facilité d'exploitation, il a d'abord été décidé de baser notre plate-forme de tests sur l'utilisation d'ordinateurs exploitant RAVENNA à travers leur interface réseau par le biais de logiciel dédié. Les solutions utilisées sont d'une part la carte son RAVENNA virtuelle (abrégée par la suite RVSC, pour RAVENNA virtual sound card) développée par ALC NETWORX, et d'autre part le logiciel JADE développé par LAWO. Tandis que RVSC est téléchargeable gratuitement sur le site de RAVENNA après enregistrement, LAWO nous a fourni à titre gracieux deux licences d'un an pour JADE afin de pouvoir mener nos tests.

1.5.1 RVSC

RVSC est un pilote RAVENNA développé pour MICROSOFT WINDOWS 7. C'est un pilote audio utilisant la technologie WDM pour s'intégrer au système d'exploitation. Après installation de RVSC, le système d'exploitation a accès à une nouvelle interface audio, nous permettant d'utiliser RAVENNA de manière transparente comme n'importe quelle interface audio. L'utilisation de WDM permet à l'interface audio d'être utilisée aussi bien comme interface par défaut du système que d'être dédiée à un logiciel spécialisé, comme la station audionumérique REAPER de COCKOS, qui sera utilisée à plusieurs reprises par la suite. La configuration de RVSC s'effectue à travers 2 interfaces distinctes : une interface logicielle accessible depuis la barre des icônes WINDOWS permet

de régler les options de synchronisation, tandis que l'état de fonctionnement de l'interface et la gestion des flux audio se fait à travers une interface Web accessible en se connectant au port 8080 de l'ordinateur hôte. Pour plus d'informations sur le fonctionnement de RVSC, on pourra se référer à l'Annexe A.

RVSC présente en outre une particularité quelque peu limitante pour son utilisation : Il n'est compatible qu'avec une interface Ethernet utilisant un contrôleur réseau Intel 82574L. Cette limitation est particulièrement rédhibitoire en ce qui concerne l'utilisation d'un ordinateur portable. Si le bon contrôleur n'est pas présent directement sur la carte mère de l'ordinateur, il sera quasiment impossible d'utiliser RVSC. En revanche, il est assez facile de l'utiliser avec un ordinateur de bureau, ce contrôleur réseau se trouvant entre autre sur la carte réseau PCI-Express Intel PRO1000 CT, avec port Ethernet Gigabit. C'est cette solution que nous avons choisie et avons donc acquis deux cartes de ce modèle.

Figure 1.3 – Carte PCI-Express Intel PRO1000 CT *(source : ldlc.com)*

Même si le choix de ce contrôleur réseau se justifie par la gestion matérielle des instructions PTPv2 qui permet d'espérer de meilleures performances de synchronisation qu'avec un contrôleur réseau standard, on pourra regretter qu'il limite drastiquement l'utilisation de RVSC, contrairement à son équivalent pour Dante, la DVS ou Dante *Virtual Soundcard*, qui permet à n'importe quel ordinateur disposant d'un port Ethernet de se connecter et d'échanger avec un réseau Dante. Cependant la documentation de RVSC

laisse entendre que le support d'autres contrôleurs réseau pourrait voir le jour avec de nouvelles versions, RVSC étant un produit encore jeune dont la première version publique date de septembre 2013.

1.5.2 JADE

Une autre solution proposée par Benoît Quiniou pour pouvoir communiquer sur un réseau Ravenna à partir d'un ordinateur personnel, est l'utilisation du logiciel JADE commercialisé par la société Lawo. JADE est un logiciel faisant office de grille de routage entre toutes les sources et interfaces audio connectées à l'ordinateur. Il est en outre possible de régler les niveaux des différents envois et d'intégrer à la grille des traitements audio directement intégrés au logiciel ou sous forme de greffons logiciel au format VST. Il fonctionne sous Windows. Il a été au départ pensé comme un outil à destination des journalistes radio (d'ailleurs JADE signifie *Journalist Audio Distribution Engine*[4]), en permettant d'enregistrer différentes configurations de routage et de les rappeler au besoin ultérieurement, par exemple en enregistrant une configuration pour l'enregistrement d'un appel téléphonique, une autre pour l'enregistrement des micros du studio et une pour le montage d'un sujet, qui peuvent être rappelées d'un simple clic.

La fonction qui nous intéressera en particulier ici est la capacité de JADE à créer ou recevoir des flux Ravenna. Ces flux apparaissent alors comme une interface supplémentaire, et on peut les router à souhait vers une application audio ou directement dans une interface audio. Contrairement à RVSC, JADE est compatible avec n'importe quel contrôleur réseau. Il est même possible de lui indiquer d'utiliser directement l'interface sans fil IEEE 802.11 d'un ordinateur. Même si cette fonctionnalité ne nous sera pas utile dans un premier temps où on souhaite pouvoir assimiler l'ordinateur à un appareil

4. Moteur de distribution audio pour journaliste

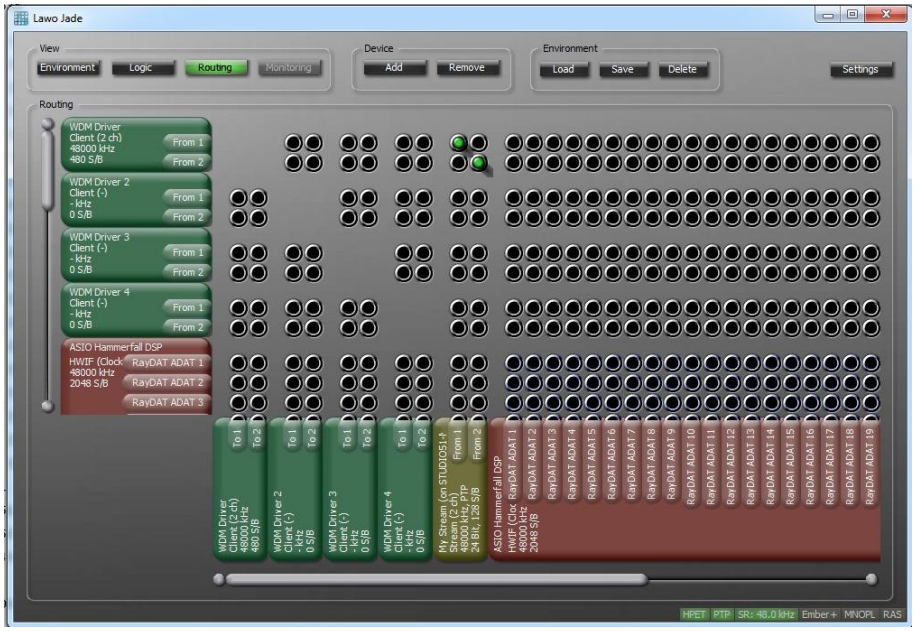

FIGURE 1.4 – Matrice de routage de JADE

RAVENNA standard, elle pourra être une intéressante ouverture à explorer dans un second temps.

1.6 Le problème du *grandmaster* PTP

Sans entrer tout de suite dans les détails du protocole PTP, nous allons ici expliquer quelques-uns de ses aspects afin d'exposer le premier problème auquel nous nous sommes confrontés lors de la mise en place de notre plate-forme d'expérimentation. Tout d'abord revenons sur le principe général de la synchronisation d'un ensemble d'appareils audionumériques : Il s'agit d'acheminer vers chaque appareil un signal d'horloge identique, chaque appareil synchronisant alors son horloge interne à ce signal, typiquement via une boucle à verrouillage de phase, ou PLL [5]. Ce mécanisme assure alors que tous les appareils traitent les données exactement à la même fréquence, et garantit une transmission sans erreur des données tout au long de la chaîne de trai-

5. *Phase-locked loop*

tement audionumérique. L'appareil délivrant le signal d'horloge d'origine est alors désigné comme étant le *grandmaster*.

Dans le cas d'une synchronisation classique, via un signal Wordclock transmis à travers des câbles coaxiaux par exemple, il n'y a pas de doute concernant l'identité de *grandmaster* : chaque appareil pouvant se synchroniser dispose alors d'un port d'entrée (souvent noté *In*) pour recevoir le signal de synchronisation et d'un ou plusieurs ports de sortie (*Out*), pour redistribuer le signal de synchronisation. L'ensemble peut être vu comme une pyramide avec à son sommet l'unique appareil ne recevant aucun signal sur son entrée, et distribuant le signal de son horloge interne, qui est donc le *grandmaster*.

Dans le cas du PTP, les choses sont un peu plus complexes. En effet, la synchronisation s'effectuant à travers un réseau IP où tous les appareils peuvent communiquer de manière bidirectionnelle, il n'y a plus d'organisation hiérarchique *a priori*. Afin de déterminer le *grandmaster* du réseau, le protocole PTP a donc développé un algorithme, nommé BMCA (pour Best Master Clock Algorithm). Concrètement, chaque appareil se synchronisant à travers un réseau PTP embarque des informations à propos de son horloge (en particulier à propos de sa précision, à travers l'information de clockClass), et ces informations sont comparées deux à deux par les différents appareils afin de déterminer lequel est le mieux disposé à assurer le rôle de *grandmaster* PTP.

Cependant, si tous les appareils peuvent participer au BMCA, certains peuvent être écartés et ne pas remplir le rôle de *grandmaster*, en étant configurés avec un clockClass égal à 255, si les spécifications de leur horloge ne sont pas assez bonnes. C'est malheureusement le cas des deux solutions logicielles précédemment décrites. En effet, les horloges internes des ordinateurs ne sont pas suffisamment précises pour remplir ce rôle, comme nous l'a précisé Andreas Hildebrand, Senior Product Manager chez ALC NetworX, et développeur de Ravenna : *« The RVSC is not designed to operate as GM*

due to the inaccuracy of the PC-internal oscillator circuitry. » [6] Et sans *grandmaster* détecté sur le réseau, les appareils RAVENNA sont dans l'impossibilité d'émettre ou de recevoir des flux audio. Nous sommes donc dans une impasse avec notre solution se voulant entièrement logicielle et ne faisant intervenir aucun produit dédié à RAVENNA. Une première solution nous a cependant été proposée par Andreas HILDEBRAND. En effet il est possible d'autoriser exceptionnellement RVSC à être *grandmaster* PTP. Cependant, il nous avertit : *« For testing purposes, you may enable the GM functionality through a registry setting. However, be aware that you may experience glitches or – depending on other participating devices – no satisfactory operation at all. »*

Ceci est donc une première solution que nous ne manquerons pas d'utiliser. Cependant, afin de mener des tests plus conformes à la réalité, il serait intéressant de pouvoir synchroniser notre réseau sur un appareil pouvant offrir de meilleures performances de synchronisation. À cette fin, nous avons contacté la société 44.1, en particulier Olivier MONTAGNON, responsable de la distribution des produits de la société MERGING, qui a accepté de mettre à notre disposition une interface audio HORUS, utilisant RAVENNA.

Ainsi même si notre première intention était d'utiliser uniquement une solution logicielle, cette configuration ne semble pas viable pour assurer des transmissions fiables sur le réseau. Nous décidons donc d'abandonner notre solution initiale et d'intégrer à nos tests un matériel dédié optimisé pour l'utilisation de RAVENNA, qui intègre une horloge interne plus fiable que celle des ordinateurs. Le deuxième ordinateur de notre configuration initiale sera donc remplacé par l'HORUS, ou éventuellement viendra en complément de celle-ci, afin de profiter de la synchronisation de l'interface dédiée même dans le cadre d'un test de liaison entre deux solutions logicielles.

6. RVSC n'est pas conçu pour opérer comme *grandmaster*, à cause de l'imprécision des circuits d'oscillateur internes aux ordinateurs.

FIGURE 1.5 – Nouvelle topologie du réseau de test

1.7 L'interface Merging Horus

1.7.1 Présentation

L'HORUS de MERGING est une interface audio permettant de faire le lien entre un réseau RAVENNA et différentes technologies audio, analogiques et numériques. Le modèle de base propose 24 canaux AES/EBU en entrée et en sortie, une entrée/sortie MADI (coaxiale ou optique) et deux ports RJ45 pour la connexion au réseau RAVENNA [7]. Il est aussi équipé d'une entrée/sortie Wordclock, d'une entrée/sortie Timecode et d'une entrée pour se synchroniser sur une référence vidéo.

FIGURE 1.6 – Interface MERGING HORUS *(source : merging.com)*

En plus de cette connectique standard, ses capacités peuvent être étendues au moyen d'ajout de cartes d'options. L'HORUS dispose à cet effet de 6 emplacements pouvant chacun accueillir une carte d'entrée comportant 8 convertisseurs analogique/numérique avec préamplificateur micro, ou une carte

7. le second port est présent afin de proposer une redondance de la connexion, même si cette fonctionnalité n'a pas été encore implémentée dans le micrologiciel actuel

de sortie avec 8 convertisseurs numérique/analogique. En outre l'Horus dispose d'un emplacement pour une carte d'entrée/sortie MADI supplémentaire, sur connectique coaxiale ou optique comme la connexion MADI primaire.

Le contrôle et la configuration de l'interface peuvent se faire directement via l'écran tactile intégré, ou via une interface Web accessible depuis n'importe quel ordinateur connecté au même réseau que l'Horus. L'interface Web est disponible en deux modes : un mode normal reproduisant à quelques détails près l'affichage de l'interface native et un mode avancé. C'est par ce mode avancé que nous pourrons configurer les différents flux Ravenna reçus et émis par l'Horus. On peut noter que cette seconde interface reprend en grande partie l'ergonomie et la philosophie de l'interface Web de RVSC.

1.7.2 Suite logicielle

Un autre point intéressant de l'Horus est la suite logicielle fournie avec par Merging. Elle comprend trois logiciels : Ravenna ASIO Driver, MTDiscovery et Easy Connect. Tous sont uniquement disponibles pour les plate-forme Windows.

Ravenna ASIO Driver est, comme son nom l'indique, un pilote ASIO pour Ravenna. Ainsi il permet à n'importe quel logiciel compatible ASIO,soit la grande majorité des logiciels professionnels dédiés à l'audio sous Windows, de pouvoir échanger avec l'Horus, et plus largement avec n'importe quel appareil Ravenna. Il nous offre donc une troisième solution logicielle, en plus de RVSC et JADE, pour rendre un ordinateur compatible avec Ravenna. Sa configuation s'effectue à travers un utilitaire appelé Merging Ravenna ASIO Pannel permettant de choisir l'interface réseau connectée au réseau Ravenna, la taille de la mémoire tampon (de 64 à 1024 échantillons) ainsi que le nombre d'entrées/sorties allouées (de 8 à 64 par pas de 8). La configuration des flux entrants et sortants se fait quant à elle via une interface

Web accessible via le port 9090 de l'ordinateur hôte.

MTDiscovery est un utilitaire affichant la liste des périphériques Ravenna sur le réseau. Même s'il a été pensé en particulier pour les différents produits Merging, il affiche aussi la liste des périphériques Ravenna tiers.

Enfin Easy Connect permet la réalisation de connexions par Ravenna entre les différents appareils du réseau. C'est une surcouche logicielle qui se veut plus facile d'utilisation que les interfaces Web, permettant la création d'un lien en un seul clic, mais cachant au passage certains mécanismes de Ravenna. Nous préférerons pour cette raison l'utilisation des interfaces Web des différents composants du réseau à celle d'Easy Connect.

1.8 Premiers tests filaires

Avant de procéder à des tests à travers une liaison sans fil, nous avons tout d'abord effectué des tests en filaire. En effet tous les appareils et solutions Ravenna sont vendus pour être utilisés seulement dans cette configuration. Cela nous apportera deux résultats.

Premièrement, nous pourrons valider ou non certaines associations de produits. En effet, même si la plupart des produits devraient être compatibles entre eux immédiatement, Ravenna est une technologie encore jeune et certains dysfonctionnements pourraient survenir. De plus, certains des ordinateurs utilisés dans ces tests respectent quelques recommandations données par les constructeurs *a minima*, et cela pourrait être un facteur limitant dans la réussite des transferts.

Secondement, ces tests nous serviront de bases de comparaison lors de nos tests en sans-fil, afin de pouvoir déterminer immédiatement l'influence des technologies sans-fil sur les performances globales des transferts.

1.8.1 Protocole de test

Tout d'abord l'objectif de ces tests était de savoir si le transfert du son était oui ou non effectif. Ensuite nous avons cherché à quantifier les performances des liaisons audio ainsi réalisées. Pour cela, le dispositif utilisé a été le suivant : Nous envoyions vers l'Horus un flux Ravenna stéréo lu depuis le logiciel Reaper installé sur notre ordinateur. Ce flux était routé vers la paire AES/EBU 1/2 de sortie de l'Horus, qui était directement rebranchée sur la paire AES/EBU 1/2 d'entrée de l'Horus. Ensuite un autre flux Ravenna stéréo prenait comme source l'entrée AES/EBU de l'Horus et était routé vers une entrée stéréo de Reaper. L'ordinateur et l'Horus étaient chacun relié à un des ports de nos routeurs TP-Link via un câble Ethernet. Tous les flux audio utilisaient une fréquence d'échantillonage de 48 kHz et une dynamique d'encodage de 24 bits.

Figure 1.7 – Configuration pour les tests filaires

Avec ce dispositif nous pouvons alors lire un son sur une piste de Reaper et obtenir directement sur une autre piste l'enregistrement de ce son après un aller-retour à travers le réseau Ravenna. Ce test nous permet donc d'évaluer la latence induite par l'utilisation d'un réseau Ravenna, en plus de l'effectivité ou non du transfert. Évidemment ce test ne se veut pas totalement exhaustif, il n'évalue pas en tant que tel le respect des données audio transférées (ne se fiant pour cela qu'à l'oreille des expérimentateurs), ni le nombre maximal de pistes que peut transporter notre réseau Ravenna. Il

s'agit là d'un test préliminaire destiné d'une part à nous familiariser avec la technologie RAVENNA et les concepts qu'elle implique, et d'autre part à nous donner un premier outil de comparaison en cas de succès d'une liaison sans fil. Mais alors il faudra bien évidemment mener d'autres tests complémentaires en filaire et en sans-fil afin d'évaluer réellement les performances de notre liaison, en testant notamment la tenue à la charge et sur le long-terme en lisant un grand nombre de pistes simultanément durant plusieurs heures et en comparant l'intégrité des données reçues, en additionnant l'enregistrement en opposition de phase resynchronisé au signal original par exemple.

1.8.2 Résultats

Nous disposons de trois solutions logicielles différentes pour lire et écrire des flux RAVENNA sur un ordinateur, l'idéal serait donc de réaliser les tests avec toutes les configurations différentes possibles. Malheureusement, si nous avions dans un premier temps réussi une synchronisation purement logicielle entre un ordinateur hébergeant une RVSC et un second disposant de JADE, la RVSC occupant alors le rôle de *grandmaster*, notre ordinateur équipé de RVSC n'a jamais réussi à se synchroniser sur l'HORUS une fois l'interface connectée à notre réseau. De plus après quelques essais, il ne nous a pas semblé pertinent de tester une solution de transfert purement logicielle. En effet, ce sont clairement ces solutions logicielles et les différents étages de mémoire tampons qu'elles incluent qui sont responsables de la quasi-totalité de la latence que nous avons pu mesurer. Une liaison entre deux solutions logicielles n'auraient pu aboutir qu'à de moins bons résultats qu'une configuration incluant l'HORUS. De plus, l'utilisation d'au moins un matériel dédié conçu spécialement pour utiliser RAVENNA est la norme de toute utilisation professionnelle. Il nous reste alors deux configurations à tester, une avec le pilote ASIO RAVENNA de MERGING, l'autre avec JADE.

La première solution que nous avons testée est celle du pilote ASIO fourni par MERGING. Il nous suffit alors d'indiquer dans les options REAPER de l'utiliser comme périphérique audio et, de fait, comme une carte son classique. Nous avons effectué le test avec différentes valeurs de mémoire tampon. Les résultats sont présentés dans le tableau ci-dessous.

Taille mémoire tampon	**Latence mesurée**
64 échantillons (∼1,3 ms @ 48 kHz)	134 échantillons (∼2,8 ms @ 48 kHz)
512 échantillons (∼10,7 ms @ 48 kHz)	1029 échantillons (∼21,4 ms @ 48 kHz)
1024 échantillons (∼21,3 ms @ 48 kHz)	2054 échantillons (∼42,8 ms @ 48 kHz)

TABLE 1.1 – Résultats des tests filaires avec le pilote ASIO Merging

Taille mémoire tampon entrée	**Taille mémoire tampon sortie**	**UDP Driver Cache**	**Latence mesurée**
512 échantillons (∼10,7 ms @ 48 kHz)	1024 échantillons (∼21,3 ms @ 48 kHz)	0 ms	2758 échantillons (∼57,5 ms @ 48 kHz)
		10 ms	3315 échantillons (∼69,0 ms @ 48 kHz)
		20 ms	3784 échantillons (∼78,8 ms @ 48 kHz)
		30 ms	4260 échantillons (∼88,8 ms @ 48 kHz)
64 échantillons (∼1,3 ms @ 48 kHz)	64 échantillons (∼1,3 ms @ 48 kHz)	0 ms	2879 échantillons (∼60,0 ms @ 48 kHz)
		30 ms	4196 échantillons (∼87,4 ms @ 48 kHz)

TABLE 1.2 – Résultats des tests filaires avec JADE

Tous ces essais ont été reconduits plusieurs fois et les résultats obtenus ont été à chaque fois identiques, à un échantillon prêt, conforme donc à la marge d'erreur de notre méthode de mesure. Le résultat intéressant à retenir de ces tests est la latence ajoutée par l'aller retour dans le réseau et dans l'Horus qui peut se calculer comme la différence entre le retard mesuré et la somme des mémoires tampons d'entrée et de sortie du pilote ASIO. On obtient alors une latence ajoutée par le réseau de 5 à 6 échantillons (toutes nos mesures étant réalisées à un échantillon prêt), ce qui correspond à une latence de 0,1 ms à la fréquence d'échantillonnage de 48 kHz. Cette durée est la somme de celle de l'aller-retour des signaux à travers le réseau et du temps de traitement interne à l'Horus (routage entre l'interface réseau, le moteur audio et les interfaces AES/EBU, dans un sens puis dans l'autre).

Concernant JADE, la configuration est un peu plus complexe : Il faut indiquer à Reaper d'utiliser le pilote ASIO fourni avec JADE pour voir ses entrées/sorties disponibles sur la grille de routage. On peut alors créer et ajouter les flux Ravenna à transmettre et reçus à cette même grille à travers les entrées Stream Source et Stream Sink du menu Add de la rubrique Device de l'interface de JADE. On peut alors router les sorties de Reaper dans un flux Ravenna et vice-versa. Concernant les réglages pouvant influer sur la latence, nous avons ici deux niveaux d'action. D'une part le réglage des mémoires tampons comme avec le driver ASIO de Merging, mais cette configuration se fait à travers trois valeurs réglées par l'utilisateur (minimale, préférée et maximale), le logiciel choisissant ensuite automatiquement la valeur qui lui semble la plus adaptée. D'autre part, nous avons aussi accès à un second paramètre réglable pour chaque flux Ravenna émis depuis JADE, l'UDP Driver Cache, qui peut être choisi à 0, 10, 20 ou 30 millisecondes. Certaines combinaisons se sont révélées infructueuses. Le tableau suivant récapitule les résultats pour une sélection de combinaisons de valeurs.

Les résultats sont cette fois-ci un peu plus compliqués à interpréter. On note bien l'influence du paramètre UDP Driver Cache qui ajoute une latence équivalente à sa valeur sur la valeur totale mesurée. Cette observation est cohérente avec le fait que l'option se règle directement sur le flux émis par JADE, et donc n'agit que sur le flux aller, de JADE à l'Horus. En revanche la valeur de latence restante est beaucoup plus importante que dans le cas précédent et ne peut pas être imputée au réseau seul, toujours au vu des résultats précédents. De plus on ne note pas de corrélation directe entre les valeurs mesurées et les réglages des paramètres des tailles de mémoire tampon. On peut donc raisonnablement supposer qu'il existe d'autres mémoires tampons internes à JADE, sûrement afin d'assurer une compatibilité maximale entre les multiples sources que le logiciel peut prendre en compte simultanément.

1.9 Conclusions et observations complémentaires

Cette première partie nous a permis de faire le tour des solutions qui s'offrent à nous pour tester une liaison à travers un réseau Ravenna. En particulier, on pourra retenir deux solutions à travers l'interface Horus, l'une utilisant le driver ASIO de Merging, l'autre le logiciel JADE de Lawo. Au vu des performances, on pourrait être tentés de n'utiliser que la première solution. Mais c'est là qu'intervient une seconde considération. En effet, d'après les résultats du mémoire de William Leveugle et les différentes lectures et échanges que nous avons eus à propos de Ravenna, nous savons d'ores et déjà que l'information la plus sensible à transmettre à travers le réseau est la synchronisation et que celle-ci est effectuée via le protocole PTP. Ainsi au cours des premiers tests nous avons été curieux d'observer les échanges en PTP entre les différents éléments du réseau, dans un fonctionnement « normal »,

lorsque les échanges se font à travers un réseau câblé. Pour cela nous avons utilisé le logiciel WIRESHARK, qui permet de capturer l'ensemble des paquets IP transitant sur le réseau. Nous avons alors été surpris de constater que, si les échanges PTP entre l'HORUS et JADE étaient conformes à ce que nous attendions vis-à-vis du standard, les échanges entre l'HORUS et le driver ASIO de MERGING en différaient quelque peu. Nous aurons l'occasion de revenir en détail sur ces questions après la présentation en détail du protocole PTP dans la deuxième partie, mais c'est cette considération qui nous pousse à tester une liaison sans fil via les deux méthodes et non seulement avec celle offrant les meilleures performances.

Chapitre 2

La problématique du transport d'un signal de synchronisation à travers un WLAN

2.1 Premiers tests de la liaison sans-fil

2.1.1 Préparation du réseau

Après les conclusions de la première partie, nous avons donc cherché à tester le transport de flux Ravenna à travers un WLAN IEEE 802.11. La configuration utilisée a donc été la suivante : d'un côté l'Horus reliée par un câble Ethernet à notre premier routeur sans-fil, de l'autre notre ordinateur équipé du driver ASIO Merging et du logiciel JADE relié par un câble Ethernet à notre second routeur sans-fil.

Concernant la configuration du réseau sans-fil en lui-même, le premier routeur est configuré pour être le point d'accès d'un réseau sans-fil dont le SSID est Wifi-AES67, dans la bande des 5 GHz, tandis que le second routeur est configuré pour être client de ce même réseau sans-fil. Les interfaces Ethernet et sans-fil de chacun des routeurs ont été pontées (ou « bridgées »)

logiciellement, de manière à rendre les changements de médium de transport transparents pour les différents équipements du réseau. En procédant ainsi, chaque routeur dispose alors d'une unique adresse IP pour ses deux interfaces réseau, et les échanges d'informations entre celles-ci se font immédiatement, tous les appareils du réseau appartenant au même sous-réseau (en l'occurrence pour notre réseau, il s'agit du 192.168.1.0/24 en notation CIDR).

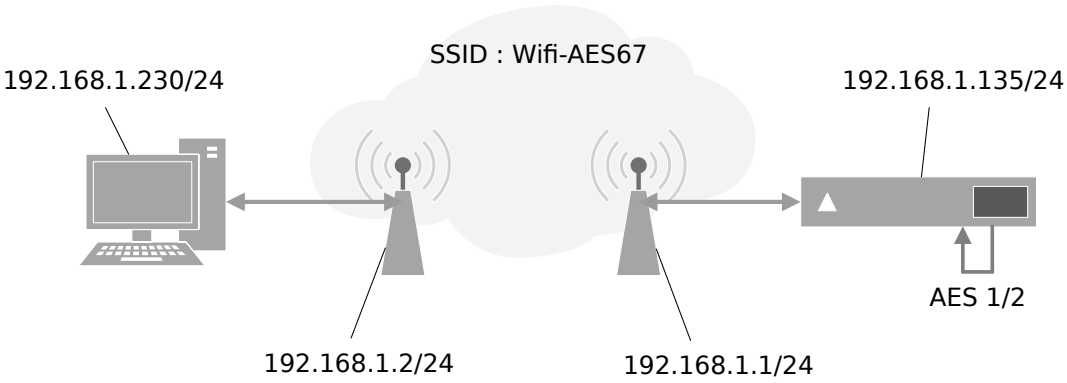

FIGURE 2.1 – Configuration du réseau pour les tests sans-fil

2.1.2 Avec le driver ASIO de Merging

Nous avons tout d'abord cherché à transférer un flux audio sans-fil en utilisant le pilote ASIO de MERGING, solution qui présentait les meilleures performances lors de nos tests préliminaires. Les premières étapes préalables à la transmission sans-fil ne font remarquer aucun dysfonctionnement : l'HORUS est bien détectée sur le réseau à travers l'interface du logiciel MTDISCOVERY et l'interface de gestion du pilote, le MERGING RAVENNA ASIO PANNEL indique bien être synchronisée sur l'HORUS. Mais la mise en œuvre de notre test précédent ne donne aucun résultat. En effet si le logiciel REAPER accepte de lire un fichier audio à travers le pilote ASIO, aucun son n'a jamais été transmis.

À l'aide du logiciel WIRESHARK à nouveau, nous avons cherché à

comprendre l'origine du problème. Nous avions vu dans la partie précédente que dans un fonctionnement « normal », le driver n'émettait aucun paquet PTP sur le réseau, mais recevait quand même les informations de synchronisation de l'Horus. En analysant les paquets échangés cette fois-ci, nous nous sommes rendu compte qu'aucun paquet PTP transportant les informations de synchronisation n'arrivait jusqu'à l'ordinateur hébergeant le driver ASIO. Cette constatation est d'autant plus étonnante que les paquets transportant l'audio étaient quant à eux bien acheminés, à travers des paquets UDP *multicast*, exactement comme les paquets PTP. Mais cela explique les résultats obtenus : les flux Ravenna une fois créés émettent des paquets UDP en permanence sur le réseau, mais l'Horus et l'ordinateur ne disposant pas de référence de temps commune, les paquets arrivent soit en retard par rapport à l'horloge interne, soit trop en avance et ne peuvent pas être stockés dans la mémoire tampon, limitée à quelques milliers d'échantillons au maximum.

2.1.3 Avec JADE

Reprenant l'autre configuration fonctionnelle de notre premier test, le second essai de liaison Ravenna sans-fil s'est fait avec le logiciel JADE et l'Horus. JADE dispose dans son interface d'un voyant « PTP » indiquant la synchronisation du logiciel sur un *grandmaster*. Au lancement du logiciel, lors de nos premiers tests, ce voyant clignotait en général une dizaine de secondes avant de se stabiliser, lorsqu'il avait « accroché » la synchronisation de l'Horus. Avec la liaison sans-fil, c'est le même phénomène que nous observons, augurant de meilleurs résultats qu'avec le driver ASIO. Les flux sont eux aussi immédiatement détectés et nous pouvons donc passer au test à proprement parler, d'autant plus qu'une capture avec Wireshark nous confirme un échange de données conforme à la norme PTP. Ce résultat permet d'ailleurs de lever un doute sur l'expérience précédente, car ayant gardé la même configura-

tion concernant le réseau entre ces deux expériences, on peut conclure que la non synchronisation du driver ASIO et la non transmission des paquets PTP ne sont pas dues à un problème de configuration des routeurs sans-fil, qui dans le cas présent ne font pas obstacle à ces mêmes données.

Les résultats de ce test sont à double tranchant : nous avons réussi à faire transiter un signal stéréo à travers une liaison sans-fil IEEE 802.11, mais la connexion n'a jamais été stable plus de quelques dizaines de secondes avant de « décrocher », de perdre le signal de synchronisation. Cette constatation nous a été permise par le voyant « PTP » de JADE, qui se mettait alors à clignoter quelques instants avant de repasser au vert, signe d'un verrouillage d'horloge à nouveau effectif. Pendant les moments de décrochage, la qualité de l'audio était nettement dégradée, avec de nombreux clics, voire des ralentissements du tempo de lecture, suivis de brutales accélérations lorsque la synchronisation revenait à la normale. Bref, en l'état, une telle liaison n'est clairement pas envisageable, quelle que soit son utilisation.

2.1.4 Conclusion de ces tests

Ces résultats sont certes un peu décevants, dans le sens où rien *a priori* dans les spécifications de Ravenna ou de l'AES67 n'empêche un réseau IEEE 802.11 d'être le médium de transport des données, car compatible avec le transport de données IP. Cependant cela confirme les résultats obtenus par William Leveugle avec le protocole Dante, où la synchronisation avait également fait défaut. Nous allons donc nous pencher maintenant sur le protocole PTP pour essayer de comprendre ce qui peut expliquer l'incompatibilité entre PTP et IEEE 802.11, dégager les différentes solutions qui s'offrent à nous pour surmonter ce problème et étudier pour chacune leur possible mise en pratique.

2.2 Présentation du protocole IEEE 1588-2008 ou PTPv2

2.2.1 Présentation générale

Le protocole PTP, pour Precision Time Protocol a été créé en 2002 par la norme IEEE 1588-2002. Le but de ce protocole est de pouvoir synchroniser toutes les machines d'un réseau sur une horloge commune, avec une grande précision (un réseau PTP bien configuré peut assurer une synchronisation avec une précision de l'ordre de la dizaine de nanosecondes à chaque machine). Il a été développé comme suite au protocole NTP, pour *Network Time Protocol* [16]. Là où NTP permet d'assurer la synchronisation de différentes machines à travers le réseau Internet avec une précision d'une dizaine de millisecondes (c'est le protocole qui est utilisé pour régler automatiquement l'heure des ordinateurs de bureau par exemple, qui se connectent alors à un serveur NTP), PTP se concentre sur la synchronisation des machines d'un réseau local, avec une plus grande précision.

Depuis 2008, il existe une nouvelle version de PTP, nommée communément PTPv2, et définie par la norme IEEE 1588-2008. Contrairement à Dante, publié en 2006 et se basant sur la première version de PTP, Ravenna (publié en 2010) et l'AES67 (juin 2013) se basent sur l'utilisation de la seconde version du protocole, et c'est cette version que nous allons présenter en détail dans la partie qui va suivre. Sauf indication contraire, nous ferons désormais référence à la seconde version en utilisant simplement le terme « PTP ». Il est à noter qu'une troisième version de la norme est actuellement à l'étude et devrait être publiée en 2015.

Comme expliqué dans la première partie, PTP repose sur une architecture maître/esclave, le maître appelé *grandmaster* diffusant alors les informations de synchronisation contenues dans son horloge interne à tous les

appareils du réseau qui synchronisent donc leur propre horloge sur cette référence. Les informations de synchronisation sont transmises sous la forme d'un timestamp. Habituellement le *timestamp* est une donnée utilisée en informatique qui représente le nombre de secondes écoulées depuis le 1er janvier 1970. Mais la précision exigée par le PTP ne permet pas de se limiter aux secondes. Ainsi le *timestamp* transporté par le PTP est divisé en deux parties : d'une part un entier positif codé sur 48 bits contenant la partie entière du *timestamp* exprimé en secondes, et d'autre part un entier positif codé sur 32 bits représentant la partie décimale du *timestamp* exprimé en nanosecondes. La précision des *timestamps* PTP est donc la nanoseconde.

Le PTP est prévu pour pouvoir utiliser différents moyens de transmission des données : Ethernet, UDP sur IPv4, UDP sur IPv6, ControlNET, DeviceNET et PROFINET. Les trois derniers cités sont des protocoles destinés à l'industrie et ne rentrent donc pas dans le cadre de l'utilisation de PTP pour des applications audiovisuelles. Concernant Ethernet, même si quasiment tous les réseaux utilisés pour transporter de l'audio sont aujourd'hui basés sur cette technologie, la plus grande flexibilité d'un réseau IP, indépendant du moyen physique de transport, justifie que cette solution n'a pas été adoptée pour Ravenna et l'AES67. Nous nous intéresserons donc ici uniquement au transport de PTP par des paquets UDP à travers un réseau IP. De plus, même si Ravenna et l'AES67 ont été prévus pour fonctionner à terme sur des réseaux IPv6, ils sont aujourd'hui opérationnels uniquement sur des réseaux IPv4. Nous nous concentrerons donc sur les opérations PTP dans ce mode de fonctionnement.

2.2.2 Principe de synchronisation de deux appareils par le protocole PTP

Le principe de base du protocole PTP peut être résumé ainsi : dans un premier temps, le décalage entre les horloges du maître et de l'esclave ainsi que le délai de propagation d'un message sur le réseau entre le maître et l'esclave PTP sont mesurés. Ensuite, quand l'esclave reçoit un *timestamp* de la part du *grandmaster*, il ajoute à la valeur transmise le délai précédemment mesuré et en déduit le *timestamp* du *grandmaster* au moment de la réception du message lui permettant ainsi de mettre son horloge interne à jour. De plus, en répétant cette opération régulièrement, l'esclave peut aussi estimer l'écart de fréquence entre son horloge interne et celle de son maître et ainsi s'aligner dessus, ce processus s'appelle la syntonisation.

La mise en place de ce mécanisme peut se faire à travers deux modes opératoires : le mode 1-Step ou le mode 2-Step. Nous allons tout d'abord présenter le mode 1-Step. Le maître envoie un message Sync comportant le *timestamp* du moment du départ du message, noté t_1. L'esclave reçoit ce message et note le *timestamp* d'arrivée de celui-ci, noté t_2. Ensuite l'esclave renvoie un message Delay_Req au maître, et note le *timestamp* de départ de ce message, noté t_3. Enfin, le maître note le *timestamp* d'arrivée du message, t_4, et le renvoie à l'esclave dans un message appelé Delay_Resp. Au terme de cet échange, l'esclave connaît les *timestamps* t_1, t_2, t_3 et t_4. Il peut alors calculer l'écart entre son horloge et celle du master donné par la formule :

$$\mathsf{offset} = \frac{1}{2} \cdot \left[(\mathsf{t}_2 - \mathsf{t}_1) - (\mathsf{t}_4 - \mathsf{t}_3) \right],$$

et le délai de transmission d'un message sur le réseau, défini par :

$$\mathsf{delay} = \frac{1}{2} \cdot \left[(\mathsf{t}_2 - \mathsf{t}_1) + (\mathsf{t}_4 - \mathsf{t}_3) \right].$$

Ce mode de calcul, en particulier la division par deux, provient du fait que les deux horloges mises en jeu n'indiquent pas le même temps. C'est d'ailleurs là l'essence même du PTP que de permettre par la suite aux différents protagonistes du réseau de disposer d'une base de temps commune. Cette division par deux n'est donc en aucun cas une moyenne des temps de parcours aller et retour. Au contraire, ce mode de calcul assume que ces temps sont identiques, ce qui est un prérequis du fonctionnement du protocole. En cas de dissymétrie dans le réseau, les performances de PTP sont drastiquement réduites. On peut éventuellement compenser manuellement ces dissymétries, mais en aucun cas le PTP ne peut les détecter et les traiter de lui-même.

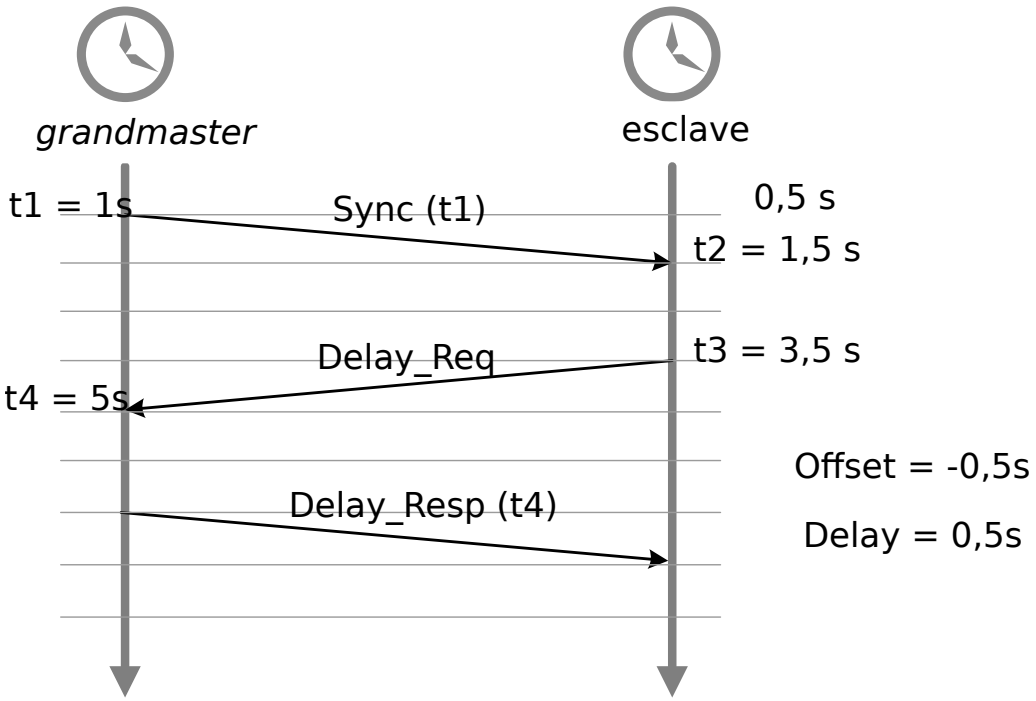

FIGURE 2.2 – Exemple d'échanges PTP 1-Step

Le mode 2-Step diffère du mode 1-Step par l'envoi d'un message Follow_Up par le maître juste après l'envoi d'un message Sync. Pour comprendre l'intérêt de ce mode opératoire, il faut savoir que pour être le plus précis pos-

sible, les *timestamps* de départ et d'arrivée des messages échangés doivent être mesurés le plus bas possible au niveau du matériel, afin d'éviter tous les biais et retards que peuvent provoquer les différents mécanismes intervenant dans l'implémentation de la gestion des données de la carte réseau, comme la mise en attente de certains paquets par exemple. Selon le matériel réseau utilisé et les pilotes associés, il peut alors se révéler difficile voire impossible de relever un *timestamp* de départ d'un message et de l'insérer dans ce message en connaissant avec certitude le temps qui le sépare de son envoi. C'est pour palier à ce problème que le mode 2-Step a été développé. Comme il est plus facile de relever seulement le *timestamp* exact de départ d'un message sans le placer dans ce même message, un *grandmaster* en mode 2-Step se contente d'envoyer une approximation du *timestamp* de départ dans le message Sync, enregistre le *timestamp* exact et envoie l'information à l'esclave dans le message Follow_Up consécutif au message Sync.

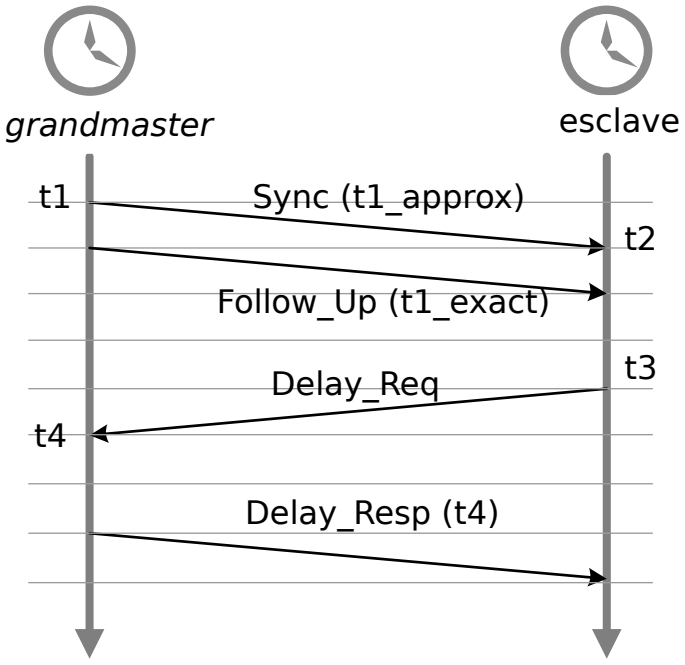

FIGURE 2.3 – Principe du PTP 2-Step

Le fonctionnement en mode 1-Step ou 2-Step est indiqué respectivement par la mise à 0 ou 1 du bit twoStepFlag, qui se trouve dans l'en-tête de chaque message Sync. L'utilisation de l'un ou de l'autre mode ne dépend que du *grandmaster*, les capacités réseau de l'esclave n'étant pas mises en œuvre.

2.2.3 *Hardware timestamping*

Comme on a pu le voir dans la partie précédente, toute la précision du PTP repose sur la capacité des différents systèmes à horodater (ou *timestamper*) les paquets le plus précisément possible. La difficulté de cette opération vient des différents mécanismes entrant en jeu lors de l'envoi de données sur le réseau, en particulier, le passage entre les différentes couches du modèle OSI. En effet, il est très difficile, voire impossible, pour un protocole d'une couche supérieure de connaître le temps exact qui va s'écouler avant l'envoi physique des données sur le réseau. Or c'est précisément cette information qui nous intéresse dans le cas du PTP, car les retards aléatoires sont pris en compte dans les différents *timestamps* PTP, et l'asymétrie et la variabilité au cours du temps qu'ils introduisent faussent les mesures, aussi bien du délai que des fréquences relatives des différentes horloges, jusqu'à rendre le protocole inopérant.

Ainsi, pour assurer les meilleures performances possibles de synchronisation, du matériel dédié au PTP a été développé, permettant l'horodatage des paquets à l'intérieur même de la couche physique. Ce processus est couramment dénommé *hardware timestamping*. C'est par exemple ce qui est présent dans les cartes réseau Intel PRO1000 et dans le contrôleur réseau de l'Horus.

À l'opposé, si une telle implémentation du *timestamping* n'est pas possible, ou peut la reléguer à une couche applicative supérieure, au prix d'une baisse de performances. On trouve ainsi des implémentations au niveau des

pilotes du contrôleur réseau, ou bien au niveau de l'application elle-même. On peut citer comme exemple l'implémentation du PTP par JADE, qui ne nécessite aucun matériel spécifique. On parle alors d'horodatage logiciel ou *software timestamping*.

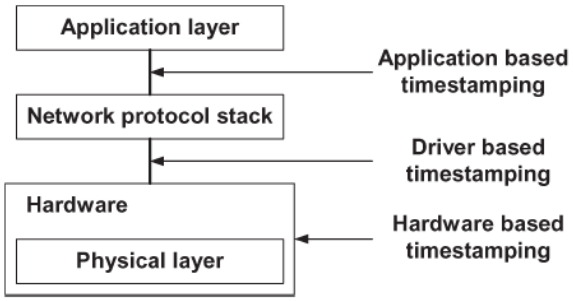

FIGURE 2.4 – Différentes possibilités pour le timestamping

Pour avoir une idée de la différence de performances entre un horodatage matériel ou logiciel des paquets, on pourra se référer à l'étude menée par Bálint Ferencz de la *Budapest University of Technology and Economics* [4], et plus particulièrement les résultats de ses tests reportés Figure 2.5. On peut observer qu'un horodatage matériel assure à la fois une meilleure précision dans le calcul du décalage entre les deux horloges, mais aussi une plus grande stabilité dans le temps de ce calcul. Les résultats donnent un décalage entre les deux horloges de l'ordre de la microseconde dans le cas d'un horodatage matériel et s'étalant entre la microseconde et la milliseconde pour un horodatage logiciel. De plus l'horodatage matériel est beaucoup moins résistant à la montée en charge du trafic réseau.

2.2.4 Modes **End-to-End** et **Peer-to-Peer**

En fait, pour être plus précis, il existe deux modes de fonctionnement global pour la transmission des *timestamps* qui peuvent cohabiter à

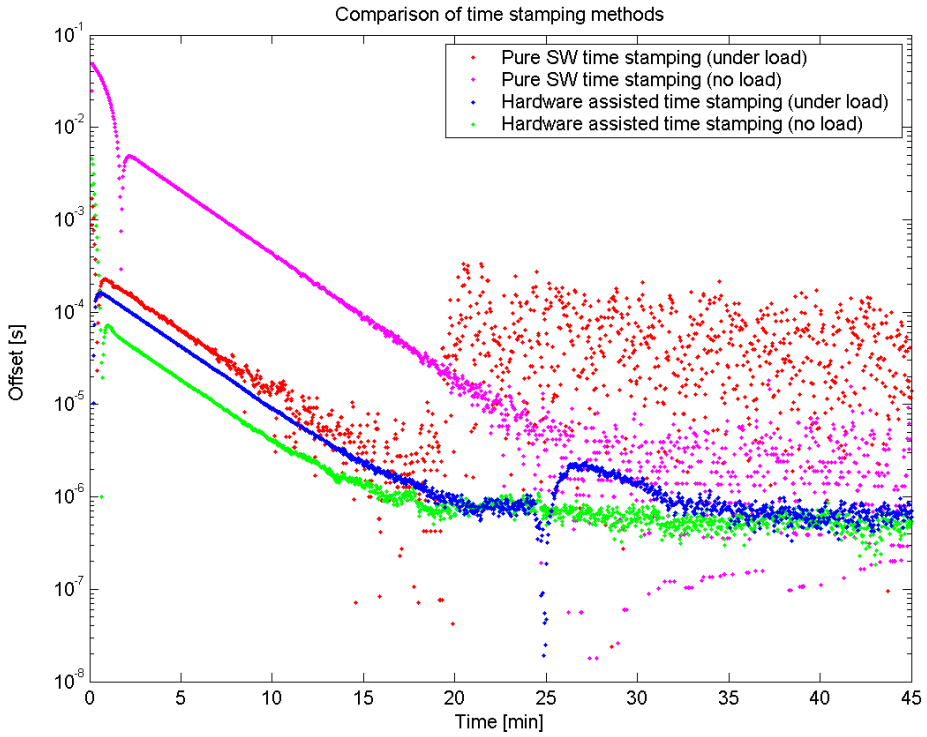

FIGURE 2.5 – Comparaison *hardware* et *software timestamping*

(source : [4])

l'intérieur d'un même réseau PTP : le mode End-to-End et le mode Peer-to-Peer. Le mode présenté dans le paragraphe précédent est le mode End-to-End, où chaque esclave échange directement avec le maître les différentes informations. Il existe un second mode de fonctionnement, le mode Peer-to-Peer. Dans ce mode, chaque esclave reçoit toujours les messages Sync et Follow_Up (si on opère en mode 2-Step) du *grandmaster* afin de calculer l'écart de son horloge avec la sienne, mais c'est la méthode de calcul du délai d'acheminement qui change. Au lieu de calculer le temps global du délai entre le *grandmaster* et chaque esclave par un échange direct de messages Delay_Req et Delay_Resp, chaque élément du réseau calcule le temps d'acheminement seulement avec ses voisins directs (correspondant au terme anglais de *peer*). C'est ensuite en

calculant la somme de ces temps d'acheminement entre voisins qu'un esclave détermine le délai d'acheminement entre lui et le *grandmaster*.

Le calcul d'un temps d'acheminement entre deux voisins se fait par un mécanisme semblable à celui utilisé dans le mode End-to-End. Un appareil envoie un message Pdelay_Req à son voisin et note le *timestamp* t_1 auquel le message est envoyé. Le voisin note le *timestamp* t_2 auquel il reçoit le message Pdelay_Req. Il envoie alors un message Pdelay_Resp contenant ce *timestamp* t_2 ainsi que le *timestamp* t_3 de départ de ce message. L'initiateur du transfert reçoit alors ce message et enregistre en plus le *timestamp* de son arrivée. Il peut alors connaître le délai d'acheminement avec son voisin, défini par :

$$\text{path\_delay} = \frac{1}{2} \cdot [(t_2 - t_1) + (t_4 - t_3)] .$$

Pour les mêmes raisons que citées précédemment, si un appareil ne peut pas déterminer le temps de départ t_3 du message Pdelay_Resp et l'intégrer à ce même message, il peut opérer en mode 2-Step et envoyer simplement une estimation de t_3 dans ce message, puis faire suivre cet envoi d'un message Pdelay_Resp_Follow_Up, qui contiendra la valeur exacte de t_3. Cette opération sera régulièrement renouvelée. À la fin de l'échange, seul l'initiateur de celui-ci connaît le délai d'acheminement. Cette opération est donc initiée aussi régulièrement par l'appareil à l'autre bout du lien local.

Pour transmettre cette information à travers un réseau, il existe un champ correctionField modifié par chacun des appareils traversés. Ce champ correctionField est soit présent directement dans les paquets Sync dans le cas d'un mode opérationnel 1-Step, soit dans le message Follow_Up suivant directement un message Sync dans le cas de l'utilisation du mode 2-Step. Ainsi si un appareil A reçoit un message Sync transmis par un appareil B, il ajoute la valeur du temps de délai d'acheminement entre A et B calculée précédemment

au champ correctionField. Lorsqu'un appareil en bout de chaîne reçoit le message Sync et l'éventuel message Follow_Up correspondant, il a juste à rajouter la valeur du champ correctionField à la valeur du *timestamp* contenue dans le message Sync pour connaître l'heure du *grandmaster* à l'instant où il a reçu le paquet Sync. Il en déduit alors immédiatement le décalage entre son horloge et celle du *grandmaster*.

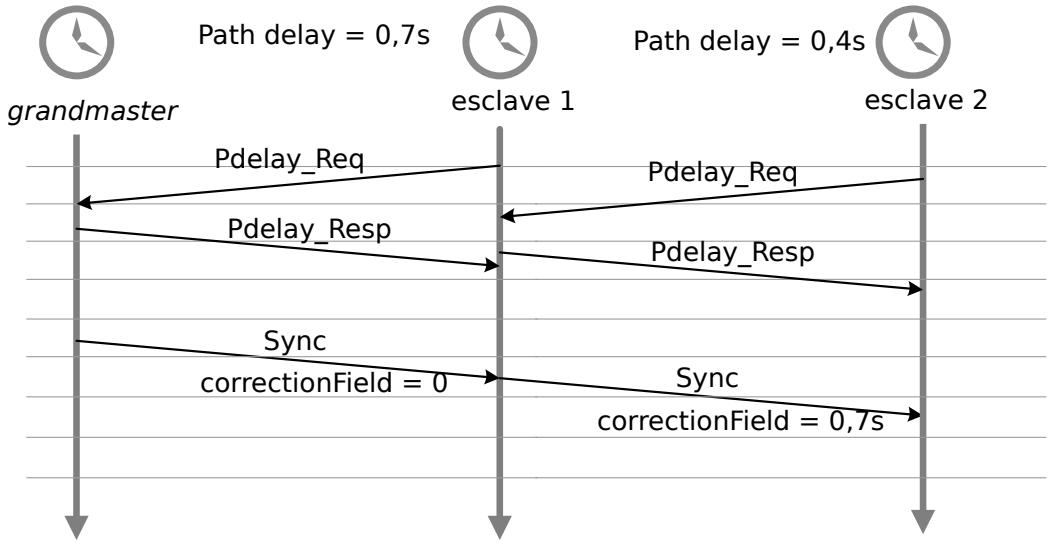

FIGURE 2.6 – Exemple de PTP Peer-to-Peer

Concernant l'AES67, et donc RAVENNA, l'annexe A du standard conseille l'implémentation du mode Peer-to-Peer, mais n'oblige qu'à l'implémentation du mode End-to-End.

2.2.5 Différents types d'horloges PTP

Chaque appareil se synchronisant sur un réseau PTP est une horloge PTP. Pour l'instant, nous n'avons évoqué que des horloges reliées à une seule autre horloge. En terme PTP, ces horloges ne disposent que d'un seul port de synchronisation. Ce type d'horloge est appelé par la norme Ordinary

Clock, ou horloge ordinaire. De plus chaque horloge de ce type peut être maître ou esclave. Mais il existe également des horloges chargées de redistribuer un signal à plusieurs appareils, il s'agit alors d' une horloge avec plusieurs ports. La norme PTP décrit le fonctionnement de deux horloges de ce type, la Transparent Clock (horloge transparente) et la Boundary Clock (horloge d'interface). Ces deux types d'horloge sont particulièrement intéressants à implémenter dans des appareils tels que des routeurs, qui sont par nature les appareils dont les temps de transfert sont les plus aléatoires. Un routeur doté de fonctionnalités PTP, peut corriger les *timestamps* en fonction de la durée de transit des paquets, et assure alors un fonctionnement optimal d'un réseau PTP.

Transparent Clock

Une Transparent Clock est tout d'abord une horloge PTP et dispose donc à ce titre d'une horloge interne lui permettant de timestamper toute entrée ou sortie de paquet. Afin d'annuler son influence sur le temps de transfert (d'où l'utilisation du terme transparent) elle procède ainsi : à chaque réception d'un paquet Sync, elle note le *timestamp* d'arrivée de celui-ci par rapport à son horloge interne, puis le *timestamp* de départ du paquet. En comparant ces deux valeurs, la Transparent Clock dispose du temps de résidence du paquet en son sein. Elle peut alors ajouter cette valeur au champ correctionField prévu à cet effet. En cas de passage dans plusieurs Transparent Clock, les corrections sont additionnées au fur et à mesure.

Boundary Clock

La Boundary Clock fonctionne sur un principe quelque peu différent : au lieu de corriger les messages de synchronisation qu'elle reçoit, elle les intercepte totalement et en réémet de nouveaux. Ainsi, elle dispose d'un port esclave, sur lequel elle reçoit les informations de son maître pour synchroni-

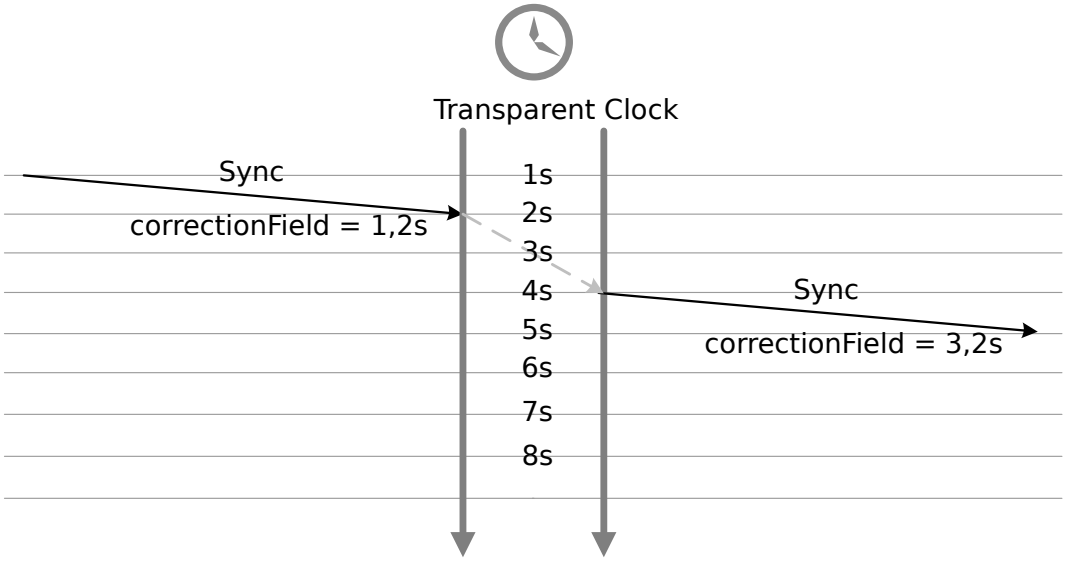

FIGURE 2.7 – Principe d'une Transparent Clock

ser son horloge interne, et tous ses autres ports sont eux-mêmes maîtres des différents appareils auxquels ils sont connectés. Ainsi elle peut se substituer en quelque sorte au *grandmaster*, même si les informations qu'elle diffuse sont toujours synchronisées sur celui-ci.

2.2.6 BMCA

Préalablement à toutes ces opérations de synchronisation, une étape est indispensable : la détermination du *grandmaster*. En effet, comme expliqué dans la première partie, dans un réseau PTP, chaque appareil peut potentiellement être *grandmaster*, contrairement à un réseau de synchronisation traditionnel où le *grandmaster* est déterminé de fait par sa position physique dans le réseau. Pour procéder à ce choix, un algorithme particulier a été développé, il s'agit du *Best Master Clock Algorithm*, en abrégé BMCA. Le fonctionnement de cet algorithme est assez simple puisqu'il consiste principalement en l'échange et la comparaison d'informations entre toutes les horloges deux à

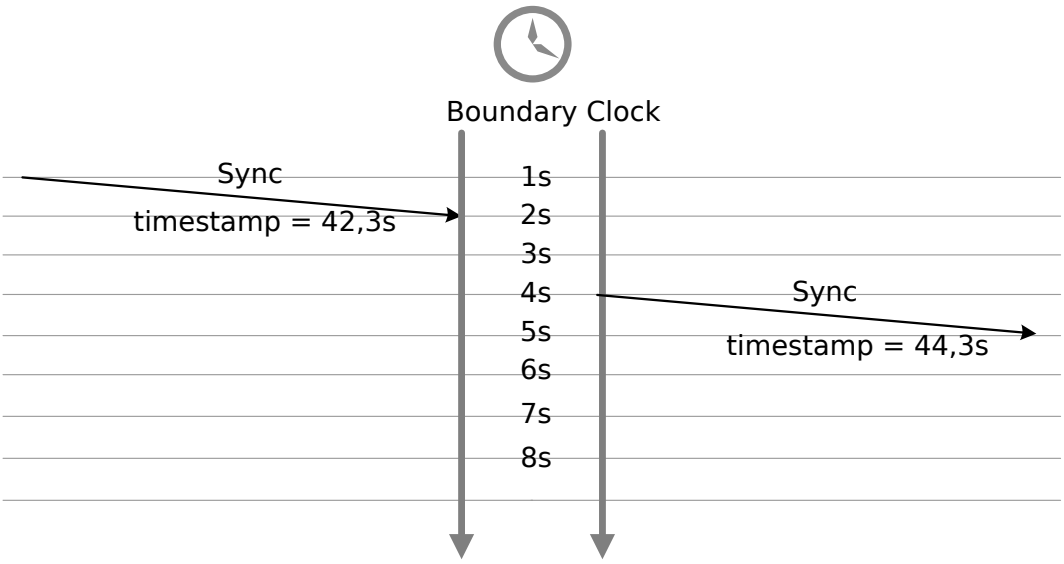

FIGURE 2.8 – Principe d'une Boundary Clock

deux, pour déterminer laquelle est la plus à même d'être *grandmaster*. Nous n'allons pas rentrer ici dans le détail du transfert de ces informations, mais allons les lister et les expliquer.

Pour déterminer laquelle de deux horloges est la mieux adaptée le BMCA compare les critères priority1, clockClass, clockAccuracy, offsetScaledLogVariance, priority2 et clockIdentity des deux horloges, dans cet ordre. Lors de la comparaison successive de ces critères, la première différence entre les deux horloges termine l'algorithme. C'est alors l'horloge avec la plus petite valeur pour le critère comparé qui est choisie.

Voici la signification des différents paramètres :

- priority1 et priority2 sont des valeurs réglables par l'utilisateur pour donner manuellement une préférence ou un désavantage à une ou plusieurs horloges du réseau
- clockClass est un attribut qui indique la traçabilité du temps et de la fréquence distribuée par l'horloge, par exemple, si elle est ou a été synchro-

nisée sur une source de temps primaire (comme une horloge atomique). C'est aussi en attribuant la valeur 255 au paramètre clockClass que l'on indique que l'appareil concerné n'est capable d'opérer que comme esclave

- clockAccuracy désigne la précision estimée de l'horloge si elle opère en tant que *grandmaster*
- offsetScaledLogVariance est la précision des *timestamps* inclus dans les messages PTP par l'horloge quand elle n'est pas synchronisée à une autre
- clockIdentity est une valeur qui sert d'identifiant unique de l'horloge. Dans le cas d'une interface réseau par exemple, sa clockIdentity sera construite à partir de son adresse MAC. Cette valeur n'influe pas sur les performances de l'horloge, mais permet de faire un choix et d'éviter une impasse de l'algorithme si deux horloges présentent des performances identiques sur le réseau.

2.2.7 Profils PTP et utilisation dans l'AES67

Le protocole PTP a été créé pour répondre à de nombreuses problématiques dans des domaines très différents, et, comme on a pu le voir précédemment, dispose de plusieurs options et de différents modes opératoires pour s'adapter aux diverses situations. Pour définir quelles caractéristiques une implémentation PTP doit contenir en fonction de sa finalité d'utilisation, les profils PTP ont été créés. Un profil PTP restreint donc l'éventail des possibilités offertes par la norme, en définissant des valeurs par défaut, des domaines de réglages ou en spécifiant les modes de fonctionnement supportés ou non. L'annexe J de la norme IEEE 1588-2008 définit deux profils par défaut, un pour le mode End-to-End [1] et un pour le mode Peer-to-Peer [2]. Ainsi, le profil par défaut pour le mode End-to-End fixe par exemple la valeur des paramètres priority1 et

1. [6], J.3, p.237, *Delay Request-Response Default PTP profile*
2. [6], J.4, p.238, *Peer-to-Peer Default PTP profile*

priority2 à 128, oblige à l'utilisation du BMCA présent dans la norme (décrit au paragraphe ci-dessus), et fixe des performances minimales de précision pour les horloges.

Dans le cadre de l'utilisation d'un réseau faisant intervenir des appareils répondant au standard AES67, les profils à utiliser sont définis par le standard dans sa section *Synchronization* ([2], Section 4, p.12). Ainsi, chaque appareil AES67 se doit d'implémenter les deux profils par défaut définis par la norme PTP. De plus, le standard AES67 définit un autre profil dans son annexe A, le media profile que le standard recommande d'utiliser. En particulier, il limite l'encapsulation des messages PTP à des paquets UDP transportés par IPv4, ajoute des valeurs possibles au paramètre clockClass, afin de pouvoir rendre compte des performances des horloges par rapport au standard AES11 [3] [3] et définit les performances minimales des horloges impliquées dans un réseau AES67 vis-à-vis de ce même standard AES11.

2.3 Compatibilité entre IEEE 802.11 et PTP

Les réseaux sans-fil IEEE 802.11, permettant le transport de données IP, sont *a priori* compatibles avec le transport de données PTP utilisant cette technologie, comme pour les profils PTP utilisés dans le cadre de l'AES67. Pourtant, notre mise en pratique a échoué, entre autre à cause d'une synchronisation instable. Cet échec peut être imputé en grande partie à l'instabilité que peut présenter un réseau IEEE 802.11 par rapport à un réseau câble Ethernet (IEEE 802.3). En effet, au contraire des réseaux câblés actuels (comme les réseaux Ethernet Gigabit) où toutes les liaisons sont *full-duplex*, c'est-à-dire que les différents appareils disposent de canaux de transmission séparés pour les deux sens de communication, les WLAN ne peuvent par définition qu'être

3. Le standard AES11 est une recommandation de l'AES concernant les méthodes de synchronisation des différents appareils audio-numériques, définissant entre autre la précision des signaux d'horloge à utiliser

half-duplex, autorisant les communications dans un sens, puis dans l'autre, mais jamais les deux en même temps. En effet, le médium de transport utilisé, l'air, étant commun à tous les appareils d'un tel réseau, deux appareils émettant au même moment verraient leurs communications respectives brouillées.

Pour éviter ce phénomène, l'IEEE 802.11 fait appel au mécanisme CSMA/CA ([17], 6.7.3). Le problème de ce mécanisme est la mise en attente des paquets pour un temps indéterminé afin de s'assurer qu'aucun autre appareil n'émet, ce temps d'attente dépendant alors fortement du trafic sur le réseau à cet instant donné. En particulier ce mécanisme introduit des asymétries de temps de transfert à travers le réseau aléatoire pour chaque échange. Or le PTP ne sait pas gérer des asymétries : Il dispose de mécanismes pour les corriger si on les lui indique mais ne peut ni les détecter ni les corriger automatiquement. Il peut s'accommoder de légères variations dans les transferts, mais les trop grandes irrégularités de notre réseau sans-fil se révèlent supérieures à ces limites. Ce problème est critique surtout car nos routeurs sans-fil n'implémentent aucune fonctionnalité PTP, et ne peuvent donc pas corriger ce temps d'attente comme peut le faire une Transparent Clock ou une Boundary Clock, ce qui rend la transmission des informations de synchronisation particulièrement instable et finit par faire échouer le processus global.

La recherche de solutions à ce problème nous a conduit dans un premier temps à la découverte du protocole gPTP, pour *generalized Precision Time Protocol*, défini par la norme IEEE 802.1AS, et proposant des solutions spécifiques au cas des réseaux IEEE 802.11. C'est donc ce protocole et ses particularités que nous allons maintenant étudier, ainsi que sa possible utilisation conjointe avec le protocole PTP.

2.4 IEEE 802.1AS et gPTP

2.4.1 Les normes AVB

Le terme AVB, pour *Audio Video Bridging*, fait référence à un ensemble de normes développées par l'IEEE afin de permettre le transport de données audio et vidéo à travers des réseaux répondant aux standards IEEE 802 (comme les réseaux Ethernet IEEE 802.3 et les réseaux sans-fil IEEE 802.11). Actuellement, on compte cinq normes ratifiées faisant partie de l'ensemble AVB, que nous allons présenter succinctement. Ces normes sont :

- IEEE 802.1BA
- IEEE 802.1Q, reprenant les éléments définis dans les précédentes normes IEEE 802.1Qav et IEEE 802.1Qat
- IEEE 802.1AS
- IEEE 1722
- IEEE 1733

Tout d'abord, la norme IEEE 802.1BA [11] définit comment identifier les différents éléments d'un réseau AVB et ainsi découvrir la topologie du réseau. La norme IEEE 802.1Q [12] définit quant à elle des outils de gestion des ressources réseaux afin d'assurer un acheminement optimal des paquets AVB, en particulier vis-à-vis des contraintes de temps réel. Par exemple, elle définit un mécanisme de réservation de ressources pour les flux à travers un réseau Ethernet, ou bien des mécanismes de gestion des paquets AVB à l'intérieur de routeur. Cette norme fait partie des moyens de gestion de la qualité de service d'un réseau.

Ensuite, la norme qui nous intéresse particulièrement pour notre situation est la norme IEEE 802.1AS [10], puisqu'elle définit un protocole de synchronisation à travers un réseau semblable au PTP, le gPTP, et qu'elle com-

prend entre autre des dispositions spécifiques pour les réseaux IEEE 802.11.

Enfin les normes IEEE 1722 [8] et IEEE 1733 [9] définissent des protocoles de transport des données à travers un réseau AVB. La norme IEEE 1722 définit un protocole s'appuyant sur la couche 2 du modèle OSI, en particulier l'encapsulation de données FireWire sur un réseau Ethernet synchronisées via gPTP. La norme IEEE 1733 s'appuie elle sur la couche 3 du modèle OSI et le protocole RTP, en définissant un format d'encapsulation des données permettant à celles-ci d'être synchronisées via les informations transmises en parallèle par le protocole gPTP.

2.4.2 Différences entre PTP et gPTP

La norme IEEE 802.1AS définit le protocole de synchronisation gPTP, ou *generalized* Precision Time Protocol. La première version de cette norme a été publiée en 2011 [4], soit trois ans après la norme définissant le PTPv2. Le gPTP repose d'ailleurs sur les mêmes mécanismes de base que le PTP. Mieux, nous verrons que gPTP et PTP peuvent s'utiliser conjointement. Mais avant de développer ce point, nous allons ici décrire les principales différences qui existent entre PTP et gPTP, en commençant par celles qui nous ont finalement orientés vers le gPTP.

Toutes les communications entre les différents appareils d'un réseau gPTP se font uniquement à travers la couche 2 du modèle OSI. Ainsi, contrairement au PTP, gPTP ne gère pas les échanges à travers un réseau IP. Le lien entre l'horloge interne d'un appareil implémentant gPTP et son interface réseau peut être découpé en deux parties : une couche d'interfaçage indépendante du médium de transfert, puis une couche spécifique au médium utilisé. Cette approche permet à un appareil gPTP de facilement relier les informa-

4. Une nouvelle version de cette norme a été publiée en ce début 2014, il s'agit de la norme ISO/IEC/IEEE 8802-1AS:2014. Nous n'avons pas pu y avoir accès en entier, mais *a priori* elle serait identique à la précédente norme et ne serait qu'un élargissement de la norme à plusieurs organismes de normalisation.

tions de synchronisation entre les différentes technologies réseau. Elle permet de plus d'étendre facilement les capacités de gPTP à d'autres technologies réseau, ayant été pensée de façon modulaire dès sa conception. En particulier, la norme IEEE 802.1AS définit des couches spécialisées pour quatre médias de communication :

- les connexion Ethernet *full-duplex*
- les réseaux IEEE 802.11
- les connexions EPON, ou *Ethernet Passive Optical Network*[5]
- les réseaux CSN, ou *Coordinated Shared Network*[6]

Au cas particulier, nous nous intéresserons, aux deux premières implémentations du gPTP, pour les réseaux Ethernet *full-duplex* et les réseaux IEEE 802.11.

Contrairement au PTP, le protocole gPTP ne distingue que deux types d'appareils : les End Stations et les Bridges. Une End Station gPTP (ou station terminale) est l'équivalent d'une Ordinary Clock PTP, ne disposant que d'un seul port de synchronisation gPTP, alors qu'un Bridge gPTP, peut être apparenté à une Boundary Clock PTP.

Autre différence, le protocole gPTP ne permet la transmission des informations de synchronisation sur un réseau qu'entre des appareils implémentant le protocole gPTP, appelés par la norme *time-aware systems*[7]. Ainsi, dans le cas de notre plate-forme d'expérimentation, si nous utilisons le protocole gPTP pour synchroniser les différents éléments de notre réseau, les deux routeurs utilisés doivent obligatoirement implémenter eux aussi le protocole. Alors que dans le cas de l'utilisation du PTP, il est autorisé de transmettre des messages à travers des éléments ne supportant pas le protocole, mais cela engendre alors des performances moindres voire insuffisantes, comme on l'a

5. Réseau Ethernet optique passif
6. Réseau partagé coordonné
7. Systèmes prenant en compte le temps

constaté précédemment avec nos premiers tests sans-fil, dans lesquels les routeurs n'implémentaient aucune fonctionnalité PTP.

De cette construction du gPTP découle une autre différence. En effet, les échanges se faisant uniquement entre voisins sur le réseau, la notion de mode opératoire End-to-End n'a pas de sens en gPTP, et les éléments utilisant une interface Ethernet en gPTP sont contraints de reproduire le fonctionnement Peer-to-Peer du PTP. De plus, le gPTP oblige à l'utilisation du mode 2-Step dans le cas de liaisons Ethernet. Aussi, le gPTP oblige à la syntonisation des différentes horloges, quand celle-ci n'est qu'une option PTP. Enfin, le gPTP utilise un BMCA légèrement différent de celui défini dans la norme PTP.

2.4.3 Utilisation conjointe de PTP et gPTP

Comme on l' a vu dans les précédents paragraphes, PTP et gPTP diffèrent sur certains points, mais, étant basés sur les mêmes principes, on peut utiliser les deux protocoles conjointement pour assurer une synchronisation commune à travers tout un réseau. Nous allons maintenant détailler les moyens à mettre en œuvre pour arriver à ce résultat.

Comme on l'a vu précédemment, sur un lien Ethernet, les opérations gPTP sont identiques à une synchronisation par PTP en mode Peer-to-Peer et 2-Step. Cette équivalence est normalisée et complétée par l'annexe F de l'IEEE 802.1AS qui définit justement un profil PTP compatible avec le gPTP pour les liens Ehernet *full-duplex*. D'ailleurs pour les applications qui nous intéressent, le profil AVB est pris en compte par le standard AES67.

Le standard AES67 ne s'arrête pas là, et développe dans son annexe D la manière d'interfacer plusieurs réseaux, certains synchronisés via l'IEEE 1588, les autres via l'IEEE 802.1AS. Ainsi, comme le précise cette annexe, il est possible pour une horloge PTP disposant de plusieurs ports d'appliquer

un profil PTP différent pour chacun de ses ports. Typiquement, une horloge à deux ports, l'un utilisant le profil PTP média et l'autre le profil PTP AVB constitue un élément offrant une interface de liaison entre les deux protocoles de synchronisation sur un réseau AES67.

On parle ici d'une simple interface entre deux réseaux, mais il est possible d'étendre ce raisonnement au-delà, comme expliqué dans *Using an IEEE 802.1AS Network as a Distributed IEEE 1588 Boundary, Ordinary or Transparent Clock* [5]. Ainsi l'idée ici n'est plus d'interfacer deux réseaux utilisant des protocoles différents, mais d'utiliser un réseau entier synchronisé en gPTP pour véhiculer une synchronisation entre deux réseaux synchronisés en PTP. Au niveau du fonctionnement global, on peut alors considérer l'ensemble du réseau IEEE 802.1AS comme une unique horloge PTP avec des ports utilisant le profil PTP AVB. Cette approche nous intéresse particulièrement, car gPTP proposant une méthode spécifique au transport à travers un WLAN IEEE 802.11, on peut alors imaginer pour notre plate-forme de tests un nouveau schéma de transport de la synchronisation : de chaque côté des routeurs sans-fil, on utilise le PTP à travers un réseau Ethernet câblé, et entre les routeurs, on utilise le gPTP et son implémentation spécifique à l'IEEE 802.11. Ainsi, on peut espérer s'affranchir des instabilités subies lors de nos premiers tests sans-fil, et propager proprement un signal de synchronisation à travers notre réseau mixte Ethernet et sans-fil, comme le montre la Figure 2.9.

2.4.4 gPTP et IEEE 802.11

Maintenant que nous avons décrit l'architecture globale du réseau mixte gPTP/PTP à laquelle nous souhaitons parvenir, nous allons nous intéresser aux mécanismes mis en œuvre par l'IEEE 802.1AS pour transmettre une synchronisation à travers un WLAN IEEE 802.11. En effet, si le principe est toujours le même que pour un réseau câblé, c'est-à-dire l'échange de *ti-*

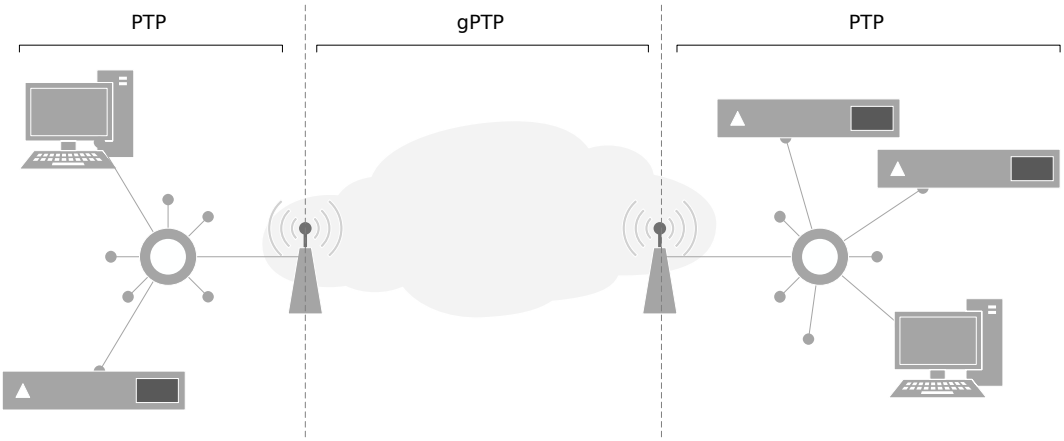

FIGURE 2.9 – Exemple de topologie de réseau mixte PTP/gPTP

mestamps entre deux appareils pour se synchroniser, cette fois on utilise non pas des trames Ethernet ou des paquets IP, mais des trames IEEE 802.11 invoquées par des primitives, regroupées sous le nom de MLME-TIMINGMSMT. Ces primitives ont initialement été définies par l'avant projet IEEE P802.11v [7], puis intégrées à la révision du standard IEEE 802.11 de 2012 [14]. Ces primitives sont au nombre de trois :

- MLME-TIMINGMSMT.request
- MLME-TIMINGMSMT.confirm
- MLME-TIMINGMSMT.indication

L'initiateur de l'échange invoque la primitive MLME-TIMING-MSMT.request, ce qui déclenche l'envoi d'une trame Timing Measurement Action Frame vers l'appareil avec lequel l'initiateur souhaite mesurer l'écart de synchronisation et le délai de transmission. Le *timestamp* de départ de cette trame est noté t_1, le *timestamp* d'arrivée de cette trame est noté t_2. L'arrivée de cette trame déclenche l'envoi d'une trame ACK faisant état de la réception de la trame, et contenant t_2 et t_3, *timestamp* de départ de la trame ACK. De plus, l'envoi de cette trame ACK déclenche l'invocation de la primitive MLME-

TIMINGMSMT.indication au receveur de la première trame, qui lui indique entre autre les *timestamps* t_2 et t_3. À la réception de la trame ACK par l'initiateur, la primitive MLME-TIMINGMSMT.confirm est invoquée et transmet à l'initiateur les *timestamps* t_2 et t_3, ainsi que t_4, *timestamp* d'arrivée de la trame ACK. L'initiateur dispose alors des quatre *timestamps* et peut procéder comme dans le cas des échanges PTP. Ainsi la trame Timing Measurement Action Frame s'apparente à un message PDelay_Req et la réponse ACK à un PDelay_Resp.

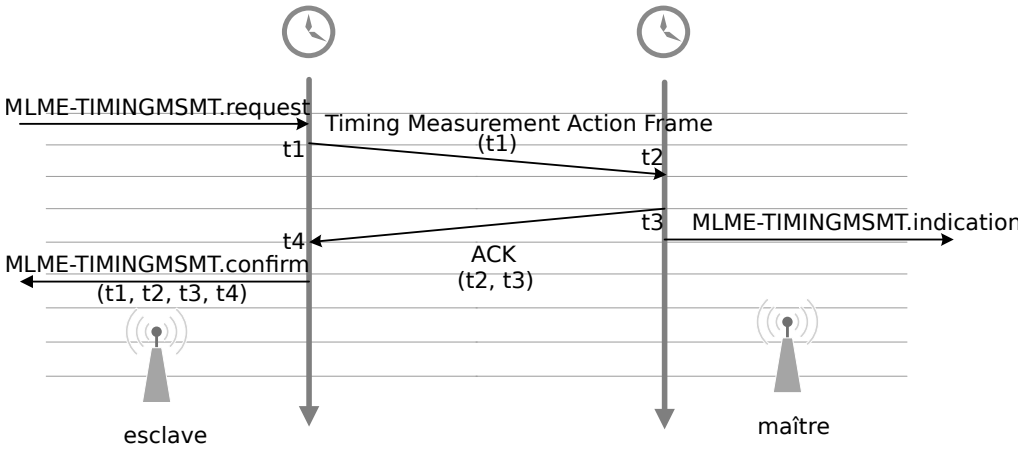

FIGURE 2.10 – Principe du gPTP sur un lien IEEE 802.11

Quelques remarques supplémentaires : les trames Timing Measurement Action Frame, sont uniquement envoyées par les appareils esclaves. À la fin d'un échange, seul l'initiateur connaît donc les 4 *timestamps* . Cependant, les trames Timing Measurement Action Frame contiennent les valeurs de t_1 et t_4 de l'échange précédent. Le maître connaît également les *timestamps* , avec un échange de décalage. Ces données peuvent être utiles par exemple dans le calcul des fréquences relatives entre deux horloges voisines. En effet cette valeur peut être obtenue en calculant $(t'_1 - t_1) / (t'_2 - t_2)$ avec t'_1 et t'_2 les valeurs d'envoi et de réception d'une autre trame Timing Measurement Action Frame.

Avec les éléments développés dans cette partie, nous avons pu ex-

plorer en détail le mécanisme de synchronisation d'un réseau audio-numérique, et proposer une première approche afin de résoudre les problèmes de synchronisation que nous avons expérimentés lorsque nous avons cherché à étendre notre réseau au sans-fil. Nous allons maintenant étudier la mise en pratique d'une telle solution.

Chapitre 3

Perspectives d'évolution et tests complémentaires

3.1 Implémentation du gPTP pour IEEE 802.11

Aujourd'hui, aucun produit commercialisé ne propose de solution clé en main à notre problème de transmission d'un signal de synchronisation entre différents appareils Ravenna à travers un WLAN IEEE 802.11. Dans la partie précédente, nous avons présenté une première possibilité pour résoudre ce problème, mais cela restait théorique. Nous allons donc maintenant étudier une implémentation pratique de cette solution, et évaluer si nous sommes en mesure de la mettre nous-même en œuvre.

Une remarque avant de poursuivre : dans la partie précédente, nous nous sommes contentés de présenter le schéma d'une solution faisant intervenir le gPTP pour la transmission de la synchronisation à travers un réseau sans-fil. Or notre constatation initiale faisait état de la dégradation du signal à cause de la non implémentation d'un protocole de synchronisation à l'intérieur des routeurs. On peut alors imaginer une simple implémentation du protocole PTP interne aux routeurs sans-fil comme solution. Cependant le PTP ne peut être

transmis à travers une liaison 802.11 uniquement s'il utilise des paquets UDP, eux même transformés en trames 802.11 ensuite. Ainsi l'implémentation la plus naturelle de l'horodatage des paquets serait au niveau de la couche UDP. Or la plupart des retards aléatoires introduits à l'origine de l'échec de nos premiers tests sont dus aux mécanismes des couches inférieures, et de la sous-couche MAC en particulier. On pense en particulier au mécanisme CSMA/CA. On pourra se référer à [18], document rédigé avant la norme IEEE 802.1AS et expliquant le besoin d'une synchronisation spécifique pour les réseaux IEEE 802.11. En effet d'après cette étude, le point critique concernant les délais ajoutés aléatoirement dans le protocole 802.11 se situe à la jonction entre la couche MAC et la couche PHY. La couche UDP se situant au-dessus de ces couches, si l'horodatage se fait seulement au niveau de la couche UDP, celui-ci subira les effets des délais aléatoires liés au 802.11 mais ne pourra pas les intégrer. Ainsi une telle solution pourrait améliorer les performances en permettant de prendre en compte les délais d'acheminement des paquets dus au routage logiciel interne, entre les différentes interfaces réseau, mais ne permettrait pas de prendre en compte et de corriger les retards inhérents aux particularités des technologies sans-fil utilisées ici. Une implémentation à une couche plus basse pourrait être envisagée, comme lorsque l'UDP est transmis à travers un contrôleur Ethernet implémentant un mécanisme d'horodatage matériel. Cependant une telle implémentation pour notre interface 802.11 nécessiterait la modification de la couche MAC de celle-ci. Nous préférerons donc par la suite privilégier les solutions faisant intervenir le gPTP et ses mécanismes utilisant des fonctionnalités natives du format 802.11.

L'implémentation de notre solution de synchronisation sans-fil repose sur deux éléments distincts mais reliés : tout d'abord une implémentation de la norme IEEE 802.11 incorporant les primitives MLME-TIMINGMSMT, et ensuite un client gPTP implémentant les deux spécifications relatives aux

réseaux IEEE 802.3 et IEEE 802.11.

3.2 Implémentation des primitives MLME-TIMINGMSMT

La première étape est donc de pouvoir disposer d'un matériel sans-fil implémentant les primitives MLME-TIMINGMSMT de l'IEEE 802.11. Ces primitives sont des fonctions définies par la norme IEEE 802.11, ici au niveau de la sous-couche MAC, et donc intégrées aux pilotes des contrôleurs réseau 802.11 qui peuvent être appelées par des applications utilisant des couches applicatives supérieures. Tout d'abord nous pouvons regarder plus en détail la définition de ces primitives par la norme. Elles sont intégrées au MLME, c'est-à-dire le *MAC Sublayer Management Entity*. Cette partie de l'implémentation du IEEE 802.11 gère les états des points d'accès et des stations, c'est-à-dire tout ce qui est périphérique au transfert de données, comme l'authentification, l'association à un SSID ou encore la gestion des trames Beacon que le point d'accès transmet à toutes les stations afin de les informer sur les caractéristiques du réseau sans-fil.

D'après nos recherches, même si ces primitives sont définies par la norme depuis 2012 et existaient déjà auparavant dans un document de travail de l'IEEE, nous avons trouvé très peu d'informations sur une quelconque implémentation de ces primitives [1] En particulier ce sujet n'est nullement abordé dans les différents documents relatifs à OpenWrt ou aux contrôleurs réseau Atheros auxquels nous avons pu avoir accès. Cette affirmation doit cependant se confronter à une limitation indépendante de notre volonté : Si OpenWrt est

1. Le seul document que nous avons pu consulter en faisant mention est la thèse de Zhang Cui Zhi de l'université de Zhejiang, intitulé *Research of Clock Synchronization Over WIFI Networks* [19]. Malheureusement ce document étant rédigé dans la langue natale de l'auteur, nous n'avons pas pu en tirer beaucoup d'informations. La seule information que nous avons pu en extraire est que cette implémentation ferait appel à l'ajout d'un FPGA.

un projet *open source* disposant donc de nombreuses ressources en libre accès, il n'existe en revanche que très peu d'informations techniques ayant filtré sur les contrôleurs réseau ATHEROS qui nous intéressent, en particulier à propos du circuit gérant la bande des 5GHz. La norme IEEE 802.11 nous a donné de plus un moyen de vérifier cette absence de primitives : l'information de la compatibilité d'un élément du réseau avec les procédures de *timing measurement* est en effet annoncé par la mise à 1 du bit 23 du champ Extended Capabilities présent dans les trames de gestion. La capture de ces trames à l'aide du logiciel WIRESHARK a alors permis de confirmer nos précédentes allégations.

Si ni nos routeurs, ni aucun autre matériel sur le marché n'intègrent ces primitives, que pouvons-nous faire ? Une des pistes à explorer est celle des pilotes GNU/Linux pour les contrôleurs réseau sans-fil. En effet dans les pilotes récents, l'implémentation de la couche MLME est faite logiciellement, à travers un module du noyau GNU/Linux nommé mac80211. En particulier c'est dans le fichier source net/mac80211/ieee80211_sta.c que sont écrites les instructions relatives au MLME. La philosophie *open source* du projet GNU/Linux nous permet d'avoir accès au code source, et même mieux, de le modifier afin d'insérer une implémentation des primitives nous intéressant. Malheureusement, après étude du fichier en question, il est apparu que nous ne disposions pas des connaissances suffisantes en programmation système pour mener à bien cette idée dans le temps imparti à la réalisation de ce mémoire. Cependant, le fait que nous ayons accès à ce fichier et que nous puissions le modifier permet d'envisager cette solution comme une solution viable à plus long terme.

3.3 Implémentation de l'interface gPTP/IEEE 802.11 à l'intérieur d'un client gPTP

Deuxième élément indispensable à l'expérimentation de notre solution : un client gPTP implémentant les recommandations du protocole pour les interfaces IEEE 802.3 et IEEE 802.11. Les résultats de nos recherches précédentes constatant qu'il n'existe pas aujourd'hui d'implémentation des primitives MLME-TIMINGMSMT implique de fait qu'il n'existe aucune implémentation du protocole gPTP pouvant utiliser un WLAN IEEE 802.11. Cependant, nous pouvons explorer les possibilités d'implémenter une telle solution en supposant avoir à disposition un pilote gérant les primitives MLME-TIMINGMSMT, afin d'évaluer le travail et les recherches qu'il restera à accomplir une fois la première étape validée.

Nous nous tournons une nouvelle fois vers les solution *open source*. Il en existe plusieurs implémentant le PTP et le gPTP à des degrés divers. Ainsi nous commencerons avec le code de base de ces différentes implémentations et ajouterons le code nécessaire pour le support du gPTP à travers une interface IEEE 802.11. Cela évite de devoir coder un client PTP/gPTP *ex nihilo*. Toutes les implémentations *open source* dont nous allons parler sont compatibles avec GNU/Linux. Nous pouvons alors imaginer la configuration suivante : la distribution GNU/Linux OpenWrt étant maintenant installée sur nos routeurs, il sera possible d'installer notre client gPTP directement dans la mémoire du routeur. Ainsi si nous parvenons à coder ce client gPTP, aucun matériel supplémentaire ne sera nécessaire par rapport à notre configuration actuelle pour apporter les nouvelles fonctionnalités de synchronisation à notre réseau. C'est cet aspect qui nous a d'emblée particulièrement attirés dans OpenWrt, et a motivé notre choix initial d'un routeur compatible avec cette distribution.

L'implémentation du gPTP qui nous a semblé la plus intéressante à

première vue est OpenAVB, qui regroupe plusieurs projets autour des normes AVB et propose différents programmes d'exemple, dont des clients de test pour des transferts audio, ou des greffons pour certains lecteurs audio afin de les rendre compatibles avec un réseau AVB. Ce projet est soutenu par Intel et semble l'un des plus complets à l'heure actuelle. En particulier, parmi les différents programmes, se trouve un client gPTP implémentant la partie « Ethernet »du protocole. Malheureusement, après exploration du code source, il est apparu que l'implémentation du protocole gPTP qui est proposé ici n'est compatible qu'avec un contrôleur Ethernet I210 d'Intel, ceci pour assurer un *timestamping hardware* des paquets de synchronisation. Ce contrôleur réseau n'étant disponible que sur des cartes réseaux PCI-Express ou des cartes mère, nous ne pouvons donc pas envisager d'intégrer une implémentation de gPTP se basant sur OpenAVB à l'intérieur de l'un de nos routeurs OpenWrt.

Cependant OpenAVB n'est pas la seule solution gPTP *open source* existante. Une alternative est le projet PTPd, pour *Precision Time Protocol daemon*, un *daemon* étant un programme s'exécutant en arrière-plan dans un système informatique. Ce projet assez ancien a aujourd'hui donné naissance à deux projets distincts, ptpd2 et ptpv2d, ajoutant par rapport à leur ancêtre commun le support de la seconde version du PTP, et disposant tous deux du support du profil de l'IEEE 802.1AS pour le PTP. Il existe aussi un autre projet, The Linux PTP Project, qui a pour but d'implémenter le PTP en utilisant des API plus récentes et plus performantes, et intègre aussi le profil PTP 802.1AS. En particulier, par rapport aux deux solutions basées sur PTPd, cette alternative propose nativement un support pour le *hardware timestamping* pour certains matériels [2]. De plus, son implémentation semble plus modulaire que les précédentes, et donc plus à même d'être adaptée pour l'uti-

2. il est cependant à noter qu'il existe une version modifiée de PTPd créée pour les besoins de [4], offrant un support du *hardware timestamping* uniquement pour les contrôleurs réseau Intel 82576 et 82580

Solution	Solution basée sur	gPTP	Implémentation Hardware Timestamping	Conception du code modulaire
OpenAVB	-	Oui	Intel I210	-
PTPd	-	-	Intel 82576 et 82580	-
ptpd2	PTPd	Oui	-	-
ptpv2d	PTPd	Oui	-	-
Linux PTP Project	-	Oui	Oui	Oui

Table 3.1 – Aperçu des différentes solutions PTP et gPTP logicielles

lisation d'une interface IEEE 802.11 ; elle est également prévue initialement pour gérer plusieurs ports PTP, ce qui ne constitue qu'une caractéristique expérimentale pour le logiciel ptpd2. Un des avantages cependant de la solution ptpd2 est la présence dans les dépôts d'OpenWrt d'une version pré-compilée du logiciel pour nos routeurs. Ainsi nous pouvons d'ores et déjà bénéficier d'une synchronisation PTP logicielle sur nos routeurs qui pourra faire l'objet de tests complémentaires et nous sommes sûrs de la compatibilité de ce logiciel avec notre plate-forme de développement.

Concernant la réalisation pratique de cette implémentation, nous n'avons pas cherché à l'effectuer car elle ne saurait aboutir sans disposer au préalable des primitives MLME-TIMINGMSMT. Cependant l'observation des différents codes source nous permet d'être plutôt optimistes quant au succès de son accomplissement, pour peu que l'on dispose de suffisamment de temps, les algorithmes à implémenter étant détaillés dans les normes et les différentes implémentations ne les adaptant qu'à la marge.

3.3.1 L'utopie d'une solution logicielle intégrée au sein d'un routeur ?

La partie précédente nous a présenté les différentes solutions de clients logiciels *open source* pour le PTP/gPTP, et en prenant en compte un certain nombre de modifications, on pourrait les adapter à notre problématique et les intégrer directement à nos routeurs. Comme on l'a vu précédemment,

certains de ces logiciels savent tirer profit des capacités d'horodatage matériel de différents contrôleurs Ethernet. Et on sait qu'un horodatage matériel est un gage de bonnes performances du protocole PTP.

Mais nous nous heurtons alors à un autre problème : si ces composants peuvent être facilement trouvés sur des contrôleur Ethernet au format PCI-Express, il n'existe en revanche pas aujourd'hui de routeurs sans-fil compatibles OpenWrt intégrant de tels composants. Ainsi notre implémentation présenterait forcément des performances réduites en étant forcée d'utiliser un horodatage logiciel. Qu'en serait-il alors si nous abandonnons la contrainte d'OpenWrt ? Il existe en effet des routeurs implémentant le protocole IEEE 1588. Malheureusement ceux-ci sont limités à des interfaces filaires, et leur prix élevé les restreint à des déploiements importants. De plus, même s'ils disposaient d'interfaces sans-fil, leur modèle propriétaire nous empêcherait toute modification interne, et donc l'implémentation de notre client gPTP modifié pour prendre en compte des interfaces 802.11, à moins d'un partenariat avec un des constructeurs actuels, ce qui nous semble peu probable présentement.

Si on peut se contenter d'une implémentation uniquement logicielle dans un premier temps, afin de tester les concepts que nous souhaitons mettre en œuvre, pouvons-nous envisager ces solutions à long terme ? Il nous semble qu'à ce stade de nos travaux deux voies se présentent. D'une part, nous pouvons imaginer réaliser un routeur répondant à nos besoins en se basant sur le châssis d'un ordinateur de bureau auquel nous ajouterions plusieurs contrôleurs réseau gérant le PTP sous forme de cartes d'extension PCI-Express par exemple. Cette solution a plusieurs inconvénients, à commencer par son prix, qui serait nettement plus élevé que notre routeur actuel (il est difficile de concevoir une telle configuration à moins de 300 €, à comparer aux 65 € de notre routeur de test). On peut aussi critiquer son encombrement plus important et la surconsommation d'énergie engendrée par l'utilisation d'un tel

matériel. Cependant, il répondrait à nos besoins, et, de plus, il serait envisageable de ne pas limiter l'utilisation de cet ordinateur à une fonctionnalité de routeur, et d'y implémenter d'autres fonctionnalités, comme des possibilités de surveillance de l'état du réseau, ou même en faire un lecteur/enregistreur audio, celui-ci se trouvant au cœur du réseau audio-numérique.

D'autre part, si le matériel embarqué dédié actuel ne nous convient pas, il est peut-être envisageable de développer un matériel spécifiquement dédié à cette tâche. En effet, plusieurs compagnies développent des circuits dédiés au PTP ou au gPTP destinés à l'intégration. On peut par exemple citer les micro-contrôleurs XMOS, qui contiennent une implémentation AVB et coûtent une quarantaine d'euros à l'unité, ou bien les modules PTP M50 ou M60 de Qulsar, disponibles pour une centaine de dollars pièce. Nous aurons l'occasion de décrire plus en détail ces derniers par la suite, leurs fonctionnalités ne se limitant pas qu'au PTP à travers une interface Ethernet. Il est évident qu'un développement matériel est beaucoup plus lourd qu'un développement purement logiciel et ne fait pas appel aux mêmes connaissances, mais si les éléments de bases disponibles sont fournis avec une grande partie des fonctionnalités souhaitées, ce développement peut être envisageable.

3.4 Une solution alternative : l'utilisation d'horloges GPS

3.4.1 Principe de l'horloge GPS

Depuis le début de cette partie, nous avons envisagé la solution à notre problème de synchronisation en cherchant comment augmenter la précision du transfert de celle-ci à travers notre réseau IEEE 802.11. Nous allons maintenant explorer une autre approche : transmettre la synchronisation tou-

jours en sans-fil, mais sans utiliser notre WLAN. Si le PTP est très efficace pour transmettre une synchronisation sur un réseau local, il devient en revanche plus problématique à utiliser sur des réseaux plus étendus. Cependant le besoin d'une synchronisation précise au niveau mondial est aujourd'hui essentiel dans plusieurs domaines. Ainsi d'autres possibilités existent, comme le NTP que nous avons déjà évoqué, mais c'est une autre solution qui va nous intéresser ici : la synchronisation GPS.

En effet, si le système GPS, pour *Global Positioning System*, est principalement connu pour sa capacité à nous localiser, il est à noter que chacun des vingt-quatre satellites du dispositif embarque aussi à son bord une horloge atomique au césium. La fiabilité de ces horloges est déterminante pour pouvoir donner avec précision une position, le mécanisme reposant sur une triangulation des données provenant de plusieurs satellites. Les informations reçues comportant ces données de temps, il est possible de les utiliser comme référence de temps pour contrôler un oscillateur interne. Un tel assemblage permet d'obtenir une horloge avec une excellente précision de l'ordre de la nanoseconde. On peut donc synchroniser différents appareils à travers le monde en utilisant une référence commune transportée par les signaux GPS, tout cela sans aucun fil reliant lesdits appareils, avec une précision de l'ordre de quelques dizaines de nanosecondes.

3.4.2 Application à un réseau AES67/Ravenna

L'étape suivante consite en l'application de cette solution à notre réseau audio-numérique. Nous avons vu que les signaux GPS peuvent transmettre des données de synchronisation sans-fil, mais nos appareils audio ne sont capables de se synchroniser que grâce au PTP. Il nous faut donc trouver une interface entre les signaux GPS et PTP. Or, il se trouve que ce besoin de synchronisation à l'échelle mondiale est assez répandu dans l'industrie, tout

comme le PTP, et l'horloge GPS étant le moyen le plus simple aujourd'hui pour obtenir une référence de temps mondial, tous les appareils dédiés à être des *grandmaster*s d'un réseau PTP acceptent une référence GPS en entrée.

L'idée serait donc la suivante : de chaque côté du lien sans-fil, nous ajouterions un appareil dédié à recevoir la synchronisation GPS et qui serait *grandmaster* du réseau PTP. Ainsi tous nos appareils audio seraient bien synchronisés sur une référence commune, celle du GPS. D'ailleurs cette possibilité n'est pas limitée au sans-fil, et le document RAVENNA *Operating Principles*[20] mentionne également la possibilité d'utiliser une synchronisation GPS pour étendre l'utilisation de RAVENNA aux WAN, alors qu'une synchronisation PTP traditionnelle ne garantit un fonctionnement qu'à travers des réseaux locaux.

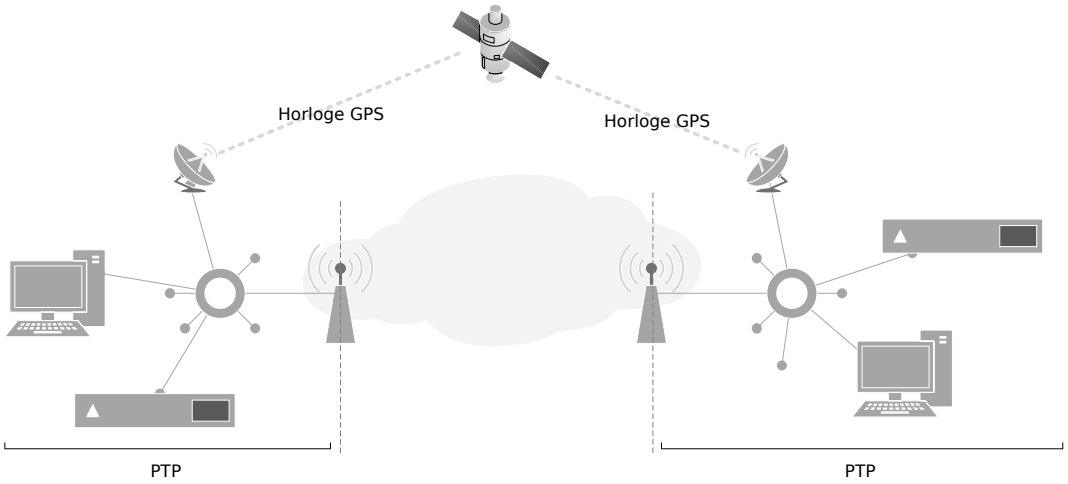

FIGURE 3.1 – Exemple de synchronisation mixte PTP/GPS

Cette solution présente un avantage non négligeable par rapport aux précédentes évoquées : elle n'a besoin *a priori* d'aucun développement logiciel ou matériel, et peut donc être la solution la plus rapide à mettre en œuvre. Malheureusement nous n'avons pas pu la tester pour des raisons de budget. En effet le prix d'un *grandmaster* PTP est bien plus élevé que le budget alloué pour notre mémoire : par exemple le modèle OTMC 100i d'Omicron

Lab coûte environ 2000 €, tandis que le modèle Sonoma D12 d'EndRun Technologies est vendu un peu moins de 4000 €, ce coût étant de plus à doubler pour convenir à notre utilisation. Cependant, si nous n'avons pas le budget adéquat dans le cadre de ce mémoire, et n'avons pas réussi à trouver un partenaire pouvant mettre un tel objet à notre disposition, il est intéressant de noter qu'un tel investissement n'est absolument pas infondé dans le cadre d'une production audiovisuelle et que, si cette solution s'avère fonctionnelle, elle semble aussi viable économiquement.

Il pourrait tout de même être intéressant de disposer d'une telle solution à plus faible coût, pour des applications avec un grand nombre d' « ilôts »filaires reliés entre eux en sans-fil, chaque « ilôt »devant alors disposer d'une synchronisation sur le réseau GPS. On pourrait même pousser notre raisonnement plus loin et imaginer un module GPS/802.11 directement intégrable à des appareils audio. On peut se référer à ce que propose aujourd'hui Powersoft avec ses enceintes Deva. Ce sont des enceintes amplifiées, qui peuvent être alimentées par une batterie et/ou un panneau solaire, et qui offrent la possibilité de recevoir des flux audio à travers un réseau 802.11. Mais ces flux étant limités à une stéréo compressée et utilisant un protocole propriétaire, sortent de la visée de ce mémoire. Néanmoins, le concept tout-en-un du produit est intéressant. Proposer un tel produit compatible avec un réseau AES67/Ravenna pourrait alors ouvrir de nouvelles possibilités de sonorisation et offrir un nouveau support d'expérimentations sonores.

Pour être acceptée dans le milieu professionnel et être viable économiquement une telle fonctionnalité peut difficilement engendrer un surcoût de plusieurs milliers d'euros par rapport à un appareil traditionnel (même si une enceinte Deva est vendue près de 3700 € sans son panneau solaire). Nous avons dans la partie précédente évoqué les modules dédiés M50 et M60 de Qulsar comme solutions matérielles à intégrer pour bénéficier d'un horoda-

tage PTP matériel. Il se trouve que ces modules intègrent aussi un récepteur GPS sur lequel ils peuvent synchroniser leur horloge interne, et offrent les performances nécessaires pour pouvoir opérer en tant que *grandmaster* PTP. Ainsi ils pourraient convenir comme solution adéquate à intégrer au sein d'un routeur sans-fil, qui ferait alors office de *grandmaster* synchronisé sur une horloge GPS, aussi bien qu'au sein d'un appareil sans-fil isolé qui profiterait lui aussi de la synchronisation par GPS.

Si la synchronisation par GPS peut s'avérer, au vu des arguments précédents, une excellente solution en terme de précision et de rapidité de mise en œuvre, elle souffre cependant d'un gros handicap. En effet, la réception d'un signal GPS n'est possible qu'en extérieur, avec une antenne disposant d'une visibilité directe du ciel. S'il est bien entendu possible d'étendre un câble entre l'antenne et le récepteur, cela perd tout intérêt dans notre recherche de la suppression de ces câbles. Néanmoins la solution du GPS reste viable en extérieur, et pourrait être utile dans certaines applications intérieures, mais elle ne peut en aucun cas résoudre à elle seule notre problème.

3.5 Au-delà de la synchronisation

Depuis le début de la deuxième partie, nous nous sommes concentrés sur le problème du transfert de la synchronisation entre les deux extrémités de notre réseau sans-fil. C'est en effet le premier obstacle qui nous est apparu, et qui empêche pour l'instant toute opération. Cependant l'absence de synchronisation n'est pas le seul facteur qui peut empêcher le bon déroulement de nos liaisons audio. Nous devons aussi nous assurer que les paquets transportant les données audio elles-mêmes, à travers le protocole RTP, arrivent tous à destination sans erreur et avec la latence de transport la plus faible possible. Or nous ne sommes pas actuellement en mesure de mettre en œuvre les différentes

solutions que nous avons évoquées auparavant pour acheminer une synchronisation à travers notre réseau et tester les éventuels problèmes qui pourraient se présenter. Le contenu de cette sous-partie traite donc de la question suivante : Est-il possible de mettre en place une double liaison filaire/sans-fil, et de configurer nos routeurs afin que seules les données relatives à la synchronisation empruntent le chemin câblé, tandis que toutes les autres données sont transférées à travers notre lien 802.11.

3.5.1 Configuration des routeurs pour la liaison mixte filaire/sans-fil

Notre idée est donc la suivante : par rapport à notre configuration de test initiale, nous rajoutons un câble entre les deux routeurs, et nous cherchons ensuite à configurer nos routeurs pour que seuls les paquets relevant de la synchronisation, c'est-à-dire ceux du protocole PTP, empruntent le chemin câblé. À première vue, cette configuration est simple d'un point de vue intellectuel : nous disposons de deux routeurs connectés à travers deux liens, l'un radio, l'autre câblé, et chacun de nos routeurs est capable d'appliquer des règles de routage pour réguler les échanges d'informations sur le réseau.

Cependant plusieurs différences par rapport à notre configuration initiale sont à prendre en compte. Tout d'abord, nos routeurs avaient jusqu'à présent leurs interfaces configurées en mode *bridge* : toutes les interfaces filaires et sans-fil sont alors vues par le routeur comme une unique interface virtuelle. Cette configuration est celle par défaut des routeurs et est parfaite pour gérer un unique sous-réseau. Mais dans le cas de notre configuration mixte, elle trouve ses limites : en effet, tous les paquets apparaissant provenir d'une même interface, on ne peut pas appliquer de règle de routage pour les rediriger vers l'une ou l'autre des interfaces réseau physiques.

La solution est donc de supprimer le mode *bridge*, et de configurer

chacune des interfaces séparément. Nous obtenons donc quatre sous-réseaux : deux dédiés aux appareils Ravenna (un connecté à chaque routeur), un comportant les deux interfaces 802.11 et un dernier dédié aux interfaces Ethernet pour le transport de la synchronisation. Cette topologie réseau demande plus de réglages pour être fonctionnelle, les échanges entre différents sous-réseaux n'étant pas possibles de base, il faut configurer les chemins empruntés par les paquets IP afin d'autoriser le trafic d'une extrémité à l'autre du réseau. Ceci se fait en utilisant le logiciel iproute intégré de base à toute distribution GNU/Linux et donc à notre distribution OpenWrt. Une fois cette configuration effectuée, il est possible de réfléchir aux règles de routage que nous voulons mettre en place.

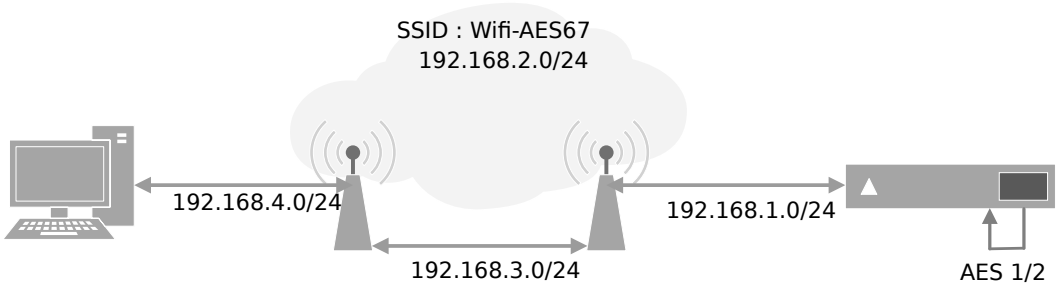

Figure 3.2 – Configuration des sous-réseaux pour la liaison mixte filaire/sans-fil

Les premiers tests de cette configuration ont révélé un problème majeur : si les différents éléments de nos sous-réseaux peuvent communiquer entre eux, cela ne concerne que les échanges unicast, c'est-à-dire d'un ordinateur à un autre. Or tous les protocoles utilisés par l'AES67 et Ravenna reposent sur l'utilisation de paquets multicast, c'est-à-dire qu'un unique message envoyé par un ordinateur peut être acheminé à plusieurs destinataires.

3.5.2 Routage multicast

Avant d'examiner les solutions à notre problème du routage multicast, nous allons brièvement expliquer quelques principes de son fonctionnement, et surtout lister précisément les différents messages multicast échangés par les appareils RAVENNA.

Pour envoyer un message IP à différents ordinateurs, on utilise des adresses particulières, communément appelées adresses multicast. En IPv4, on utilise la plage 224.0.0.0/4, soient les adresses de 224.0.0.0 à 239.255.255.255. Pour qu'un appareil connecté reçoive un message MULTICAST, il doit s'inscrire au groupe de diffusion correspondant à l'adresse de diffusion de ces messages. Cela peut se faire au travers du protocole IGMP, lui-même utilisant un échange de messages multicast sur l'adresse de diffusion 224.0.0.22. C'est le premier type de messages multicast que l'on peut retrouver dans un réseau RAVENNA. Pour la découverte des différents appareils et services du réseau, Ravenna et l'AES67 proposent différentes solutions, mais nous avons relevé en particulier dans notre cas des messages IP provenant du protocole mDNS utilisant l'adresse 224.0.0.251.

Reste maintenant les deux principaux types de données véhiculées par notre réseau : les données de synchronisation et les données audio elles-mêmes. Le protocole PTP utilisé à travers des paquets UDP/IPv4 utilise l'adresse multicast 224.0.1.129. Pour l'audio, les choses sont un peu différentes. Ravenna est basée sur la définition de flux audio, comprenant un ou plusieurs canaux audio, et chaque flux dispose de sa propre adresse multicast. On ne peut donc pas les connaître précisément à l'avance. Cependant après obersvation on peut noter que les flux créés par JADE utilisent des adresses de la plage 232.0.0.0/8. Ceci dénote une différence entre RAVENNA et l'AES67, qui impose l'utilisation d'adresses de la plage 239.0.0.0/8. De plus pour nos expériences ultérieures, il est à noter que toutes les solutions RAVENNA que nous avons pu

tester proposent de régler manuellement l'adresse de diffusion de chaque flux.

Maintenant que nous avons identifié les différents échanges multicast ayant lieu sur notre réseau RAVENNA, nous pouvons explorer les différents moyens de router ces paquets à travers notre réseau. En effet, si iproute est la solution par défaut pour le routage unicast, il existe différents projets pour le routage multicast. Les plus répandus sont pimd, mrouted et SMCRoute. Pour des raisons de facilité de mise en place, nous avons choisi d'utiliser SMCRoute, qui repose sur des règles de routage statiques suffisantes pour une première expérience et qui surtout est disponible pré-compilé pour nos routeurs dans les dépôts logiciels d'OpenWRT. Une règle de routage SMCRoute est très simple à mettre en place : il suffit de fournir au logiciel l'interface réseau d'entrée des paquets, l'adresse IP de l'appareil émetteur, l'adresse multicast et l'interface réseau de sortie sur laquelle on souhaite router ce flux multicast.

Nous avons donc mis en place les règles de routage spécifiques à notre réseau RAVENNA. Les premiers résultats ont été encourageants puisque nous avons réussi à synchroniser JADE et l'HORUS en PTP, et à faire parvenir un flux audio depuis JADE jusqu'à l'interface. De plus les captures des différentes interfaces réseau ont permis de vérifier que notre routage était bien effectif pour la séparation de ces données entre nos liens filaire et sans-fil. Cependant, il nous a été impossible d'indiquer à l'HORUS d'utiliser les données audio reçues, l'interface ne détectant pas la présence d'un flux audio sur le réseau. Les captures de trafic réseau ont permis de comprendre l'origine du problème : si les données PTP et audio sont correctement transmises, les autres données multicast ne semblent pas être concernées par les règles de routage, pourtant identiques à celles du PTP et de l'audio, et c'est grâce à ces données que les informations à propos des différents flux disponibles sur le réseau sont transmises. Des recherches supplémentaires ont permis de mettre en évidence le problème : en effet la RFC 1112 [15] précise que les adresses de la plage

224.0.0.0/24 (de 224.0.0.0 à 224.0.0.255) sont réservées à l'usage de messages destinés uniquement à un unique lien réseau, qui ne peuvent donc pas être transmis à travers différents sous-réseaux. Ce comportement est implémenté par défaut dans le noyau GNU/Linux, et il est donc impossible de parvenir à faire transiter les données manquantes à travers cette topologie réseau.

3.5.3 Conclusion

La réalisation de cette liaison mixte semblait une possibilité aisée à mettre en œuvre pour pouvoir tester les aspects de notre liaison sans-fil autre que la synchronisation. Cependant la mise en œuvre s'est avérée nécessiter la mise en place de solutions logicielles spécialisées, mais surtout de nombreuses connaissances en ingénierie réseau, et les nôtres se sont montrées rapidement insuffisantes, ce qui n'a pas facilité la recherche des solutions adéquates. Nous sommes cependant arrivé à une solution proche d'être effective, et nul doute qu'avec du temps et des réponses techniques supplémentaires, une telle solution serait pleinement fonctionnelle.

Si nous n'avons pas de réponse pratique à la question de la compatibilité des liens 802.11 avec les demandes de Ravenna et de l'AES67 pour le transfert de données audio, nous pouvons en revanche avancer quelques éléments de réflexion qui seront à confirmer par des expériences futures. Tout d'abord Ravenna et l'AES67 utilisent le protocole RTP, et on sait que celui-ci est capable d'acheminer des données audiovisuelles en temps réel à travers un lien 802.11 sans être associé à une synchronisation PTP, par exemple pour le visionnage de vidéo en ligne ou la diffusion de son à travers un réseau domestique. Ceci est un bon point de départ, mais les débits et latence mis en jeu dans ces cas de figure n'ont rien à voir avec les demandes de nos réseaux audio-numériques professionnels, comme nous l'avons dit en introduction de ce mémoire. Cependant la demande croissante pour certains services, comme

la télévision haute définition par Internet pousse au développement de solutions de plus en plus performantes en ce sens. Il est à ce titre intéressant de remarquer que l'un des derniers amendements validés de la norme IEEE 802.11 concerne justement le transfert de contenus audiovisuels. Il s'agit de l'amendement IEEE 802.11aa publié en 2012 et intitulé *MAC Enhancements for Robust Audio Video Streaming*[3] [13]. On peut donc espérer que si les produits grand publics intègre peu à peu des fonctionnalités dédiés aux contenus audiovisuels, il ne sera que plus facile de trouver des produits satisfaisant nativement aux contraintes d'un réseau Ravenna/AES67.

3.5.4 Présentation des liaisons sans-fil *full-duplex*

Depuis le début de ce mémoire, nous nous sommes concentrés sur les liaisons sans-fil conformes au standard IEEE 802.11 afin de garder une compatibilité à terme avec le plus grand nombre de produits. Cependant, comme nous l'avons vu précédemment, la nature *half-duplex* de cette liaison et l'utilisation du mécanisme CSMA/CA dégradent les performances des liaisons en introduisant une latence aléatoire, fatale aux mécanismes de synchronisation actuellement employés dans les réseaux audio-numériques. Pourtant nos recherches parmi les produits de différents constructeurs dédiés aux liaisons informatiques sans-fil nous ont conduit à découvrir des appareils proposant des liaisons sans-fil point-à-point *full-duplex*. En particulier notre attention a été retenue par la série AirFiber d'Ubiquiti et le protocole nstreme-dual de MikroTik. Ces deux solutions se basent sur le même principe : pour pouvoir offrir une liaison *full-duplex*, les transferts sont répartis sur deux fréquences différentes, chacune étant dédiée à un sens de la liaison. Chaque appareil dispose alors de deux circuits radio distincts, un pour la réception et l'autre pour l'émission, réglés sur une fréquence différente. C'est l'addition des opérations

3. Améliorations de la couche MAC pour des flux audio/vidéo robustes

de ces deux circuits qui assure alors la prise en charge des fonctionnalités de la couche MAC de l'interface réseau. Mais, de par sa nature, une liason *full-duplex* ne peut fonctionner qu'entre deux appareils. Par exemple, au cas particulier, chaque sens de liaison entre deux appareils distincts nécessite un circuit radio dédié à ce sens et à cet appareil. Ainsi l'augmentation du nombre de participants à ce réseau multiplierait le nombre de circuits radio que chaque appareil doit intégrer (deux pour chaque appareil avec qui il veut établir une liaison *full-duplex*), mais surtout accroîtrait rapidement le nombre de fréquences différentes occupées par le réseau, et la situation deviendrait très vite ingérable. Toutefois ces produits éliminant potentiellement les aléas dus au mécanisme CSMA/CA, il serait intéressant de pouvoir tester notre configuration en remplaçant nos routeurs par un de ces produits. En cas de succès, cette solution s'avèrerait incomplète, mais pourrait déjà représenter une solution clé en main dans certaines situations. Typiquement, pour des applications de sonorisation où un unique lien sans-fil entre la régie façade et la scène suffirait.

La série AirFiber d'Ubiquiti est une solution toute intégrée, se présentant sous la forme de deux doubles antennes paraboliques intégrant chacune un port Ethernet Gigabit pour être reliée au reste du réseau. Les données du constructeur annoncent un débit supérieur au Gigabit par seconde et des distances de transmissions supérieures à une dizaine de kilomètres, ces valeurs étant bien au-delà des minimums requis pour être appliqués à l'audio. Il existe deux modèles : un émettant dans la bande des 24 GHz et un autre dans la bande des 5 GHz. Malheureusement, la bande des 24 GHz n'est pas libre d'utilisation en France, et le modèle 5 GHz n'est pas encore disponible à la vente en France. Nous n'avons donc pas pu les avoir à disposition pour effectuer des tests.

L'autre solution, celle de MikroTik, opte pour une approche plus modulaire. Ainsi n'importe quel routeur de la marque disposant de deux inter-

faces sans-fil peut-être configuré pour utiliser le protocole NSTREME-DUAL et opérer une liaison sans-fil *full-duplex*. Notre intérêt s'est porté sur une configuration à base d'un routeur RB433GL à chaque extrémité de notre pont sans-fil, disposant de trois ports Ethernet Gigabit et de trois emplacements Mini-PCI. Pour les connexions sans-fil, nous pourrons alors équiper chacun des deux routeurs de deux cartes Mini-PCI R52N, compatibles avec le standard IEEE 802.11n et disposant d'un débit de transmission théorique de 300 Mbps.

FIGURE 3.3 – Routeur MIKROTIK RB433GL *(source : mikrotik-shop.de)*

3.5.5 Configuration de la liaison sans-fil

Nous nous sommes procurés deux routeurs RB433GL et quatre cartes d'extension R52N, ainsi que les boîtiers, alimentations et antennes associés. Le routeur RB433GL se présente sous la forme d'une carte électronique disposant de trois ports Ethernet Gigabit (dont un supportant le standard PoE, et pouvant donc servir à alimenter la carte), un port USB 2.0, un

connecteur d'alimentation, ainsi que trois ports Mini-PCI pouvant accueillir des cartes d'extension, en particulier nos R52n. Ces cartes supportent les standards IEEE 802.11a, 802.11b, 802.11g et 802.11n sur les bandes de fréquences 2.4 GHz et 5 GHz. Elles supportent le MIMO 2x2 et nous disposons de la version R52nM, disposant de deux connecteurs MMCX pour les antennes. Les routeurs sont fournis avec le système d'exploitation RouterOS de Mikrotik, dédié à la gestion des appareils dédiés au réseau. Bien que basé sur un noyau GNU/Linux, ce système d'exploitation n'est pas open source, et il n'est donc pas envisageable d'y intégrer nos propres programmes par exemple. En revanche, RouterOS est fourni avec une interface de gestion très complète, accessible par une interface Web, ou via un logiciel dédié pour Windows, nommé WinBox. Il est aussi possible de se connecter à distance à travers une liaison SSH pour accéder à une interface de gestion en ligne de commande des routeurs. L'interface de gestion permet de configurer tous les paramètres réseaux, mais dispose aussi de nombreux outils de surveillance et de tests, notamment pour tester la bande passante, scanner les fréquences afin de déterminer lesquelles ne sont pas déjà utilisées par d'autres appareils ou observer la qualité des liaisons entre différents appareils du réseau.

Le principe de l'utilisation de nstreme-dual est donc le suivant : après avoir installé et configuré les deux cartes 802.11 en mode nstreme-dual-slave sur le routeur, on crée une nouvelle interface réseau virtuelle nstreme-dual et on lui indique alors quelle carte utiliser en émission et en réception, les fréquences utilisées, ainsi que l'adresse MAC de la carte réseau de réception se trouvant à l'autre bout de la liaison nstreme-dual. On peut ensuite régler tous les paramètres concernant l'encodage des données en elles-mêmes, comme la largeur des bandes de fréquences utilisées, les taux de transferts supportés, ainsi que les types de codage, ou MCS, supportés. Ces réglages sont à effectuer conjointement sur les deux routeurs prenant part à la liaison *full-duplex*.

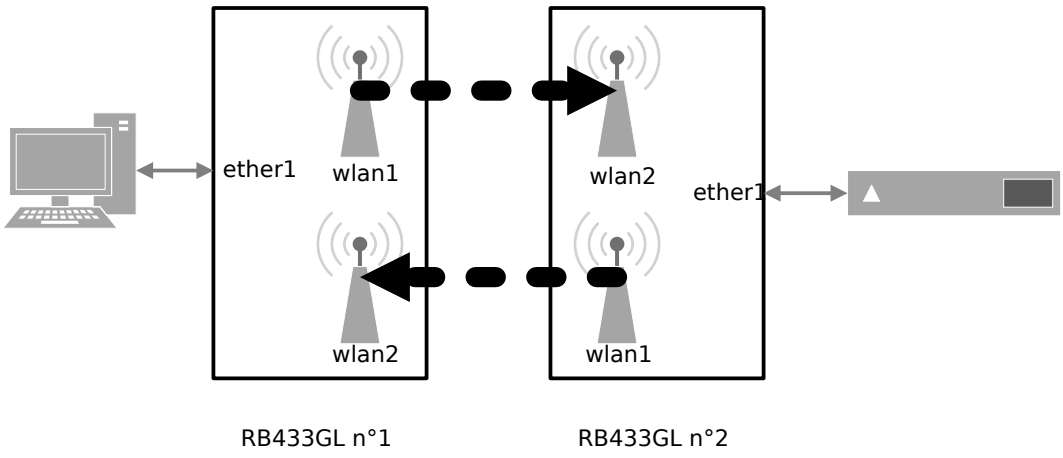

FIGURE 3.4 – Principe de la liaison *full-duplex* dual-nstreme

Nous avons tout d'abord cherché à faire fonctionner notre liaison en utilisant deux fréquences de la bande des 5 GHz, pour garder l'une de nos idées de départ qui était que cette bande étant moins utilisée que la bande des 2.4 GHz, elle est susceptible d'offrir de meilleurs résultats. Malheureusement, notre liaison ainsi configurée se révélait extrêmement instable, et surtout les performances pendant les périodes de connexion se sont révélées très décevantes. En effet, l'outil de test de bande passante ne dépassait pas les 5 Mbps, voire parfois était limité à quelques centaines de kbps seulement, très loin des 130 Mbps théoriques d'une liaison 802.11n utilisant un MIMO 2x2 et des bandes de 20 MHz. Après de nombreux tests en changeant divers paramètres, tous infructueux, nous avons réglé une des deux fréquences dans la bande des 2.4 GHz. Nous avons alors obtenu une liaison beaucoup plus robuste et les premiers tests montraient une bande passante disponible supérieure à 40 Mbps dans les deux sens.

Certes ces résultats sont très inférieurs à ceux d'une connexion Ethernet Gigabit, mais permettent à première vue d'envisager le passage de quasiment une trentaine de canaux échantillonnés à 48 kHz et codés sur 24

bits. Pour être exhaustif, nous avons alors réglé les deux fréquences dans la bande des 2.4 GHz, la liaison était alors à nouveau inopérante. Ces premiers tests ont permis d'attirer notre attention sur l'importance que peuvent avoir les interférences radios entre les différents circuits, la configuration retenue pour notre test étant loin d'être optimale de par la proximité des antennes. Ce matériel étant prévu à la base pour des liaisons à longue distance en extérieur, l'utilisation obligatoire d'antennes directives dans ce cas permet de limiter la pollution de la seconde liaison.

Afin de tester plus précisément notre liaison avec un canal de transfert en 2.4 GHz et un canal de transfert en 5 GHz, nous avons utilisé le logiciel iperf sur deux ordinateurs chacun relié à un des routeurs. iperf permet d'effectuer des tests de transfert de paquets UDP, en indiquant la bande passante désirée. Nous avons effectué tous nos tests en mode dual, c'est-à-dire avec des paquets transitant dans les deux sens de liaison. Les informations retranscrites par iperf sont de deux natures : d'une part le nombre de paquets perdus par rapport à la bande passante demandée, d'autre part la gigue de transmission, c'est-à-dire la variation de latence des paquets transmis. Ces deux critères sont primordiaux pour une liaison dédiée à un réseau AES67/Ravenna, la perte d'un paquet pouvant signifier la perte d'échantillons audio, et donc « trou » à la restitution, et la gigue pouvant mettre à mal la synchronisation, comme nous l'avons expliqué précédemment.

3.5.6 Tests audio et conclusions

Nous avons ensuite tenté de procéder à des transferts audio en reprenant notre configuration de test sans-fil, c'est-à-dire avec une Horus d'un côté et notre ordinateur équipé de JADE de l'autre. Si la synchronisation de JADE est effective assez rapidement, la qualité de l'audio transféré laisse à désirer. Si nous n'avons pas observé de perte totale de la liaison comme lors

des premiers tests, elle reste entâchée de « clics numériques » constants et réguliers. Ces résultats sont conformes à ce qu'on pouvait attendre au vu des résultats obtenus avec iperf. Ils sont toutefois quelque peu étonnants, car nous nous sommes limités à une simple liaison audio, qui ne demande qu'un débit de 2.4 Mbps, largement inférieur aux possibilités de notre configuration.

Comment expliquer ces résultats alors ? L'observation des indicateurs du logiciel WinBox nous donne quelques pistes : Il permet en effet d'observer le nombre de paquets dont le transfert a échoué et qu'il doit transférer à nouveau. Si ce processus échoue de trop nombreuses fois pour un même paquet, le transfert est tout simplement abandonné. De plus, même si le paquet est finalement transmis, un trop grand nombre d'essais engendre un retard dans son acheminement, et il peut alors arriver trop tard, c'est-à-dire après le *timestamp* auquel il devait être lu.Dans le cas d'une bonne liaison radio, ces valeurs restent nulles. Ce n'était pas du tout le cas pour notre liaison, ces deux valeurs augmentant constamment et assez rapidement. Cette observation nous pousse à mettre en doute la qualité des transferts radio de notre système : En effet, pour pouvoir profiter du MIMO 2x2 sur nos routeurs, nous avons du les équiper chacun de quatre antennes toutes réparties sur le pourtour des boîtiers de nos routeurs. Il y a donc seulement quelques centimètres entre les antennes émettrices et les antennes réceptrices. Celles-ci fonctionnant simultanément, les interférences sont nombreuses, et peuvent fortement influencer les qualités des connexions.

Si ces premiers résultats sont décevants par rapport aux espérances que nous permettait d'avoir une solution de liaison *full-duplex* sans-fil, il n'en reste pas moins qu'une telle liaison est fonctionnelle. Le temps imparti à nos recherches ne nous a pas permis de tester cette liaison dans des conditions optimales, et il serait souhaitable de tenter d'améliorer les performances radio de l'ensemble avant de pouvoir émettre un avis définitif sur cette solution. Pour

cela, nous envisageons de remplacer nos antennes actuelles par des antennes directives déportées des routeurs. Si leur but est au départ d'assurer des liaisons à longue distance, la prédominance du gain de l'antenne dans une direction nous permettra surtout de diminuer la pollution des différentes antennes entre elles, pollution d'autant plus réduite que nous pourrons écarter physiquement les antennes les unes des autres.

Annexe A

RVSC

A.1 Page d'accueil

Dans cette annexe, nous allons décrire l'interface Web de RVSC. Cette présentation peut servir à une personne découvrant ce logiciel, mais ne se veut en aucun cas une présentation exhaustive du logiciel. Elle est surtout pensée comme un support pour décrire les concepts de base de Ravenna. Il est à noter que cette interface Web est utilisée pour RVSC, mais aussi pour d'autres produits. Ainsi certaines options peuvent être désactivées car n'étant pas pertinantes pour RVSC. De plus, les autres interfaces Web que nous avons pu utiliser pour configurer les produits Ravenna de Merging présentent une interface très similaire à celle de RVSC et donc les concepts présentés ici s'appliquent pour ces produits également. Pour des informations plus détaillées, on pourra se référer au manuel de l'interface Web [22].

L'accès à l'interface Web se fait en se connectant au port 8080 de l'ordinateur hébergeant le logiciel. Nous accédons alors à une page de présentation résumant la configuration actuelle du logiciel. Sont ainsi résumés sur l'écran :

- La configuration actuelle de l'horloge

- Les flux audio émis par le pilote
- Les flux audio entrants auxquels le pilote est connecté

Un clic sur chacun de ces résumés nous amène à une présentation détaillée de chacun de ces aspects. De plus on trouve un dernier menu nous permettant d'accéder à certaines options, dont le choix des interfaces audio interfacées avec RVSC, des options de tests, et l'accès aux journaux créés par le logiciel.

A.2 Configuration de l'horloge

La page de configuration de l'horloge nous permet tout d'abord de connaître le statut du pilote par rapport au PTP, à savoir s'il est *grandmaster* ou *slave*. Elle dispose aussi d'étatd supplémentaires, qui indiquent si la synchronisation est en cours d'établissement ou si elle est défaillante.

On retrouve ensuite les différentes options que l'on peut régler concernant le PTP. On y trouve par exemple les valeurs des paramètres priority1 et priority2 du BMCA (*cf* sous-section 2.2.6). On a aussi accès au réglage de l'intervalle d'émission des messages Announce et Sync (*cf* sous-section 2.2.2). Enfin on peut choisir le mode de fonctionnement PTP, End-to-End ou Peer-to-Peer (*cf* sous-section 2.2.4).

A.3 Session source et Session sink

La gestion des flux audio par Ravenna est quelque peu différente des mécanismes traditionnelement utilisés. En effet, on a l'habitude avec les systèmes numériques, y compris avec des technologies comme Dante, de disposer d'une matrice avec toutes les entrées et sorties disponibles du système, et de réaliser ainsi les connections entre les différentes interfaces. Avec Ravenna,

on crée d'abord un flux de sortie sur l'appareil émetteur, puis sur chaque appareil récepteur, on récupère ce flux. Ces deux mécanismes sont respectivement dénommés Session source et Session sink.

Les différents paramètres lors de la création d'un Session source sont :

- Le nom
- L'adresse multicast
- La taille des paquets UDP
- Le nombre de canaux
- Les entrées audio, physiques ou logicielles, à transmettre

De l'autre côté, un Session sink se crée en sélectionnant directement dans la liste déroulante un des flux disponibles sur le réseau. On peut éventuellement rentrer une description supplémentaire du flux, et on sélectionne le nombre de canaux audio, ainsi que la sortie physique ou logicielle qui va être alimentée par ce flux. On peut enfin régler un paramètre de délai, réglé en nombre d'échantillons, qui sert à compenser les retards induits par le réseau.

Annexe B

PTP et Ravenna ASIO Driver

Les différents tests que nous avons pu mener tout au long de ce mémoire nous ont amené à faire des observations sur les différents logiciels et appareils utilisés. En particulier, nous avons pu nous rendre compte que le fonctionnement du pilote ASIO proposé par Merging s'écartait du fonctionnement que l'on attendait d'un appareil Ravenna. Cette annexe se propose de présenter les conclusions de ces observations.

Le premier écart entre le fonctionnement du pilote et le fonctionnement attendu nous est apparu lors de nos tests filaires initiaux. Une capture du trafic réseau nous a fait nous rendre compte que si les paquets PTP provenant de l'interface Horus était bien acheminés jusqu'à l'ordinateur équipé du pilote ASIO, aucun paquet Delay_Req n'est émis par cet ordinateur à destination de l'Horus, qui est pourtant le *grandmaster* du réseau PTP. Logiquement, il n'y a pas non plus de paquet Delay_Resp reçu par l'ordinateur. Pourtant nous étions dans la situation de fonctionnement normal des différents produits, et effectivement nous n'avons détecté aucun dysfonctionnement des liaisons audio. La question se pose alors de savoir comment la synchronisation est effectuée entre les deux appareils, puisque le mécanisme proposé par le PTP ne peut pas être mis en œuvre.

Le comportement du pilote lors des premiers tests sans-fil a été encore plus étonnant. Alors qu'une capture réseau montrait qu'aucune donnée PTP n'arrivait jusqu'à l'ordinateur, l'interface de gestion ASIO Pannel indiquait une synchronisation effective avec l'Horus. Même si les faits nous ont montré que les échanges audio étaient alors impossibles, il n'empêche que cela prouve que le mécanisme de synchronisation du pilote ne repose pas du tout sur le PTP, il n'exploite même pas les données de *timestamp* des messages Sync comme on aurait pu le penser après nos premières observations. À ce propos, le fait que les messages Sync et Follow_Up arrivent jusqu'à l'interface réseau de l'ordinateur s'explique par l'utilisation de liens filaires et la configuration de nos routeurs qui par défaut redistribue l'ensemble des messages multicast dans ce cas.

La dernière observation qui nous a permis de mieux comprendre le phénomène a été réalisée lors de nos tests sur la liaison *full-duplex*. Au début de nos tests, nous avons constaté qu'un débit de données constant de l'ordre de 1 Mbps était présent, provenant de l'Horus en direction de l'ordinateur, alors qu'aucun flux n'avait été créé sur l'Horus. Une capture réseau a permis de déterminer qu'il s'agissait de paquets UDP, que nous avions déjà observés lors de précédente capture, mais que nous avions toujours associé à des données audio transportées par le protocole RTP, celui-ci apparaissant aussi simplement sous le terme d'UDP dans les captures Wireshark. Pourtant ici il était clair que l'Horus n'émettait aucune donnée RTP sur le réseau. L'observation de ces trames nous a permis de constater qu'elle transportait deux valeurs incrémentées à chaque transmission, ce qui se rapproche du comportement d'un signal d'horloge.

Pour aller plus loin, nous nous sommes de plus aperçu que lorsque le pilote ASIO était lancé sur notre ordinateur, un flux présentant les mêmes caractéristiques était émis en sens inverse, à destination de l'Horus. La compa-

raison de ces deux flux a permis de remarquer que la seconde valeur transportée par le paquet allant vers l'Horus était immédiatement répétée dans le paquet allant de l'Horus vers le pilote ASIO. On peut donc imaginer un mécanisme de question/réponse continuel entre l'Horus et le pilote ASIO qui peuvent ainsi se transmettre des données temporelles. À titre indicatif, ce processus est répété environ 760 fois par seconde.

Le pilote ASIO proposé par Merging, même s'il peut lire et créer des données sur un réseau Ravenna, n'est donc pas un produit Ravenna à part entière. En effet, il ne peut accéder à ce réseau qu'à la condition qu'une interface Horus s'y trouve aussi. De plus, la plupart de nos recherches s'étant portées sur le transport de la synchronisation PTP, les solutions évoquées ne sauraient être compatibles avec ce pilote. Même si nos premiers tests faisaient apparaître le couple Horus/pilote ASIO comme étant la solution la plus performante, la recherche d'une compatibilité avec les autres produits Ravenna et AES67 plus largement nous pousse à essayer de ne pas utiliser ce produit dans de futurs tests.

Conclusion

À travers ce mémoire, nous avons cherché à savoir s'il était aujourd'hui possible ou non d'utiliser un réseau sans-fil IEEE 802.11 pour véhiculer les technologies des réseaux audionumériques en faisant un état des lieux de la situation et en essayant d'explorer les diverses solutions s'offrant à nous, en les mettant si possible en pratique. Si ces travaux ne s'achèvent pas sur une solution directement exploitable en production, il n'empêche que l'on peut espérer que cela soit le cas dans un avenir relativement proche. Nous avons pu constater que si les deux domaines des réseaux sans-fil et des technologies audio pour les réseaux avaient été jusqu'à présent développés séparément, les ponts entre les diverses technologies commencent à apparaître, au sein de standards comme l'IEEE 802.1AS ou l'IEEE 802.1aa. Ces standards sont très récents, tout comme le sont le sont la technologie Ravenna et le standard AES67. Comme on a pu le constater, il est difficile de trouver des implémentations de ces différents standards et même le nombre de produits Ravenna aujourd'hui commercialisés est encore assez restreint. Mais on peut espérer que ces problèmes se résolvent dans un futur proche. Surtout avec la demande de plus en plus forte du grand public pour le contenu multimédia en temps réel et en haute qualité, qui ne peut qu'apporter des solutions bénéfiques pour de futures utilisations professionnelles.

De plus, il est à noter que par les ressources limitées dont nous disposions, certaines solutions n'ont pas été exploitées au maximum de leurs

performances et il est possible qu'une solution fonctionnelle puissent être déjà mise en place aujourd'hui avec les produits du commerce. Par exemple la solution de liaison *full-duplex* de MikroTik présente de nombreux avantages et demande à être tester à nouveau avec des antennes différentes. On peut aussi penser à la synchronisation GPS qui propose une solution facile à mettre en œuvre en théorie, mais se heurte au coût de l'investissement initial. Cependant comme nous l'avons fait remarqué précédemment, ce coût est tout à fait raisonnable pour des sociétés audiovisuelles, et on ne peut qu'espérer que nos recherches puissent susciter l'intérêt de certaines d'entre elles et les incite à expérimenter ces solutions.

Au-delà de l'aspect sans-fil, ce mémoire a aussi été l'occasion d'explorer en détail la technologie Ravenna et les mécanismes associés. On a vu de plus que l'implémentation de Ravenna utilise les mêmes protocoles sous-jacents que d'autres, comme Dante ou encore Livewire, et la publication de l'AES67 devrait confirmer la tendance actuelle des fabricants audio à se tourner vers ces mêmes solutions, et on peut donc raisonnablement s'attendre à ce que les protocoles utilisés ici tels le PTP et le RTP fassent demain partie du quotidien des ingénieurs du son. Même si pour une configuration standard, la mise en œuvre d'un réseau audionumérique diffère peu d'une installation audionumérique classique, il est toujours bon de connaître en détail le fonctionnement des appareils que l'on utilise et ces considérations nous amènent à penser que des connaissances en ingénierie réseau feront partie demain des compétences à maîtriser par les professionnels du son, pour pouvoir identifier et corriger les éventuels dysfonctionnements d'une installation.

Une plus grande maîtrise des technologies réseau est aussi la clé de l'exploitation de tout leur potentiel pour des applications audiovisuelles. Par exemple nous avons au cours de nos recherches exploré les possibilités de routage multicast sur notre réseau. Même si nous n'avons finalement pas

abouti au résultat escompté, il est cependant intéressant d'imaginer comment ces techniques pourraient être utilisées dans des réseau audionumériques. Sur de très grandes installations par exemple, il est alors possible de choisir de diffuser ou non un flux à travers un équipement, et ainsi créer des zones de diffusions distinctes, tout en gardant tous les équipements reliés entre eux, et donc en permettant de créer selon les besoin de nouvelles connexions, ou une reconfiguration totale des liaisons, juste par une configuration logicielle différente.

Avec ou sans-fil, il est pour nous évident qu'une partie importante de l'avenir des technologies audiovisuelles se trouve dans la mise en place et le développement des technologies des réseaux informatiques.

Bibliographie

[1] **Audio Engineering Society, Inc.** *AES10-2008 : Recommended Practice for Digital Audio Engineering – Serial Multichannel Audio Digital Interface (MADI) (Revision of AES10-1991)*, 2008.

[2] **Audio Engineering Society, Inc.** *AES67-2013 : AES standard for audio applications of networks - High-performance streaming audio-over-IP interoperability*, 2008.

[3] **Audio Engineering Society, Inc.** *AES11-2009 : AES recommended practice for digital audio engineering - Synchronization of digital audio equipment in studio operations (Revision of AES11-2003)*, 2009.

[4] **Bálint Ferencz**. *Hardware Assisted IEEE 1588 Clock Synchronization Under Linux*. Master's thesis, Budapest University of Technology and Economics, 2013.

[5] **Geoffrey M. Garner**, **Michel Ouellette**, and **Michael Johas Teener**. *Using an IEEE 802.1AS Network as a Ditributed IEEE 1588 Boundary, Ordinary, or Transparent Clock*. ISPCS Conference, sep 2012.

[6] **Institute of Electrical and Electronics Enginners**. *IEEE Std 1588^{tm}-2008 - IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems*, 2008.

[7] **Institute of Electrical and Electronics Enginners**. *IEEE P802.11v/D13.0 - IEEE Draft Standard for Information Technology-Telecommunications and information exchange between systems-Local and*

Metropolitan networks-Specific requirements-Part II : Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications-Amendment 8 : IEEE 802.11 Wireless Network Management, jul 2010.

[8] **Institute of Electrical and Electronics Enginners**. *IEEE Std 1722^{tm}-2011 - IEEE Standard for Layer 2 Transport Protocol for Time Sensitive Applications in a Bridged Local Area Network*, 2011.

[9] **Institute of Electrical and Electronics Enginners**. *IEEE Std 1733^{tm}-2011 - IEEE Standard for Layer 3 Transport Protocol for Time-Sensitive Applications in Local Area Networks*, 2011.

[10] **Institute of Electrical and Electronics Enginners**. *IEEE Std 802.1AS^{tm}-2011 - IEEE Standard for Local and metropolitan area networks - Timing and Synchronization for Time-Sensitive Applications in Bridged Local Area Networks*, 2011.

[11] **Institute of Electrical and Electronics Enginners**. *IEEE Std 802.1BA^{tm}-2011 - IEEE Standard for Local and metropolitan area networks — Audio Video Bridging (AVB) Systems*, 2011.

[12] **Institute of Electrical and Electronics Enginners**. *IEEE Std 802.1Q^{tm}-2011 - IEEE Standard for Local and metropolitan area networks - Media Access Control (MAC) Bridges and Virtual Bridge Local Area Networks*, 2011.

[13] **Institute of Electrical and Electronics Enginners**. *IEEE Std 802.11aa^{tm}-2012 - IEEE Standard for Information technology–Telecommunications and information exchange between systems Local and metropolitan area networks–Specific requirements Part 11 : Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 2 : MAC Enhancements for Robust Audio Video Streaming*, 2012.

[14] **Institute of Electrical and Electronics Enginners**. *IEEE Std 802.11^{tm}-2012 - IEEE Standard for Information technology–Telecommunications and information exchange between systems Local and metropolitan area networks–Specific requirements Part 11 : Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, 2012.

[15] **Internet Engineering Task Force**. *Host Extensions for IP Multicasting*, aug 1989.

[16] **Internet Engineering Task Force**. *Network Time Protocol Version 4 : Protocol and Algorithms Specification*, jun 2010.

[17] **William Leveugle**. *Des possibilités d'évolution vers le sans-fil des réseaux audionumériques pour la sonorisation - Cas des Wireless Lan.* Master's thesis, École Nationale Supérieure Louis Lumière, 2012.

[18] **J. Kevin Rhee**, **Kyusang Lee**, and **Yonghwan Bang**. *Timestamp jitter consideration for 802.11n.* sep 2008.

[19] **Zhang Cui Zhi**. *Research of Clock Synchronization Over WIFI Networks.* Master's thesis, Zhejiang University, 2012.

[20] ALC NetworX. *Operating Principles, Draft 1.0*, 2011.

[21] ALC NetworX. *Technology Overview*, 2013. http ://ravenna.alcnetworx.com/technology/technology-overview.html.

[22] ALC NetworX. Ravenna *Web Interface for device and stream configuration - User Guide*, 2013.

Table des figures