Contribution à la modélisation et à la détection d'anomalies du traffic Internet à partir de mesures d'un coeur de réseau opérateur

Quentin Grandemange

UNIVERSITÉ DE LORRAINE

École doctorale IAEM Lorraine

Contribution à la modélisation et à la détection d'anomalies du trafic Internet à partir de mesures d'un cœur de réseau opérateur

THÈSE

présentée et soutenue publiquement le 6 Avril 2018

pour l'obtention du

Doctorat de l'Université de Lorraine

(mention Automatique, Traitement du signal et des images, Génie informatique)

par

Quentin Grandemange

Composition du jury

Examinateurs :	Annie Gravey	Professeur de l'Institut Mines-Télécom Atlantique
	Francis Lepage	Professeur de l'Université de Lorraine
Rapporteurs :	Rachid Malti	Professeur de l'Université de Bordeaux
	Jean-Marc Thiriet	Professeur de l'Université Grenoble Alpes
Directrice :	Marion Gilson-Bagrel	Professeur de l'Université de Lorraine
Invités :	Olivier Ferveur	Docteur, Post Luxembourg
	Éric Gnaedinger	Maître de conférences de l'Université de Lorraine

CRAN - Centre de Recherche en Automatique de Nancy - UMR 7039

Mis en page avec la classe thesul.

Remerciements

Je souhaiterais commencer ce mémoire en citant toutes les personnes, qui, de près ou de loin, m'ont aidé à mener à bien cette thèse durant ces trois années.

Tout d'abord, je tiens à remercier chaleureusement la directrice de cette thèse, Mme Marion Gilson-Bagrel, Professeur à l'Université de Lorraine, pour m'avoir formé durant mon cycle ingénieur et m'avoir donné le goût à la recherche pour ensuite m'encadrer durant ma thèse. Elle a su rester à l'écoute et m'a énormément apporté durant ces trois années.

Mes remerciements vont aussi à M. Olivier Ferveur, docteur de l'Université de Lorraine, qui a su m'intégrer au sein de Post Luxembourg en me faisant confiance pour l'accomplissement de ce projet.

Je remercie également M. Éric Gnaedinger, Maître de Conférences de l'Université de Lorraine, qui a su éveiller mon intérêt envers le monde des télécoms en me formant durant mon cursus ingénieur et qui m'a beaucoup aidé durant ces trois années.

Ce travail n'aurait pas été possible sans la participation de l'entreprise Post Luxembourg qui m'a accueilli au sein de ses locaux et permis de collecter de précieuses données tout en me laissant publier mes résultats sereinement.

Je remercie également le Centre de Recherche en Automatique de Nancy grâce à qui j'ai pu mener à bien le déroulement de ma thèse. Et plus particulièrement Mme Sabine Huraux et son équipe pour toute l'aide administrative qu'elles m'ont fournie.

Je souhaite remercier le Professeur Francis Lepage pour avoir participé au suivi de mes travaux durant ces trois ans et d'avoir accepté de faire partie de mon jury en tant qu'examinateur.

Je suis très reconnaissant aux professeurs Rachid Malti et Jean-Marc Thiriet pour avoir accepté d'être les rapporteurs de mes travaux, ainsi qu'au professeur Annie Gravey pour sa participation au jury en qualité d'examinatrice.

Ce travail n'aurait pu être réalisé sans l'aide de mes collègues de Post Luxembourg. Je remercie tout particulièrement M. Pierre Scholtes, M. Carlo Richartz et M. Frédéric Mazzucotelli pour l'accueil et la confiance qu'ils m'ont accordés pour intégrer leur équipe.

De même, je souhaite remercier M. Michel Albert, M. Thomas Kirsch et M. Frank Lazzarini pour leur aide précieuse concernant la logistique et le développement de la solution proposée, ainsi que M. Sebastien Lourdez, M. Jo Hoffmann et M. David De Morais ainsi que tous les membres des équipes que j'ai côtoyés pour leur aide et leur bonne humeur tout au long de ce projet.

Je ne peux pas passer à côté de Yusuf Bhujwalla, doctorant puis docteur, avec qui j'ai partagé une partie de ces trois années et qui m'a aidé à cerner la modélisation des

systèmes, même si ce n'était pas gagné d'avance.

Je souhaite bonne chance à mon jeune apprenti doctorant, Constant Colombo, dans la suite de son épopée palpitante.

Je tiens à remercier mes amis qui ont eu à me supporter pendant ma thèse, qui m'ont soutenu moralement et qui ont dû relire de nombreuses fois mes écrits et ont toujours su y apporter des éclaircissements bienvenus.

Comment ne pas penser au piment qu'ont rajouté les deux sociétés gestionnaires des chemins de fer française et luxembourgeoise, à mon quotidien. Cependant, sans tout cela, je n'aurais pas rencontré le groupe de personnes à la bonne humeur communicative qui partage mes aventures lors de ces, parfois trop longs, trajets !

Il m'est impossible de finir ces remerciements sans penser à la personne qui partage ma vie et qui a dû supporter mes périodes de rush et les nombreuses relectures d'un sujet pas toujours évident à cerner. J'espère pouvoir te rendre la pareille lors de la finalisation de ta thèse !

Pour finir, je tiens tout particulièrement à remercier mes parents et ma famille qui ont toujours cru en moi durant toutes ces années et qui m'ont toujours poussé à donner le meilleur de moi-même. Merci pour tout.

Je dédie cette thèse
à mes parents et à ma famille.

Sommaire

Table des figures

Liste des tableaux

Glossaire

Adresse IP : Adresse logique servant à identifier un appareil sur un réseau

Adresse MAC : Identifiant physique unique d'une carte réseau d'un appareil

ARPAnet : Premier réseau de transfert de paquets développé aux États-Unis, souvent cité comme ancêtre d'Internet

AS : « Autonomous System », ensemble de réseaux informatiques intégrés à Internet, dont la politique de routage interne est cohérente, appartenant à une entité

BGP : « Border Gateway Protocol », protocole servant à échanger des préfixes IP sur Internet permettant le routage

CDN : Réseaux de diffusion de contenus ou de données à destination des utilisateurs

DARPA : Agence du département de la Défense des États-Unis chargée de la recherche et développement des nouvelles technologies destinées à un usage militaire

DDoS : Attaque informatique visant à saturer un service avec de fausses requêtes afin d'en bloquer l'accès pour les requêtes légitimes

DHR : « Dynamic Harmonic Regression » est une méthode de modélisation spécifiquement dédiée aux séries temporelles périodiques

Estimation d'un modèle : Phase pendant laquelle les paramètres du modèle sont adaptés afin de refléter au mieux le système. On parle aussi de phase d'entraînement du modèle

FAI : Fournisseur d'Accès à Internet

Gigue : Représente la variation de la latence au fil du temps

GRT : Table contenant la liste de tous les réseaux accessibles sur Internet

IP : « Internet Protocol », famille de protocole permettant de communiquer sur Internet

IXP : Point d'inter-connexion d'AS au niveau d'Internet

Latence : Délai de transport entre la source et le destinataire d'un paquet

Medium : Moyen par lequel est transmis une information

Modèle OSI : Standard de communication décrivant les fonctions de communication et leur organisation

Netflow : Netflow est un protocole de collecte d'informations de flux de trafic IP

Peering : Fait, pour deux AS, de se connecter directement afin de réduire les coups de transit

Plage IP : Une plage IP, appelée aussi un préfixe, est une liste d'adresses IP successives partageant une racine commune

Préfixe : Voir « Plage IP »

Processus Gaussien : Méthode probabiliste et statistique pour le machine learning se basant sur des noyaux gaussiens

RIR : Registre Internet Régional, organisme allouant des blocs d'adresses IP aux AS

Route : Elle définit le chemin que doit suivre le trafic pour aller de la source au destinataire

Routeur PE : Les routeurs « Physical Edge » sont situés à la frontière du réseau et représentent les points d'entrée/sortie du réseau par rapport à Internet

Série temporelle : Une série temporelle est une suite de valeurs numériques représentant l'évolution d'une quantité au cours du temps

SLA : « Service-Level Agreement » est un contrat définissant la qualité de service de la prestation d'un opérateur à un client

SNMP : « Simple Network Management Protocol » est un protocole de communication servant à gérer et superviser des équipements réseaux

Tier : Hiérarchie d'AS sur Internet. Les Tier-1 fournissent une connectivité mondiale aux Tier-2, qui fournissent eux-même cette connectivité aux Tier-3, majoritairement composés de FAI

Trafic interdomaine : Définit la catégorie de trafic traversant plusieurs AS

Transit : Possibilité, pour une AS, de transférer de données afin d'accéder à une destination via des AS d'inter-connexions

Validation d'un modèle : Phase pendant laquelle les prédictions d'un modèle sont validées par rapport à des données n'ayant pas servi à estimer le modèle

Chapitre 1

Introduction

Sommaire

Internet occupe aujourd'hui une place prépondérante dans nos vies. Que ce soit, entre autres, pour gérer notre compte en banque, notre cercle social, nos emplois du temps et même notre chauffage domestique. Cependant, son fonctionnement reste, pour la plupart des personnes, totalement flou, comme un nuage qu'on pense omniprésent et pouvant réaliser tout ce qu'on lui demande. On a tendance à oublier que des rouages régissent ce fonctionnement. Et souvent, nous n'avons aucune emprise dessus, comme l'indique Tristan Nitot [1] :

> Le Cloud c'est l'ordinateur de quelqu'un d'autre. C'est juste un joli mot qui raccourcit et qui donne un côté vaporeux et sympathique, mais, littéralement, le Cloud c'est un ordinateur, c'est l'ordinateur de quelqu'un d'autre.

Ce travail ne prétend pas répondre aux problématiques liées à la sécurité et à la vie privée, deux sujets très importants qui ont déjà été traités par des personnes beaucoup plus documentées en la matière, mais de montrer que ce Cloud a des failles dues à l'utilisation d'Internet (congestion, censure...) et qu'il est possible d'en tenir compte en réagissant rapidement afin d'en limiter les impacts.

1. Actuellement Chief Product Officer au sein de la société Cozy Cloud. Anciennement fondateur et président de Mozilla Europe.

L'étude proposée dans ce document a été réalisée grâce à la collaboration entre deux entités : le Centre de Recherche en Automatique de Nancy (CRAN) et la société Post Luxembourg, opérateur et fournisseur d'accès à Internet luxembourgeois.

Dans un contexte d'augmentation permanente du trafic international entre les réseaux qui composent Internet [1] d'environ 40% tous les ans, Post Luxembourg souhaite obtenir une vue claire et pertinente de ses échanges avec les réseaux extérieurs. En effet, une conséquence à cette augmentation exponentielle est une augmentation du trafic parfois plus rapide que les liens par lesquels il transite. Cela peut donc donner lieu à des saturations et des congestions sur certains liens, y compris sur des liaisons internationales interopérateurs, dégradant de fait l'expérience d'utilisation des clients finaux.

C'est pour détecter ces phénomènes interopérateurs et pour améliorer le ressenti client que ces travaux ont été menés. Une mesure du trafic a été mise en place afin de modéliser le comportement des échanges entre le réseau de Post Luxembourg et ses principaux voisins. Ces modèles permettent de détecter, en quasi temps réel, des anomalies de trafic et de prévenir les équipes afin de prendre des contremesures adéquates.

Dans un premier temps et afin de bien poser la problématique des travaux présentés dans le manuscrit, nous allons revenir sur ce qu'est Internet, et comment cela fonctionne.

1.1 D'ARPAnet à Internet

À la fin des années 1930, les premiers ordinateurs, programmables ou non, sont créés afin de résoudre des tâches simples. Dix ans plus tard, leur nombre grandissant, des premières recherchent sont réalisées afin de permettre la communication entre plusieurs de ces machines. En 1958, le premier modem permettant une communication de données sur une liaison téléphonique est créé par les laboratoires Bell, ce qui ouvre la porte à de nombreuses recherches sur l'utilisation de ce médium [2].

C'est le DARPA (acronyme anglais de « Defense Advanced Research Projects Agency » pour « agence pour les projets de recherche avancée de défense ») américain qui établit une première étape avec son réseau de communication nommé ARPAnet (acronyme anglais de « Advanced Research Projects Agency Network »), considéré aujourd'hui comme l'ancêtre d'Internet. Celui-ci relie d'abord des universités des États-Unis et s'étend ensuite au reste du monde en 1973 avec l'Angleterre et la Norvège comme indiqué à la figure 1.1. Il est constitué de deux types de nœuds :

Terminal Interface Processor : les TIP représentent les nœuds terminaux du réseau où les utilisateurs peuvent se connecter aux autres nœuds.

Interface Message Processors : les IMP sont des nœuds de commutation de paquets qui permettent l'interconnexion des différents réseaux participant au projet ARPAnet. Ils sont les ancêtres des routeurs actuels.

Ce n'est qu'en 1983 que le nom Internet (pour « Inter-Network ») est adopté. Depuis

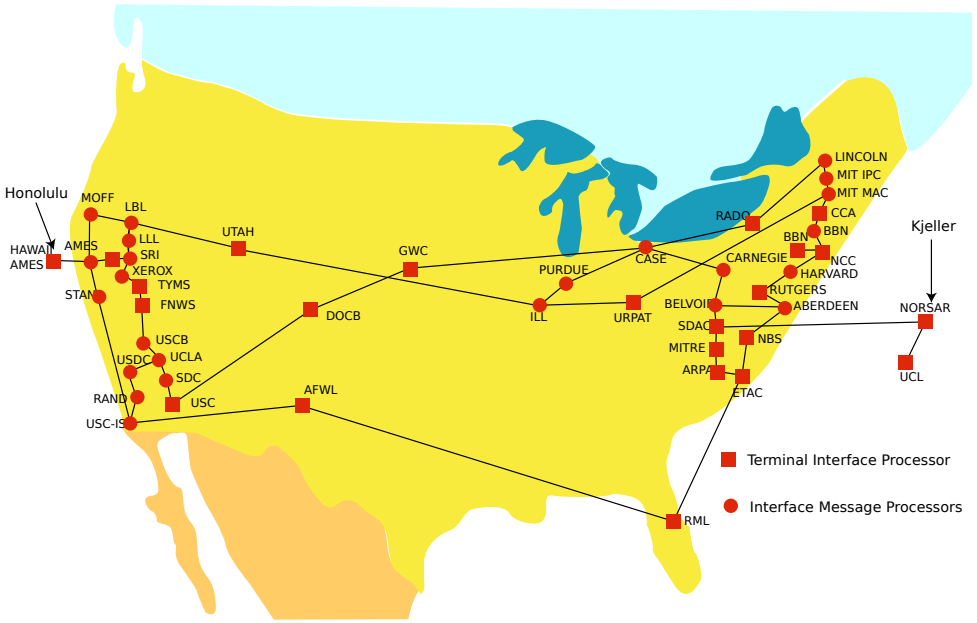


FIGURE 1.1 – Plan du réseau ARPAnet en 1974 [2]

lors, le nombre d'ordinateurs connectés à Internet a été multiplié par plus de 10 tous les 4 ans pour atteindre presque 370 millions d'ordinateurs connectés en 2000. En 2015 on dénombrait plus de 15 milliards d'appareils connectés à Internet, et les prédictions s'élèvent à 75 milliards pour l'année 2025.

1.2 Fonctionnement d'un réseau

L'information que l'utilisateur (humain ou machine) veut envoyer sur le réseau transite sous forme de paquets. Les informations échangées sur un réseau par les utilisateurs (humain ou machine) sont traduites sous forme de valeurs logiques (0 ou 1) appelées "bits". Une information est codée sur plusieurs bits, généralement sur un multiple de 8 bits ou "octet". Cette information est généralement accompagnée de données protocolaires. On parle alors d'unité de données (PDU). Nous nous intéresserons aux PDU de couche 3 du modèle OSI [3] représenté à la figure 1.2 et appelés paquets. Ces paquets peuvent être de taille variable et sont souvent limités par la configuration du médium par lequel ils transitent (la plupart du temps dans les réseaux Ethernet autour de 1500 octets). De fait, certaines informations sont fragmentées en plusieurs paquets qui seront ré-assemblés une fois arrivés au destinataire.

Ces données sont créées par une application puis sont transférées au sein de l'ordinateur vers les couches basses de communication, chaque couche ajoutant des informations

2. Tiré de `https://commons.wikimedia.org/wiki/File:Arpanet_1974.svg`, par Yngvar.
3. Open Systems Interconnection

autour des données brutes afin que l'ordinateur du destinataire sache à quelle application correspondent les données reçues. Ces éléments sont définis via un modèle en couche, comme par exemple le modèle appelé OSI représenté à la figure 1.2.

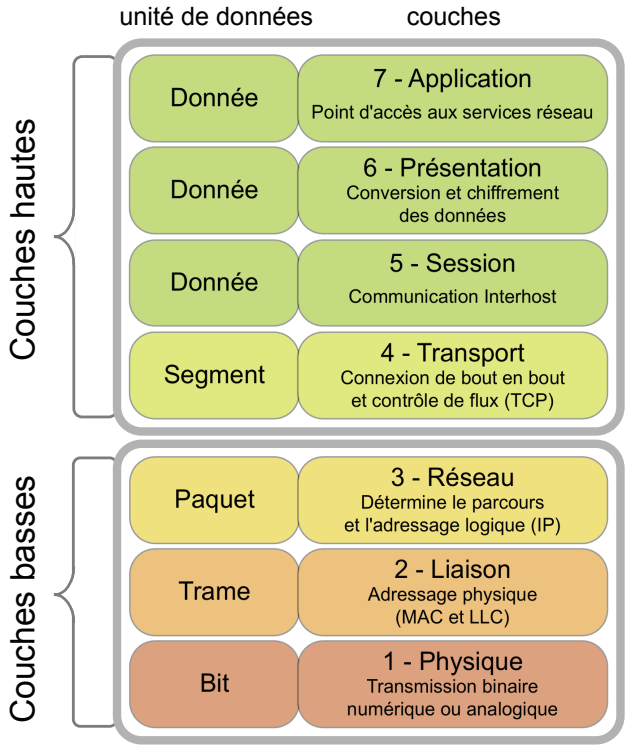


FIGURE 1.2 – Modèle OSI [4]

Au niveau d'un réseau informatique, ce sont surtout les couches 2 et 3 qui entrent en jeu. Nous ne nous intéressons pas aux couches hautes des paquets qui définissent plus particulièrement l'application utilisée. Quant à la couche 1, elle est définie par le médium physique qui n'entre pas en compte dans ce que nous voulons montrer.

Un ordinateur (ou tout autre élément de terminaison de réseau) a un double adressage qui correspond aux couches 2 et 3. La première, l'adresse MAC (« Media Access Control ») correspond à l'adresse physique de la carte réseau et est utilisée pour communiquer sur un réseau local. Celle-ci est unique dans le monde, elle contient une partie correspondante au fabricant de la carte et une seconde aléatoire sur une taille de 48 bits.

La seconde adresse, l'adresse IP (« Internet Protocol ») donne une adresse logique à une interface réseau. Elle est définie pour communiquer sur de plus longues distances et sur des réseaux de plus grande envergure que les réseaux locaux habituels. Il en existe de plusieurs catégories. Nous verrons ici surtout des IP dites publiques qui sont attribuées par les fournisseurs d'accès à internet (FAI) à leurs clients. Un autre type que l'on rencontre couramment est l'adressage privé. Ce sont des plages précises qui correspondent à des

4. Tiré de `https://commons.wikimedia.org/wiki/File:OSI_Model_v1.svg`, par Offnfopt.

adresses n'étant pas distribuées sur Internet et réservées à un usage local.

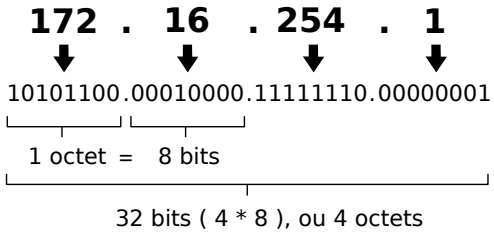


FIGURE 1.3 – Nomenclature d'une adresse IPv4 [5]

Les adresses IP de version 4 (IPv4) se définissent sous la forme $w.x.y.z$ où chaque bloc est constitué de 8 bits (ou 1 octet) (voir figure 1.3). La plage s'étend donc de 0.0.0.0 à 255.255.255.255 avec des limitations quant à l'usage de certaines plages d'adresses IP. Des adresses IP de version 6 (IPv6), de taille plus importante, sont aussi déployées notamment pour pallier le manque d'adresses IPv4 disponibles. Il n'est plus question de 4 blocs de 8 bits, mais de 8 groupes de 16 bits faisant passer le total d'IP disponibles de 2^{32} pour IPv4 (soit environ 4,3*10^9) à 2^{128} pour IPv6 (soit environ 3,4*10^{38}). Pour comparaison, le désert du Sahara compterait environ 10^{23} grains de sable.

Quelle que soit la version de l'adressage IP utilisée, le fonctionnement reste le même. En effet, des organismes appelés Registres Internet Régionaux (RIR) sont en charge de distribuer des plages d'adresses IP (appelées aussi préfixes réseaux) aux opérateurs ou entreprises. Si cette entité détient une connectivité redondante avec Internet via des opérateurs distincts (on parle alors de réseau « multi-homed »), elle peut faire la demande afin d'obtenir un numéro de système autonome (noté AS pour « Autonomous System ») [6].

1.3 Fonctionnement d'Internet

Internet est une interconnexion de nombreux réseaux informatiques, et plus particulièrement, d'AS. Début 2015, le nombre d'AS visibles sur Internet approchait les 50 000. Ce nombre est en constante augmentation comme le montre la figure 1.4, avec environ 240 nouveaux AS chaque mois [3]. La totalité des préfixes partagés, ou « routés », sur Internet est appelée la table de routage globale, ou « GRT » (pour « Global Routing Table »). Cette table est constituée, fin 2017, d'environ 650 000 préfixes IPv4 et 44 000 préfixes IPv6. Cette table est également en constante augmentation (voir figure 1.5).

Pour permettre le transfert de communications entre les AS, on parle alors de routage « interdomaine ». Chacun doit annoncer les préfixes qu'il possède à ses voisins. Mais, plus le nombre d'AS augmente, plus il est difficile pour les AS d'avoir une interconnexion directe avec chacun des autres AS [4]. C'est pourquoi Internet a été construit via une

5. Tiré de `https://commons.wikimedia.org/wiki/File:Addresse_Ipv4.svg`, par Star Trek Man.

6. Dans la suite du document nous utiliserons l'acronyme anglais AS classiquement utilisé dans la littérature des réseaux pour système autonome.

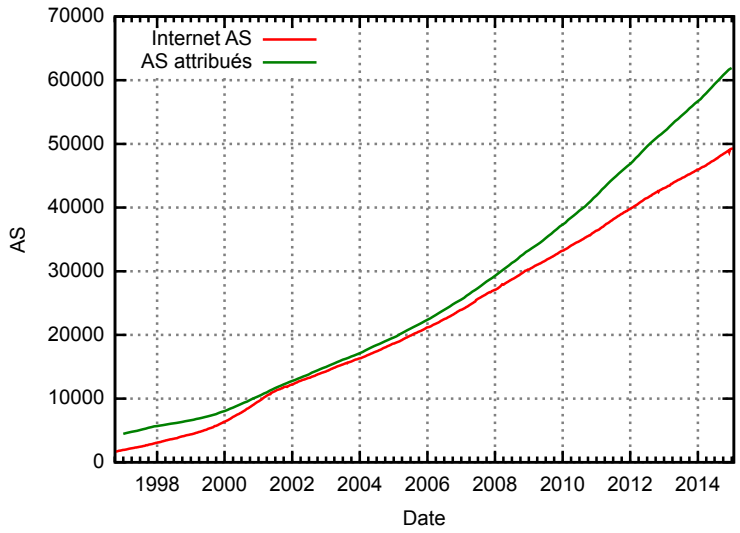


FIGURE 1.4 – Nombre d'AS sur Internet [7]

architecture en trois étages. Situés à l'étage le plus haut, les AS appartenant aux Tier-1 ont pour rôle de servir de pont d'interconnexion entre tous les AS. Ils forment la colonne vertébrale d'Internet et sont souvent méconnus du grand public.

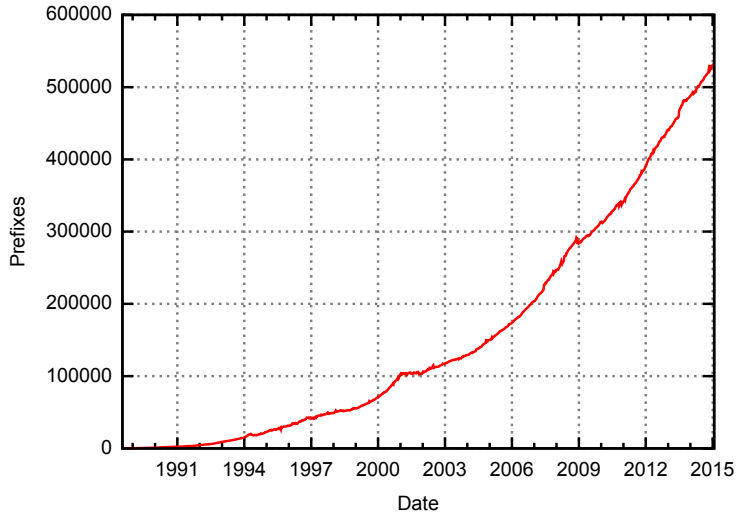


FIGURE 1.5 – Nombre de préfixes IPv4 sur Internet [8]

De l'autre côté, les Tier-3 sont chargés d'amener la connexion jusqu'au client final. C'est souvent cette catégorie qui est la plus connue car c'est par eux que le grand public accède au service qu'il souhaite. Ce groupe comporte aussi bien des FAI [9] que des réseaux de diffusion de contenu (ou CDN [10] pour « Content Delivery Network »).

7. Tiré de `https://commons.wikimedia.org/wiki/File:BGP_Table_growth.svg`, par Mro.
8. Tiré de `https://commons.wikimedia.org/wiki/File:Internet_AS.svg`, par Mro.
9. Exemple de FAI français : Orange, Free...
10. Exemple de CDN : Google, Netflix...

Entre ces réseaux à faible échelle et la colonne vertébrale d'Internet, il y a une troisième catégorie faisant la jonction entre les deux autres. Ce sont les Tier-2. Ils ont souvent un réseau plus étendu que les Tier-3 et peuvent fournir du transit à plusieurs Tier-3. Ils peuvent, comme les Tier-3, avoir des clients finaux.

Les AS peuvent, soit se connecter verticalement aux Tiers supérieurs et inférieurs dans des points de présence (appelés "PoP" pour « Point of Presence »), soit participer à des points d'échange Internet [5] (ou IXP pour « Internet Exchange Point »). Ces derniers sont des lieux sécurisés où des centaines, voire des milliers, d'AS viennent partager un équipement réseau, souvent loué par l'entreprise gérant l'IXP, afin de partager une connexion. Dans ces IXP, historiquement créés par des universités, plusieurs AS de même Tier ou non peuvent se connecter directement afin de faire communiquer leur trafic directement vers la destination.

1.4 Liaisons de transit et de peering

Chaque AS, pour avoir accès à l'ensemble d'Internet (c'est-à-dire à la totalité des préfixes), peut faire « transiter », en payant en fonction du débit, son trafic via un (ou plusieurs) Tier-1, celui-ci (ou ceux-ci) se chargeant de transférer le trafic vers le bon AS. Il se peut que celui-ci doive transiter via plusieurs AS différents avant d'arriver à destination.

La figure 1.6 représente un exemple de schéma de principe de liaisons de transit pour un AS.

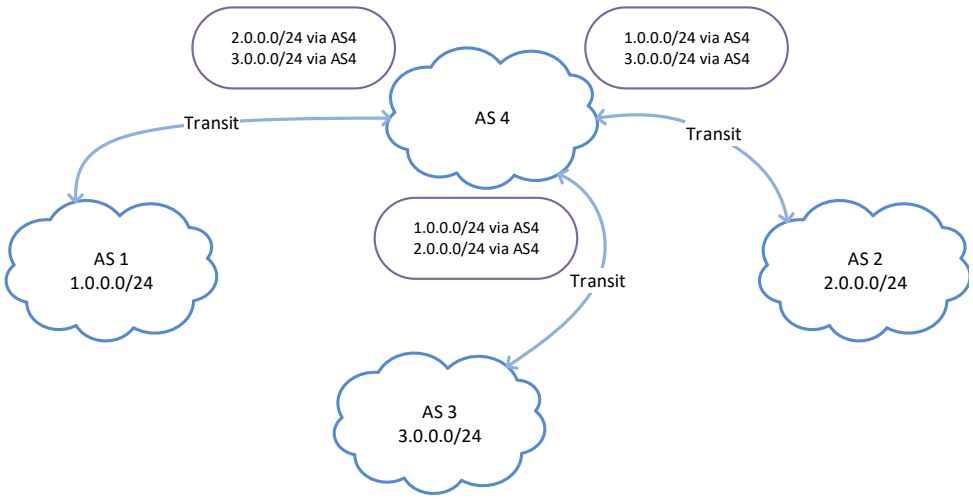


FIGURE 1.6 – Schéma de principe de liaisons de transit

On appelle ces liaisons, qui fournissent une connectivité totale à un Tier inférieur via un Tier supérieur, des liaisons de transit. Le Tier supérieur s'engage à fournir un accès à

l'ensemble d'Internet à ses AS clients. Une AS peut acheter plusieurs liaisons de transit afin de redonder sa connexion.

Ces annonces de routes Internet sont faites via un protocole bien particulier qui permet d'envoyer une liste de destinations ainsi que la liste d'AS traversé(s) pour chacune de ces destinations (Voir table 1.1).

Destination	Chemin d'AS	Route envoyée à
1.0.0.0/24	AS 4, AS 1	AS 2 et AS 3
2.0.0.0/24	AS 4, AS 2	AS 1 et AS 3
3.0.0.0/24	AS 4, AS 3	AS 1 et AS 2

TABLE 1.1 – Exemple de table de routage

En prenant exemple sur la figure 1.6, nous avons trois AS (1, 2 et 3), ayant chacun un préfixe IP, connectés via une liaison de transit à un AS 4. Les 3 AS clients envoient leur route à l'AS de transit afin d'avertir Internet que ces IP sont joignables au sein de son AS. L'AS de transit envoie les routes apprises aux autres AS (compilées dans la table 1.1). Si un client de l'AS 1 veut se connecter à un client de l'AS 2, il doit passer par l'AS 4 pour y accéder (voir figure 1.7).

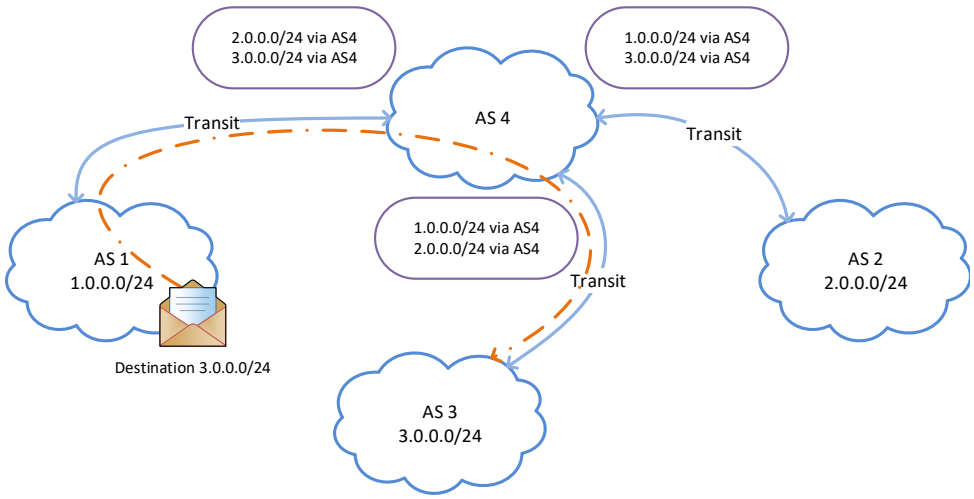


FIGURE 1.7 – Fonctionnement de liaisons de transit sur Internet

Si plusieurs routes sont possibles pour atteindre la destination, un algorithme propre au protocole de routage interdomaine est utilisé ; il s'agit du protocole BGP pour « Border Gateway Protocol ». Il définit la route à utiliser lorsque plusieurs choix sont possibles, et ce en parcourant l'arbre de décisions ci-après et en s'arrêtant s'il ne reste plus qu'une route :

Weight : la route ayant le poids le plus élevé est choisie. C'est un paramètre local défini par l'administrateur ;

Local_Pref : la route ayant la *local_pref* la plus élevée est choisie. C'est un paramètre local défini par l'administrateur ;

Self-Originated : les routes originaires de ce routeur sont préférées à des routes distantes ;

AS_PATH : la route traversant le moins d'AS est préférée ;

ORIGIN : une route apprise via un protocole de routage interne est préférée à une route apprise via un protocole de routage externe ;

MULTI_EXIT_DISCRIMINATOR : la route ayant la *MED* la plus faible est choisie. C'est un paramètre transitif défini par le réseau voisin ;

External : une route venant de l'extérieur est préférée à une route interne ;

IGP Cost : le poids de la route vers le *next-hop* le plus faible est choisi ;

eBGP Peering : la route la plus ancienne est préférée ;

Router ID : le routeur ID le plus faible est préféré.

Cependant aucun paramètre de qualité n'est pris en compte. En effet, la latence du lien ou la congestion du réseau n'entrent en rien dans le choix de la route préférée. Le trafic est alors toujours envoyé via un AS voisin et une congestion peut survenir sur le chemin sans que cela soit visible par l'AS source. C'est un des phénomènes que nous souhaitons révéler avec notre étude.

Une autre solution utilisée au sein des IXP consiste en l'interconnexion directe des différents participants sans passer par des Tiers supérieurs. Dans ce cas précis, seul le trafic direct entre ces deux AS passe sur ce lien. C'est ce qu'on appelle une liaison d'appairage (ou « peering »).

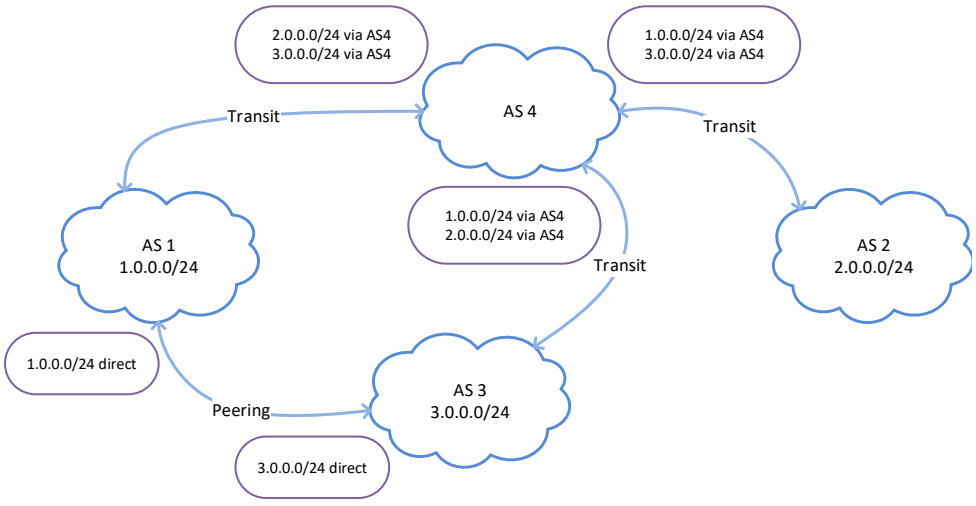


FIGURE 1.8 – Schéma de principe de liaisons de peering

Dans le cas d'une liaison de peering représenté à la figure 1.8, les deux participants s'accordent sur la bande passante et les paramètres du lien entre leurs deux réseaux.

De fait, un échange gratuit, contrairement au transit, peut s'effectuer directement entre ces deux AS (voir figure 1.9). Ces connexions directes entre les AS permettent ainsi de réduire les aléas rencontrés lors d'un routage classique. Elles offrent aussi généralement de meilleures performances concernant les métriques réseaux (latence, gigue...) car la distance et le nombre d'équipements traversés sont moins importants.

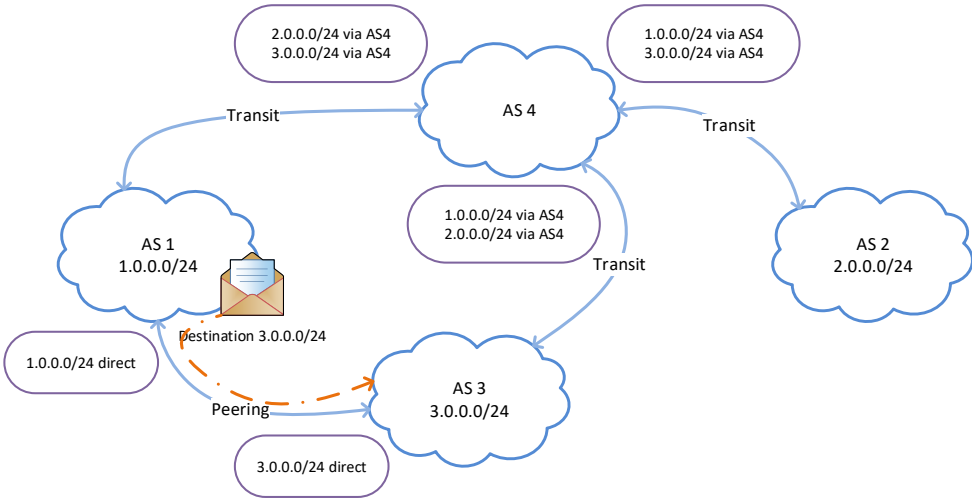


FIGURE 1.9 – Fonctionnement de liaisons de peering sur Internet

1.5 Contexte

Les données utilisées dans ces travaux ont été collectées sur le réseau de Post Luxembourg, un opérateur luxembourgeois. Cet opérateur possède des types de clients variés :

— environ 80 000 clients résidentiels, raccordés en cuivre (50%) ou fibre optique (50%) ;
— des clients professionnels, à haute valeur ajoutée et à haut niveau de technicité ;
— d'autres opérateurs locaux ;
— des clients de solution d'hébergement.

Cette diversité de solutions proposées fait de Post Luxembourg un opérateur Tier-2 du point de vue d'Internet. Pour répondre au mieux aux demandes de ses clients, Post Luxembourg a développé un réseau européen de points de présence au sein d'IXP de grandes envergures : Amsterdam (« AMS-IX »), Francfort (« DE-CIX »), Paris (« France-IX »), Londres (« LINX »), Bruxelles (« BNIX ») et Luxembourg (« LU-CIX »).

Au sein de ces emplacements, le réseau de Post Luxembourg est connecté à six Tier-1, recevant donc six fois la table globale Internet, et est appairé (*peeré*) avec 1400 réseaux voisins. En sommant le nombre de routes apprises via toutes les sources, Post Luxembourg apprend 7,1 millions de routes. En comparaison avec la table de routage globale, qui représente l'intégralité d'Internet, composée de 670 000 routes, on remarque que l'opérateur, de par la redondance de certaines routes, a un degré de liberté non négligeable quand il

s'agit de guider ces flux entrants et sortants de son réseau.

Par ces points d'entrée et sortie du réseau (voir figure 1.10), Post Luxembourg génère environ 150 Gbps de trafic répartis de manière équivalente entre transit et peering. Ce trafic est représenté sur une période de deux jours à la figure 1.11. On observe qu'une majorité du trafic entrant vient de liaisons de peering. Le comportement des clients étant stable, l'opérateur peut prendre des mesures afin de se connecter directement avec les réseaux dont Post Luxembourg est consommateur de données. Contrairement à cela, le trafic sortant est très dépendant des clients des datacenters et donc, il est difficile de prévoir les flux sortants.

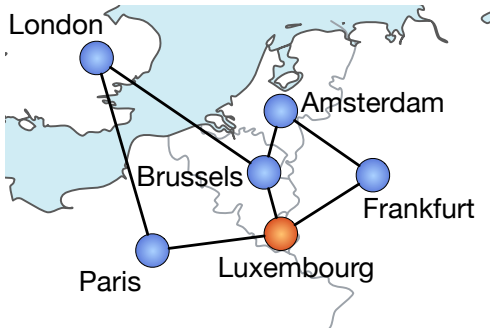


FIGURE 1.10 – Carte du réseau international de Post Luxembourg

Cette symétrie de trafic visible à la figure 1.11 est quelque chose de peu courant pour un opérateur. En effet, les FAI ont généralement un trafic majoritairement entrant. Les clients de l'opérateur consomment du trafic mais n'en génèrent que très peu vers l'extérieur. A l'opposé de ces FAI, les réseaux de type CDN, qui fournissent des données sortantes pour les consommateurs, n'ont que peu de trafic entrant. Ici, la situation particulière est due au fait que Post Luxembourg fournit une solution d'hébergement à différents clients générant du trafic sortant du réseau. Post Luxembourg est donc un AS hybride. En effet, il est à la fois un FAI et un CDN.

1.6 Une analogie du réseau informatique

Pour tout résumer en une analogie, prenons l'exemple du courrier. Votre nom et celui du destinataire de votre lettre correspondent aux adresses physiques du paquet (adresse MAC). Si vous souhaitez envoyer un courrier à quelqu'un qui n'est pas dans votre entourage direct, votre réseau local, vous allez inscrire son adresse postale, ici correspondant à son adresse IP, en plus de son nom sur l'enveloppe et l'envoyer via la Poste. Celle-ci va se charger de « router » votre courrier jusqu'à son destinataire, qu'importe le chemin. Une fois à la bonne adresse, le facteur va regarder le nom sur l'enveloppe pour trouver la bonne boîte aux lettres.

Voyons maintenant l'envoi de courrier à l'international. Si le courrier a une adresse de destination correspondant à un autre pays, la poste locale va envoyer cette lettre à

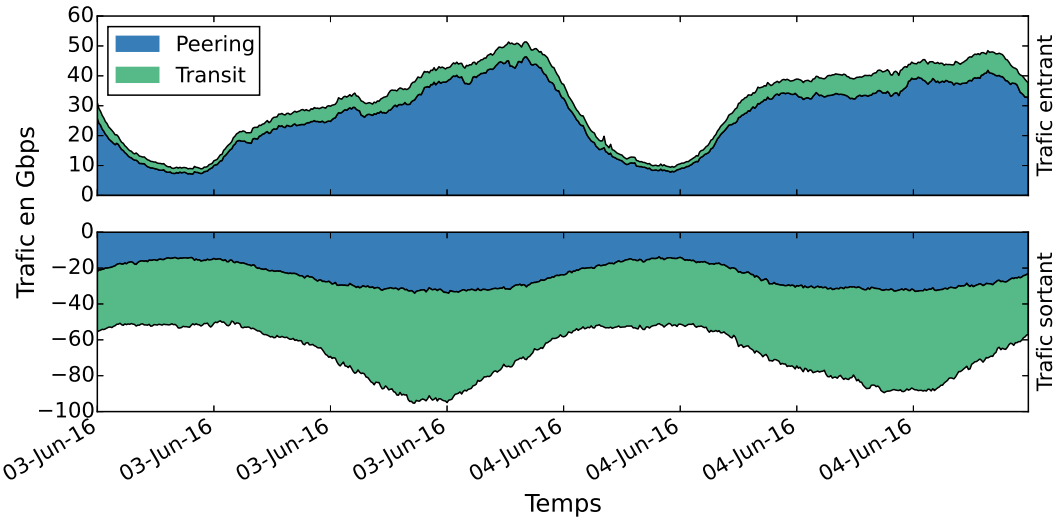


FIGURE 1.11 – Distribution cumulative du trafic de Post Luxembourg entre transit et peering (+ entrée / - sortie)

l'organisme de poste du pays de destination, ici correspondant à un AS différent. Cette dernière va ensuite livrer le courrier de la manière dont elle le souhaite, qui peut être différente des règles du pays d'expédition (durée et horaire de livraison, taux de perte de la lettre...).

L'analogie atteint sa limite ici. En effet, lors d'une communication inter-AS, il se peut que le paquet traverse plusieurs AS avant d'arriver à destination, multipliant de fait le nombre d'aléas possibles. Ce transfert de données correspond à une liaison de transit. Le peering est une liaison directement point-à-point entre deux AS, ce qui a pour avantage de limiter les aléas et de limiter les coûts. De plus, le tarif d'une liaison de peering est bien inférieur à celui d'une liaison de transit.

1.7 Implications

Du fait de cette hiérarchisation et de ce fonctionnement de proche en proche sans aucun contrôle sur les flux en dehors de son propre réseau, il est impossible pour un opérateur de prédire la qualité de service de bout en bout. Souvent, les opérateurs ont une connectivité redondée à Internet via plusieurs Tier-1 (on parle alors de réseau « multi-homed »). Pour un flux donné, le choix de la sortie du réseau à utiliser, qui implique donc un routage particulier via Internet, est déterminé par les algorithmes de routage. Ceux-ci ne prenant jamais en compte des facteurs de qualité de liens, il est difficile pour un opérateur de choisir la bonne sortie de son réseau. C'est dans ce contexte que se situent les travaux présentés ici.

Nous allons répondre à cette problématique en observant le comportement des flux

pour les principaux AS distants, vis-à-vis de Post Luxembourg, afin d'en définir un modèle. Si le trafic s'éloigne de la prédiction déterminée par ce modèle, synonyme d'évènement réseau, l'équipe en charge du réseau sera avertie et pourra prendre des mesures afin de corriger les éventuelles pertes de qualité pour les clients.

1.8 Conclusion

Les solutions envisagées pour recueillir les données de trafic du réseau de Post Luxembourg ainsi que la solution retenue vont être présentées dans le chapitre 2. Dans le chapitre 3, une étude préliminaire de ces données sera présentée, ainsi qu'une agrégation des données sur l'AS source et destination. Ces résultats seront ensuite utilisés dans le chapitre 4 afin de tester différentes méthodes de modélisation et de sélectionner la plus adaptée à notre situation. Pour finir, le chapitre 5 présentera une application développée sur la base de ces travaux permettant de détecter en temps quasi-réel les anomalies de trafic par le biais d'une modélisation fondée sur une méthode préalablement choisie. Une conclusion de ces travaux et des apports présentés dans le manuscrit seront dressés dans le chapitre 6.

Chapitre 2

Acquisition des données

Sommaire

Les travaux présentés ici ont débuté par la mise en place d'une collecte de données de trafic à grande échelle au niveau du réseau de l'opérateur Post Luxembourg. L'étude du trafic et du comportement des réseaux informatiques est un domaine très actif. En effet, de nombreuses recherches se focalisent sur les différentes méthodes disponibles afin de mener à bien ces mesures [6, 7, 8].

Lorsque l'on évoque la mesure de trafic réseau, il nous vient immédiatement en tête le fait de mesurer un lien ou le trafic d'un routeur. Mais lorsqu'il s'agit d'étudier le trafic internet à l'échelle d'un opérateur, les problématiques en jeu ne sont pas les mêmes. Il convient donc de prendre en compte l'impact que pourrait avoir cette étude sur l'ensemble du réseau, la quantité de données à traiter et à stocker, la répartition et le positionnement des points de mesures... De fait, le choix de ces données et de la méthode de récolte sont des points cruciaux à prendre en compte.

Dans l'objectif de trouver la méthode la plus adaptée, différentes solutions ont été évaluées et vont être présentées dans ce chapitre. Seule l'une d'entre elles a été retenue pour la suite des travaux.

2.1 La métrologie des réseaux

Par définition, la métrologie désigne la science des mesures. Dans le cadre des réseaux, son objectif est de connaître et comprendre le réseau afin de pouvoir non seulement intervenir en cas de problème, mais aussi d'anticiper l'évolution du réseau, planifier la mise à niveau d'équipements, et améliorer les performances pour les utilisateurs.

Il existe de nombreuses solutions pour mesurer le trafic sur un réseau [9, 10] qui sont réparties en deux grandes familles suivant l'impact qu'elles ont sur le réseau : les méthodes de mesures actives et les méthodes de mesures passives.

On se concentrera ici sur les mesures adaptées à l'étude du trafic Internet. En effet, de nombreux outils et protocoles sont définis aux différents niveaux du modèle OSI, qu'ils soient dédiés à l'opérateur ou à l'utilisateur. Ces outils sont regroupés sous le terme OAM "Operations, Administration and Maintenance". Les recommandations conjointes du Metro Ethernet Forum, de l'ITU-T et de l'IEEE [11, 12, 13, 14] couvrent essentiellement les outils OAM Ethernet, et s'adressent aux opérateurs d'infrastructures et aux clients afin de contrôler les « Service Level Agreement » (SLA, contrat entre l'opérateur et le client définissant la qualité de service attendu). D'autres protocoles de plus haut niveau, comme le protocole ICMP [15] de la couche réseau ou des outils pour MPLS [16], rentrent également dans la catégorie des outils OAM. Suivant les besoins et le positionnement de l'observateur, différents outils peuvent être utilisés, comme illustré dans [17]. Dans cette section, une liste non-exhaustive des méthodes adaptées à l'étude du trafic Internet sera présentée.

2.1.1 Méthodes de mesure actives

Le principe des mesures actives est fondé sur l'hypothèse que la qualité offerte de bout en bout ne peut être évaluée que par une application qui emprunte le réseau. Elles visent donc à mesurer directement la qualité de service du réseau telle qu'elle est ressentie par une application quelconque.

Le `Ping` et le `traceroute` sont des méthodes actives. Elles permettent de mesurer le RTT (ou « Round-Trip Time »), c'est-à-dire le temps que met l'information à faire l'aller-retour entre la source et la destination. Cette mesure peut être utilisée pour détecter des congestions [18] entre l'hôte source et des hôtes ou routeurs distants.

Ces campagnes de mesures peuvent être réalisées depuis un serveur local, mais aussi depuis des sondes mises à disposition du public par des entités de recherches. Par exemple, le RIPE NCC (pour « Réseaux IP Européens - Network Coordination Centre »), registre régional d'adresse IP pour l'Europe, une partie de l'Asie et le Moyen-Orient, développe un réseau de sondes réparties sur Internet. Ce sont les utilisateurs de RIPE qui hébergent ces sondes, celles-ci étant disponibles pour la communauté afin de participer à des campagnes de mesures. Fin 2017, le nombre de sondes actives a dépassé la barre des 10 000.

De nombreuses méthodes actives se basant sur la mesure du RTT ont été développées dans le but de mesurer la bande passante, correspondant à vitesse de transfert maximale, disponible entre deux hôtes d'un réseau en se fondant sur le fonctionnement du protocole TCP [19].

Toutes ces méthodes permettent des analyses ciblées mais elles entraînent un biais dans la mesure du trafic réel traversant un lien. En effet, en plus d'évaluer le flux légitime, elles mesurent les paquets qui ont été envoyés afin de calculer ce débit. C'est en cela que se différencient aussi les deux types de méthodes, les solutions dites « passives » ne génèrent pas de trafic supplémentaire sur les lignes observées.

2.1.2 Méthodes de mesure passives

Une autre solution est donc d'employer des méthodes d'observation passives du réseau. Par exemple, il est possible d'observer la totalité du trafic traversant une interface afin d'en faire un rapport détaillé. Dans ce cas, le trafic est souvent dupliqué (appelé en pratique « mirroring ») et envoyé vers un serveur dédié à ces mesures [20, 21]. Une limitation de ce modèle est la quantité de trafic qu'il est possible de traiter et le coût des cartes d'acquisition de trafic. Les interfaces à observer dans cette étude peuvent aller jusqu'à 100 Gbps, ce qui est bien au-dessus des limitations de ce type de carte pour un coût raisonnable.

Il est également possible d'utiliser le protocole SNMP [22] (pour « Simple Network Management Protocol »), protocole pouvant servir à la configuration d'équipements à distance mais aussi à la collecte des informations via le standard RMON (pour « Remote Network Monitoring ») [23]. En effet, chaque interface des équipements réseaux peut être configurée pour garder en mémoire des informations comme le nombre de paquets envoyés et reçus, l'utilisation du lien... Un serveur central peut ensuite demander ces informations aux équipements afin de les traiter ultérieurement. Pour notre étude, le protocole SNMP n'a pas été choisi car il ne présente pas suffisamment d'informations sur le trafic en lui même, il offre plutôt une vue sur l'utilisation d'une interface.

La dernière méthode à être présentée est le protocole Netflow [24]. Elle a d'abord été développée par Cisco avant de devenir un standard et d'être implémentée par la plupart des constructeurs (avec des noms différents : Cflowd chez Nokia et Juniper Networks ou NetStream chez Huawei Technologies). Le principe de fonctionnement est le même, le routeur regarde les en-têtes des paquets entrant et sortant de ses interfaces et les classifie en flux. Un flux est unidirectionnel et est défini par plusieurs champs (voir table 2.1). Une fois qu'un flux est terminé ou expiré (marqueurs FIN ou RST dans une connexion TCP, inactivité, expiration d'un compteur pour un flux de longue durée...), il est envoyé par le routeur vers un serveur servant de collecteur.

Afin de ne pas surcharger le CPU des équipements, il est possible de régler un paramètre définissant la périodicité à laquelle le routeur extrait les informations d'un paquet :

Champ	Numéro	Description
IN_BYTES	1	Compteur d'octets d'entrée du flow
IN_PKTS	2	Compteur de paquets d'entrée du flow
OUT_BYTES	23	Compteur d'octets de sortie du flow
OUT_PKTS	24	Compteur de paquets de sortie du flow
INPUT_SNMP	10	Index de l'interface d'entrée du flow
OUTPUT_SNMP	14	Index de l'interface de sortie du flow
DIRECTION	61	Direction du flow
FLOWS	3	Nombre de flows
PROTOCOL	4	Protocole IP du flow
SRC_AS	16	BGP AS source du flow
DST_AS	17	BGP QS destination du flow

TABLE 2.1 – Exemple de champs d'un rapport Netflow v9

le « sampling » (ou échantillonnage). L'équipement ne regardera donc plus qu'un paquet tous les x paquets. C'est aujourd'hui une des méthodes les plus utilisées afin de mesurer le trafic réseau [25], de par son implication dans la majeure partie des solutions de protection contre les attaques DDoS (pour « Distributed Denial of Service »).

C'est cette dernière solution, Netflow dans sa version 9, qui a été retenue pour la suite de cette étude. Son fonctionnement et son paramétrage vont être détaillés dans la section suivante.

2.2 Netflow

2.2.1 Explication détaillée

Aujourd'hui, Netflow est omniprésent lorsqu'il s'agit de réaliser des statistiques de trafic sur un réseau, qu'il soit d'opérateur, d'université, d'entreprise... De nombreuses études se fondent sur ce protocole. Des indications sur les bonnes pratiques sont disponibles dans [25]. Cette méthode de mesure peut être utilisée à différentes échelles. Une étude comparative entre les protocoles Netflow et SNMP pour connaître l'utilisation d'un lien peut être trouvée dans [26]. Les résultats obtenus sont similaires et l'utilisation de Netflow est validée.

Avec un taux d'échantillonnage des paquets de 1/1 (ce qui consiste en l'utilisation de tous les paquets pour faire les statistiques), il est possible de suivre le trafic très finement, au niveau des sessions de connexion. Avec ce protocole, il est possible de voir :

— quel est le taux de flux TCP par rapport à UDP [11] ;
— le taux de connexion TCP ayant reçu un drapeau "RST" en retour ;
— quels sont les numéros de port les plus utilisés ;
— ...

Augmenter l'échantillonnage (ne relever qu'un paquet sur x avec $x > 1$) a pour conséquence la perte d'une partie des paquets, ce qui entraîne une impossibilité de réaliser des vues globales du trafic par application ou par protocole. Néanmoins, l'utilisation d'un faible taux d'échantillonnage implique une utilisation plus ou moins forte du CPU du routeur sur lequel il est configuré. De plus, la taille des fichiers stockés sur le collecteur est augmentée. En effet, un taux d'échantillonnage de 1/1, pour 15 minutes de statistiques sur deux interfaces, réalisant 100 mbps de trafic, peut représenter jusqu'à 500 Mo de données à stocker.

Il est donc important de bien définir le taux d'échantillonnage des paquets en fonction de l'utilisation finale des mesures récoltées.

2.2.2 Placement des sondes

Un deuxième paramètre important à choisir pour la collecte de données via Netflow est le placement des points de mesure du trafic.

Afin d'avoir une collecte de données cohérente, nous avons décidé de ne garder que les interfaces externes des routeurs de bordure du réseau. En effet, ces routeurs PE font la liaison entre le cœur de réseau et Internet (voir figure 2.1). Nous avons choisi de comptabiliser les clients de Post Luxembourg comme faisant partie du réseau. De part ce fait, nous avons donc une frontière claire où nous pouvons collecter des données : toutes les interfaces de transit et de peering. Nous avons également défini une règle pour ne pas comptabiliser plusieurs fois les mêmes informations en ne gardant que les flux internes-externes et externes-internes.

Nous réduisons donc par la même occasion le nombre de sondes correspondant aux routeurs de bordure (un exemple de configuration de routeurs est donné en Annexe A.1 pour des routeurs de marques Cisco de la gamme IOS-XR et Nokia de la gamme SR-OS (ex Alcatel-Lucent)). Ces six équipements sont situés dans différents IXP européens. Leur position, ainsi qu'une mesure de la quantité de flux qu'ils transportent par tranche de 5 minutes, sont retranscrits dans la table 2.2.

Les informations Netflow sont ensuite envoyées sur un serveur de `nfdump`, servant de collecteur Netflow, située au sein de l'entreprise, au Luxembourg, afin d'y être traitées (une indication de la méthode de collecte utilisée est détaillée Annexe A.2).

Ces données sont stockées par l'outil de récolte et doivent donc être traitées et explorées

11. UDP (User Datagram Protocol) et TCP (Transmission Control Protocol) sont deux protocoles de couche 4, transport, présentée dans la figure 1.2 page 4.

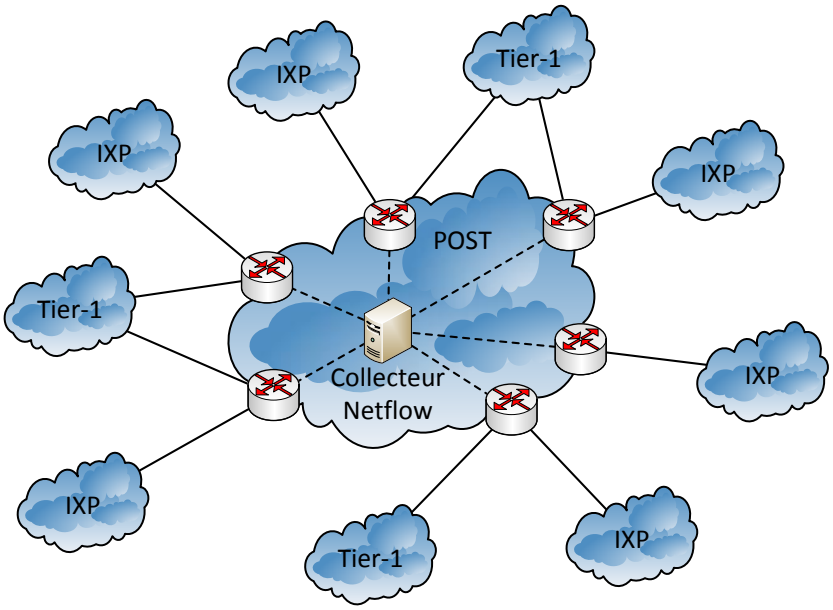


FIGURE 2.1 – Plan du placement des sondes Netflow

Routeur	Lieux	Nombre de flux par 5 min
LU-CIX	Luxembourg, Luxembourg	620 000
AMS-IX	Amsterdam, Pays-Bas	490 000
DE-CIX	Cologne, Allemagne	725 000
France-IX	Paris, France	50 000
LINX	Londres, Royaume-Uni	200 000
BNIX	Bruxelles, Belgique	200 000
	TOTAL	2 285 000

TABLE 2.2 – Emplacement des routeurs et quantité de flux les traversant

afin d'en extraire les informations de trafic qui nous intéressent. Une semaine de récolte de données représente environ 140 Go de fichiers compressés.

La quantité massive de données nous a poussés à réfléchir à une méthode plus adaptée afin de stocker les statistiques. L'agrégation des données nous a donc parue être une solution pertinente. La question subsistante est la suivante : Quel(s) critère(s) devons-nous sélectionner pour définir les conditions d'agrégation de ces données ? Des recherches comme celles décrites dans [27, 26, 28] proposent quelques pistes en démontrant que l'étude de systèmes autonomes (AS) est cohérente pour notre objectif de détection d'anomalies. La section suivante est dédiée à la discussion de ce choix.

2.3 Agrégation des données par AS

De nombreuses solutions sont possibles pour agréger le trafic, certaines d'entre elles vont être détaillées ici.

Une étude par IP est difficilement envisageable pour plusieurs points. Si nous voulons garder la matrice de flux de chaque IP toutes les 5 minutes, cela représente une quantité de données qui n'est pas envisageable. De plus, l'étude d'une IP client n'est pas intéressante pour la détection d'évènements à large échelle car une IP particulière pourrait ne pas être touchée.

Une granularité par interface ne correspond pas non plus à notre attente. En effet, le routage interdomaine (trafic interAS) subit de nombreuses instabilités [29] dues à des pannes, des mauvaises configurations, des optimisations de routages ... De par ce fait, le trafic vers un AS peut subitement changer de point d'entrée et de sortie du réseau, sans aucune modification de la configuration du réseau, seulement par la mise à jour des routes Internet. En considérant ces remarques, il nous paraît incohérent de se fonder sur les interfaces afin d'agréger les données. Il est donc nécessaire de prendre du recul et de faire abstraction du réseau physique.

On peut étendre les remarques faites pour l'étude des interfaces à l'étude d'un IXP. Celui-ci étant souvent connecté via une (ou plusieurs) interface(s), cela apporterait les mêmes désavantages que l'agrégation par interface.

Une autre méthode consisterait à réaliser des statistiques sur un niveau d'AS. Un système autonome étant la somme de différents préfixes IP et donc d'IP client détenues par une même entité. Nous pouvons donc en déduire que les caractéristiques de trafic de ces IP sont, dans l'ensemble, équivalentes au sein d'un même système autonome [30]. Par exemple : la situation géographique, le type de business (réseau absorbant du trafic entrant ou créant du trafic sortant), la même forme de trafic... Nous pouvons espérer retrouver les mêmes caractéristiques de trafic en agrégeant le trafic sur ces AS.

Le but de ces travaux étant de modéliser le trafic, il est intéressant de réduire le nombre d'objets à observer tout en gardant une cohérence et en les consolidant. En effet, certaines IP ne peuvent apparaître que sporadiquement alors qu'il est probable qu'agrégés au sein d'un AS, la somme des IP qui compose cet AS génère un trafic assez important pour être modélisé.

De la même manière, une agrégation par préfixe, ou somme de préfixes, serait aussi envisageable. Cependant, l'ensemble des préfixes d'un même AS est très souvent annoncé et routé de la même façon sur Internet. Il n'y a donc pas d'avantage à l'agrégation par préfixe comparé à celle par AS.

Un point important à ne pas oublier quand il est question de données réelles sur un réseau opérateur est la préservation de la vie privée [8]. Il ne doit pas être possible de retrouver une communication particulière entre deux personnes. Cela est d'autant plus

vrai qu'une réglementation très stricte au niveau européen est mise en place. Or, les statistiques Netflow sont définies par un couple adresse IP source et destination. De par sa nature, l'agrégation des flux garantit une anonymisation des données, de sorte qu'il ne soit plus possible de tracer les connexions d'une personne physique ou morale car les IP source et destination ne sont plus présentes.

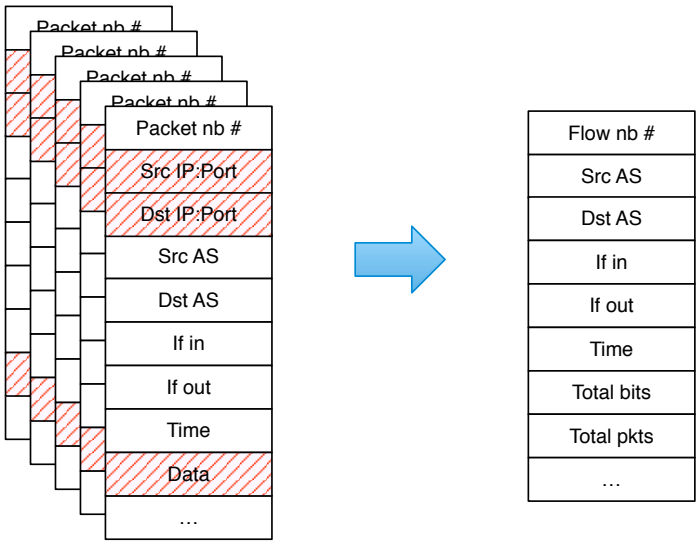


FIGURE 2.2 – Agrégation des données Netflow sur les AS

Parmi tous ces choix possibles, nous avons donc décidé d'agréger les données Netflow sur les champs 16 et 17 qui correspondent à l'AS de source et de destination (voir table 2.1 et figure 2.2). Nous obtenons un tableau à entrées (voir table 2.3) de taille réduite par rapport à une table contenant toutes les informations Netflow qui ne seront pas utiles pour notre étude.

Timestamp	Source AS	Destination AS	Bits	Paquets

TABLE 2.3 – Résultat de l'agrégation sur AS

Les données sont ensuite consolidées sur chaque AS (voir table 2.4) avec un script dont le déroulé est présenté ci-après. La totalité des statistiques pour un intervalle de temps de 5 minutes est parcouru et les données sont consolidées suivant ces règles :

flux de AS_x venant **de** l'*autonomous system* x vers le réseau Post ;
- la somme de tous les flux ayant pour source l'AS_x ;

flux vers AS_x allant **vers** l'*autonomous system* x depuis le réseau Post ;
- la somme de tous les flux ayant pour destination l'AS_x.

De par ces agrégations multiples, il est possible de réduire drastiquement le nombre de statistiques à enregistrer tout en gardant un point de vue global du trafic inter-AS

Timestamp	AS	Bits de	Bits vers	Paquets de	Paquets vers

TABLE 2.4 – Résultat de la consolidation sur chaque AS

qui traverse le réseau. Grâce à cette méthode nous passons de plus de deux millions à quelques milliers de lignes par tranche de 5 minutes.

Dans cette explication, nous avons consciemment allégé le processus réel. En effet, lors de l'étape d'agrégation sur les AS, il est important de garder l'information des routeurs et des interfaces afin de pouvoir filtrer les statistiques que nous voulons conserver. Cela permet de trier les statistiques en amont afin de ne garder que ce qui est utile, par exemple nous avons écarté :

— les interfaces des routeurs vers des clients ;
— des flux venant d'une interface internet et sortant vers une autre interface interne au réseau ;
— des flux venant d'une interface externe et sortant par la même interface.

Cela implique aussi de devoir tenir compte des changements, parfois imprévus, de l'environnement de production (changement d'équipement suite à une panne, nouveau client/interface ...).

Enfin, il est aussi pertinent de s'intéresser à des sous ensembles des clients de Post Luxembourg tout en agrégeant en extérieur au niveau des AS. Il serait possible d'observer non pas un AS, mais un groupe prédéfini par l'utilisateur comme tout le réseau résidentiel ou le datacenter, par exemple. Nous évoquerons ce point plus en détails dans le chapitre 5.

2.4 Conclusion

Nous avons montré dans ce chapitre 2 comment et sous quelle forme nous avons recueilli les données qui serviront dans le reste de cette étude. Les différents choix ont été expliqués et détaillés pour présenter les solutions retenues.

Une proposition de cette thèse est d'avoir développé une nouvelle technique de mesure du trafic Internet fondée sur les AS. Pour cela, une méthode de mesure passive, Netflow, a été choisie afin de récolter les statistiques de trafic. Le paramétrage de ce protocole ainsi que le placement des sondes ont été décrit. Une solution aux problématiques liées à la récolte massive de ces données, une agrégation par AS, est proposée.

Dans le prochain chapitre, nous allons présenter une analyse plus approfondie des statistiques obtenues afin d'en déceler des particularités pouvant nous aider à modéliser le trafic.

Chapitre 3

Analyse des données agrégées à un niveau de systèmes autonomes

Sommaire

L'objectif de ce chapitre est d'acquérir un maximum de connaissances *a priori* pouvant être utilisées par les différentes méthodes d'identification qui seront présentées dans le chapitre 4.

3.1 Points d'étude

Dans ce chapitre, nous allons détailler nos observations sur le trafic interdomaine au niveau de l'agrégation de systèmes autonomes. De nombreuses études ont été réalisées pour déterminer la consommation d'un réseau résidentiel d'opérateur [31, 32, 33] ou d'université [34]. Dans [33], une étude journalière de la consommation de trafic est réalisée pour deux opérateurs européens (Orange et Telefónica) et permet de définir les types d'applications et d'utilisateurs qui consomment le plus de données. Nous allons voir dans la suite que le trafic des AS d'utilisation résidentielle est conforme aux observations (heure de pic

de trafic, sursaut de trafic lors de la pause déjeuner). Nous retrouvons aussi ces analyses de comportement de trafic dans [32]. Dans cette étude, il est question d'examiner le trafic d'une ville en fonction de l'application utilisée (web, transfert de fichier, streaming...) et du type de connexion des clients (1 Mbps symétrique, 10 Mbps symétriques...). Une observation similaire à un niveau protocolaire (TCP/UDP, ports) peut être trouvée dans [34]. L'étude [31] dresse l'état de FAI japonais en fonction des différents types de clients et montre déjà l'impact de l'augmentation des trafics globaux dès 2006.

La plupart des études prennent appui sur les applications utilisées pour en définir les usages. Une des propositions de cette thèse est de se concentrer sur la forme qu'a le trafic en se focalisant sur une granularité posée sur les AS et non pas sur les applications, afin de pouvoir en détecter des anomalies.

L'étude des flux d'un réseau opérateur montre de nombreuses similitudes avec l'étude d'autres types de flux à large échelle. En effet, nous pouvons retrouver les mêmes problématiques de périodicités dans le réseau électrique ou dans le réseau d'acheminement d'eau potable. Dans [35], un lien est réalisé entre les différentes périodicités (heures, semaines, saisons) et la consommation en eau d'une ville afin de prévoir au mieux la demande. Ce chapitre présente des résultats similaires extraits de l'étude du réseau Internet de Post Luxembourg.

Dans un premier temps, il est bon de rappeler que nous ne pouvons observer que ce qu'il nous est possible de voir. C'est-à-dire, sur les 60 000 AS qui peuplent Internet, nous en avons décompté 30 000 de visible, soit la moitié, depuis le réseau de Post Luxembourg. Cela peut s'expliquer par différents points :

— Post Luxembourg n'échange pas avec le monde entier mais uniquement avec une fraction de celui-ci. En effet, les clients du réseau résidentiel ont une consommation d'Internet en grande partie locale (majoritairement luxembourgeois, français, allemand ou belge) ou d'Amérique du Nord. Ainsi, il n'y a pas ou peu d'échanges avec d'autres réseaux très éloignés ;
— on peut aussi imaginer que l'échantillonnage des paquets à 1/1000 introduit un masquage de certaines discussions les plus faibles en terme de paquets par seconde. Des AS à très faible débit peuvent ainsi être dissimulés des AS de poids plus important.

Si l'on regarde maintenant la distribution de trafic entre ces différents AS, nous observons très clairement que le trafic n'est pas réparti équitablement entre les différents réseaux. La figure 3.1 représente le trafic cumulé des AS les plus importants sur 24h. On observe que la majorité du trafic est originaire ou à destination d'un nombre restreint d'AS. On peut également remarquer à la figure 3.1 que 35 % du trafic n'est dû qu'à seulement 10 AS. Ce nombre passe à 60 % lorsqu'on regarde seulement le trafic entrant sur le réseau. Notons que ces résultats ayant été calculés sur une période de trafic de 24h, mais ces observations sont reproductibles sur chaque 24h.

La table 3.1 présente la top 5 des AS générant le plus de trafic dans les deux directions

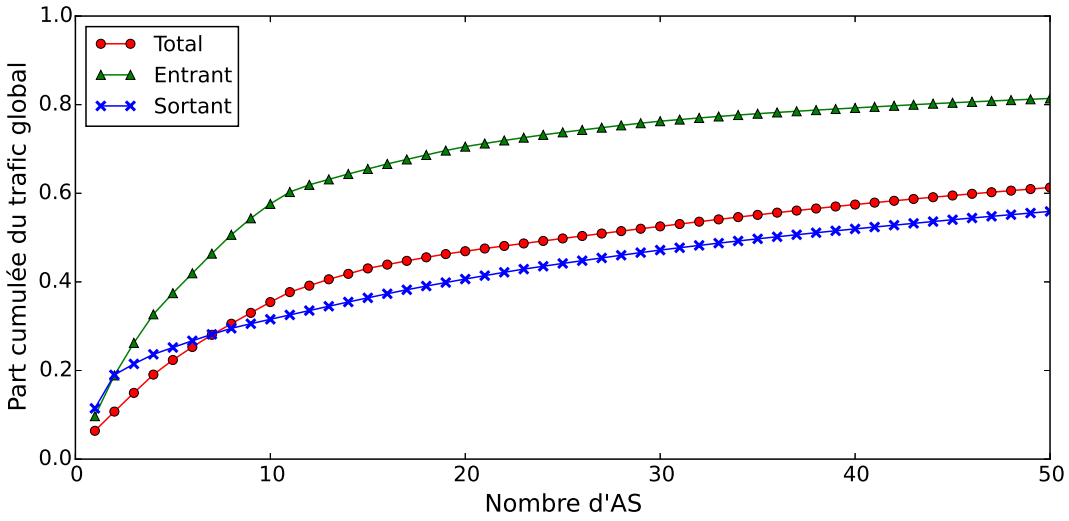


FIGURE 3.1 – Part cumulée du trafic des AS les plus importants

de trafic, puis globalement. Parmi les AS qui génèrent le plus de trafic à destination de Post Luxembourg (voir table 3.1a), nous retrouvons sans grande surprise les plus grands CDN d'Internet, à savoir, Netflix, Apple, Akamai, Facebook. Ici, Netflix est un fournisseur de service de streaming vidéos. Akamai met à disposition ses serveurs pour réaliser du cache pour les entreprises, ce qui explique sa position prédominante dans le trafic. Une découverte dans ce top 5 est de retrouver LU-CIX, IXP commercial luxembourgeois, en première position. Ceci s'explique par le fait qu'il est utilisé comme cache pour plusieurs services (entre autres Google). Pour plus d'informations voir section 3.3.

La table 3.1b présente les AS qui absorbent le plus de trafic sortant du réseau de Post Luxembourg. On y retrouve d'autres FAI mondiaux : Uninet S.A. de C.V (filiale de Teléfonos de México), HINET, Deutsche Telekom AG (DTAG), Telefónica España. Ceci est un point très particulier pour un opérateur d'accès à Internet. En effet, ce trafic sortant est majoritairement dû à l'offre d'hébergement de Post Luxembourg qui génère beaucoup de trafic sortant et qui est très dépendant de ses clients. Il peut donc varier suivant la politique de routage de ces clients.

Une fois les deux directions combinées, nous retrouvons bien les AS les plus importants en terme de trafic bidirectionnel. Le choix de l'étude d'AS s'est porté sur ceux qui génèrent le plus de trafic afin d'en extraire les informations les plus caractéristiques. C'est à ce point qu'est consacrée la prochaine section 3.2.

3.2 Caractéristiques de trafic des systèmes autonomes

Dans cette section vont être détaillées des caractéristiques de trafic qui permettront de classifier les AS dans différents groupes. Ces indicateurs seront utilisés dans le chapitre

Numéro d'AS	Nom	% du trafic global
AS 2906	Netflix	9,7 %
AS 49624	LU-CIX	9,2 %
AS 6185	Apple	7,3 %
AS 20940	Akamai	6,5 %
AS 32934	Facebook	4,8 %

(a) Top 5 du trafic entrant

Numéro d'AS	Nom	% du trafic global
AS 8151	Uninet S.A. de C.V	11,4 %
AS 3462	HINET	7,6 %
AS 3320	DTAG	2,5 %
AS 3352	Telefónica España	2,2 %
AS 14754	Google	1,5 %

(b) Top 5 du trafic sortant

Numéro d'AS	Nom	% du trafic global
AS 8151	Uninet S.A. de C.V	6,4 %
AS 2906	Netflix	4,3 %
AS 3462	HINET	4,2 %
AS 49624	LU-CIX	4,1 %
AS 6184	Apple	3,3 %

(c) Top 5 du trafic global

TABLE 3.1 – Top 5 du trafic observé sur une période de 24h

4 pour guider le choix de modélisation.

3.2.1 Pic de trafic journalier

La figure 3.2 présente deux semaines de trafic pour les trois AS : Netflix, Amazon et Uninet S.A. de C.V. Une caractéristique du trafic facilement visible sur les courbes (voir figure 3.2) est l'heure à laquelle le trafic est le plus élevé. La table 3.2 présente l'heure et la forme des pics de trafic de trois AS. On y peut voir que le moment de la journée auquel ce trafic atteint un maximum journalier varie d'un AS à l'autre. De par ces résultats triviaux, il est possible de déduire le type de business de l'AS. En effet, même sans savoir ce que propose le réseaux de Netflix, nous inférons qu'il s'agit d'un service à destination des particuliers, souvent consulté le soir. On observe dans la figure 3.2a un faible trafic durant

la journée, la courbe du trafic forme un pic entre 12h et 13h, puis augmente fortement dès 18h pour finalement atteindre un pic à 21h30. Le trafic redescend ensuite à un niveau faible durant la nuit.

De la même manière, on déduit de l'AS d'Amazon qu'il s'agit d'un usage professionnel. En effet, le trafic (voir figure 3.2b) est maximum et stable durant toute la plage d'heures ouvrées pour ensuite diminuer dans la soirée. Un comportement similaire est observable mais décalé dans le temps pour l'AS de Uninet S.A. de C.V. (voir figure 3.2c).

Nous retrouvons ici les mêmes résultats que dans l'étude [26] pour les différentes catégories d'AS selon l'heure de la journée où le trafic est le plus élevé. Il est aussi décrit trois catégories : business domestique, client domestique et international.

AS	Heure	Forme du pic
Netflix (AS 2906)	21h30	Étroit autour de 21h30
Amazon (AS 16509)	Entre 8h et 17h	Partagé sur toute la plage
Uninet S.A. de C.V (AS 8151)	Entre 17h et 7h	Partagé sur toute la plage

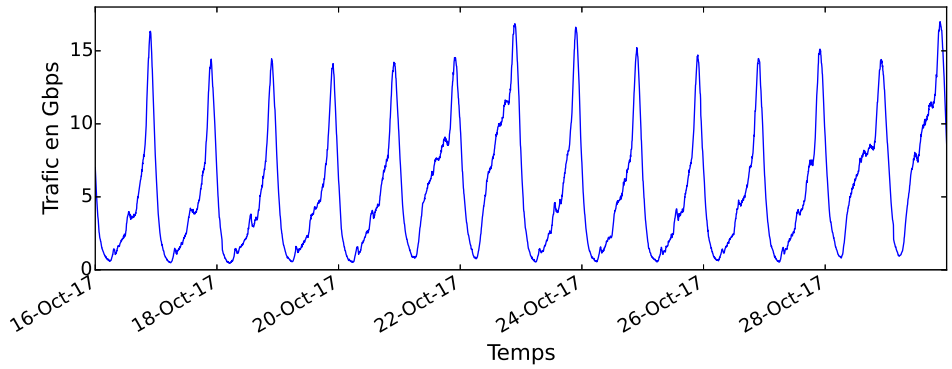
TABLE 3.2 – Quelques exemples de pics de trafic journalier

Pour certains de ces AS, on peut observer, en plus du maximum global, un ou plusieurs maxima locaux. C'est le cas par exemple pour Netflix, où on observe un pic lors de la période 12h-14h. Cela s'explique en connaissant le type de trafic associé à la plate-forme, ou bien le marché sur lequel se situe l'entreprise. Ici, il est facilement imaginable que le pic de trafic vers et depuis Netflix soit dû aux personnes profitant de leur pause déjeuner pour regarder une vidéo sur la plate-forme de vidéos à la demande.

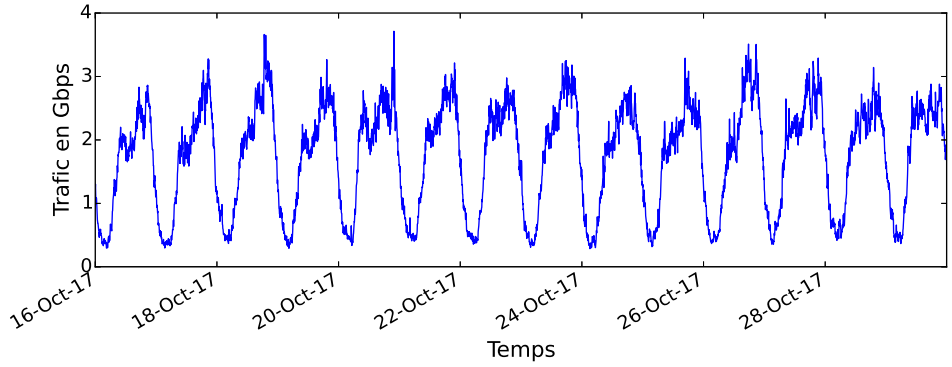
3.2.2 Distribution du trafic dans la journée

Certains AS suivent à la même enveloppe générale de trafic que ceux évoqués dans la section 3.2.1, mais avec des différences notables. En effet, lorsque l'étude porte sur des AS dont les clients sont locaux, on observe des comportements de trafic similaires. Or, le trafic sortant du réseau de Post Luxembourg, originaire des datacenters et à destination du monde, montre quelques différences.

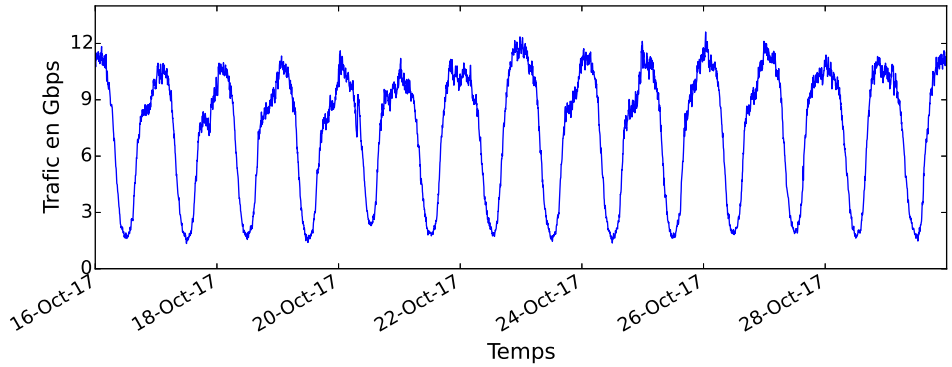
La figure 3.3 présente le trafic de l'AS 14420 sur une semaine. On peut y observer que le trafic est proche de ce que nous avons vu précédemment mais avec un décalage temporel de plusieurs heures. Si on regarde de plus près, on remarque qu'il est de 7h, ce qui correspond à l'écart de fuseau horaire entre le Luxembourg et l'Équateur, pays où est enregistré l'AS 14420.



(a) Trafic venant de l'AS 2906 (Netflix)



(b) Trafic venant de l'AS 16509 (Amazon)



(c) Trafic allant vers l'AS 8151 (Uninet S.A. de C.V.)

FIGURE 3.2 – Deux semaines de trafic de Netflix (AS 2906) (a), Amazon (AS 16509) (b) et Uninet S.A. de C.V. (AS 8151) (c)

3.2.3 Périodicité du trafic

Un autre point important du trafic Internet est sa périodicité (voir figure 3.2a représentant deux semaines de trafic de Netflix). Il est clair qu'il existe une périodicité journalière, car le trafic semble se répéter quasi-identiquement d'un jour à l'autre. Mais lorsque l'on regarde plus en détails le trafic sur une plus longue échelle, on remarque que

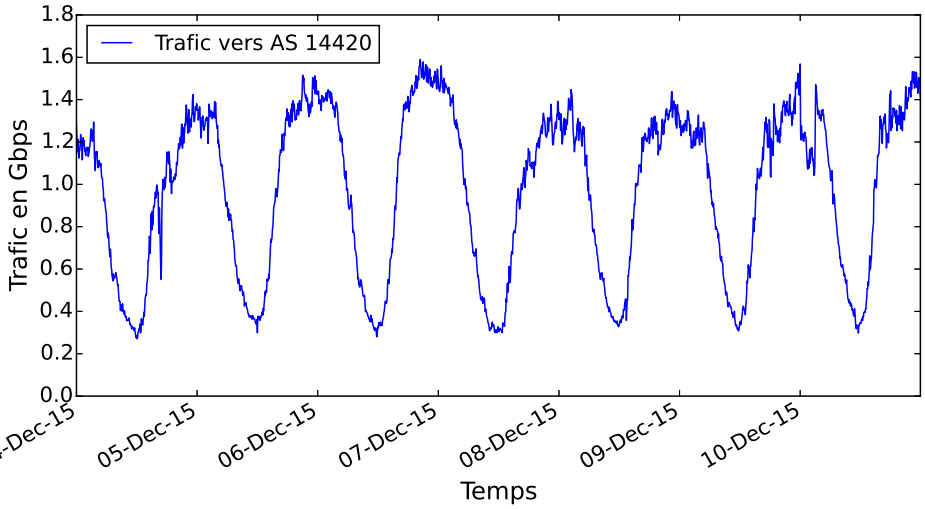


FIGURE 3.3 – Une semaine de trafic vers l'AS 14420

les flux semblent varier entre le week-end et les jours de la semaine. Par exemple, sur la figure 3.2a, on remarque que le trafic diffère légèrement entre les jours de semaines et les jours de week-end (les 21, 22 et 28, 29 octobre 2018). On note des similitudes entre les jours de week-end et les jours fériés. En semaine, le trafic est plus condensé en fin d'après-midi et soirée, mais, lors des jours non travaillés, il est plus étendu et élevé tout au long de la journée. De par ces observations, il est clair que le trafic est soumis à deux périodicités : journalière et hebdomadaire.

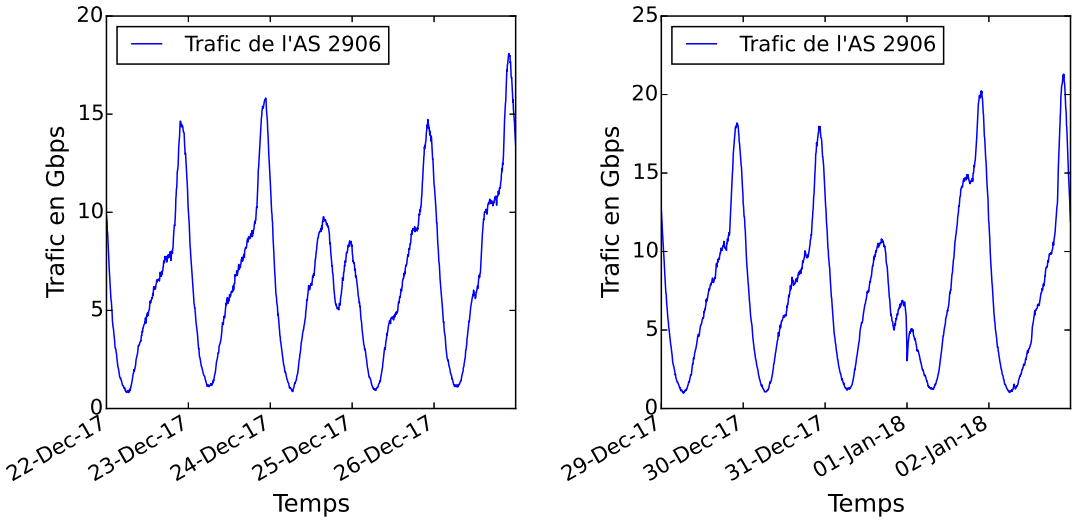
Une différence dans l'enveloppe de trafic peut aussi être observée à d'autres occasions comme par exemple durant les jours fériés. Ceux-ci n'ont pas tous le même effet sur la courbe des flux. En effet, lors des jours fériés de Noël et de Nouvel An (voir figure 3.4), le trafic est bien moindre que lors d'un jour férié ordinaire. On peut corréler ceci, encore une fois, à l'usage qui est proposé par l'AS.

3.2.4 Ratio de trafic

Le ratio, défini par l'équation (3.1), est une caractéristique permettant de classifier les AS de manière rapide. Il est calculé en divisant le trafic venant d'une AS x, $T_{de}(x,t)$, par le trafic allant vers cet AS x, $T_{vers}(x,t)$, le tout au même temps t.

$$R_{ft}(x,t) = \frac{T_{de}(x,t)}{T_{vers}(x,t)} \tag{3.1}$$

La figure 3.5 représente le ratio calculé par (3.1) et le trafic de Facebook de trois jours. Il est alors possible d'observer les variations du ratio au cours du temps. On peut constater une forte variation au cours de la journée. Ces variations s'accentuent encore



(a) Période de Noël (b) Période de la saint Sylvestre

FIGURE 3.4 – Trafic de Netflix (AS 2906) durant les périodes de Noël (a) et de la saint Sylvestre (b)

plus lorsque le trafic est le plus faible, généralement la nuit. Le ratio observé à l'heure du pic de trafic semble quant à lui stable d'une journée à l'autre. Il peut donc servir d'indicateur de classification :

Ratio < 1 : Le trafic venant de cet AS est inférieur au trafic allant vers cet AS. Le réseau distant consomme plus de données qu'il n'en produit, il agit donc comme un FAI.

Ratio > 1 : Le trafic venant de cet AS est supérieur au trafic allant vers cet AS. Le réseau distant produit plus de données qu'il n'en consomme, il agit donc comme un CDN.

Ratio ≈ 1 : Le trafic venant de cet AS est équivalent au trafic allant vers cet AS. Le réseau distant produit autant de données qu'il en consomme. Il agit comme un réseau hybride entre FAI et CDN.

Ces catégories se retrouvent dans la figure 3.6 représentant un histogramme de la distribution des AS en fonction de leur ratio au pic de trafic. Ces résultats sont similaires à ceux décrits dans la littérature [36]. Ceci valide notre étude en trois groupes, avec la majorité des AS appartenant aux deux premiers groupes. D'autres regroupements d'AS sont possibles, en se fondant par exemple sur le business de chaque AS [37]. Cependant cela nécessite une connaissance *a priori* de chaque réseau.

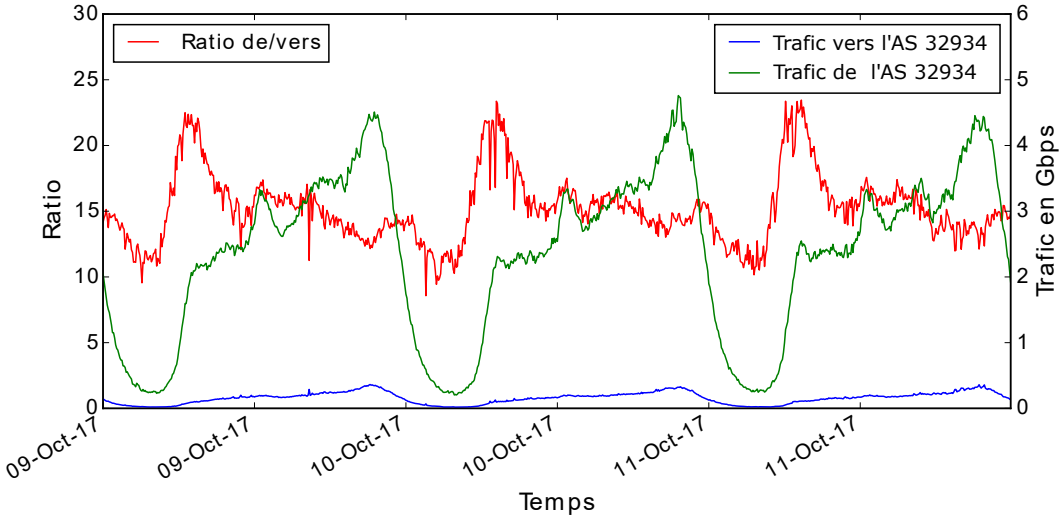


FIGURE 3.5 – Trafic et ratio de Facebook (AS 32934) sur trois jours

3.3 Évolution du comportement des AS

3.3.1 Observations d'évènements particuliers

Hormis les jours fériés, d'autres évènements extérieurs, difficilement prévisibles, peuvent influencer le trafic. Lors de l'été 2016, une étude du trafic Netflix a été réalisée et les données recueillies sont représentées à la figure 3.7. Cette étude, comparée au trafic habituel représenté à la figure 3.2a, montre des particularités lors de certains jours : le trafic de Netflix baisse considérablement les 25, 26, 27, 30 juin et 1, 2, 3, 6, 7 et 10 juillet 206. Une corrélation avec les actualités a permis de comprendre que les jours où le trafic était moindre (presque divisé par deux) correspondaient à des soirs de match du championnat européen de football.

3.3.2 Commentaires sur la caractérisation par AS

La contribution de ce chapitre est fondée sur la caractérisation des AS. Cette granularité apporte des avantages mais aussi plusieurs défauts qui doivent être pris en compte (voir [28]). Plus particulièrement, une analyse sur les AS peut parfois amener à une vue biaisée du trafic.

Pour illustrer ce propos, nous allons évoquer un cas réel survenu pendant cette étude : un cache Google (AS 15169) a été activé au sein de l'IXP luxembourgeois LU-CIX (AS 49624). Ce dernier a comme caractéristique de proposer du cache pour certains services à ses clients, c'est pour cette raison qu'il apparaît comme émettant et consommant du trafic.

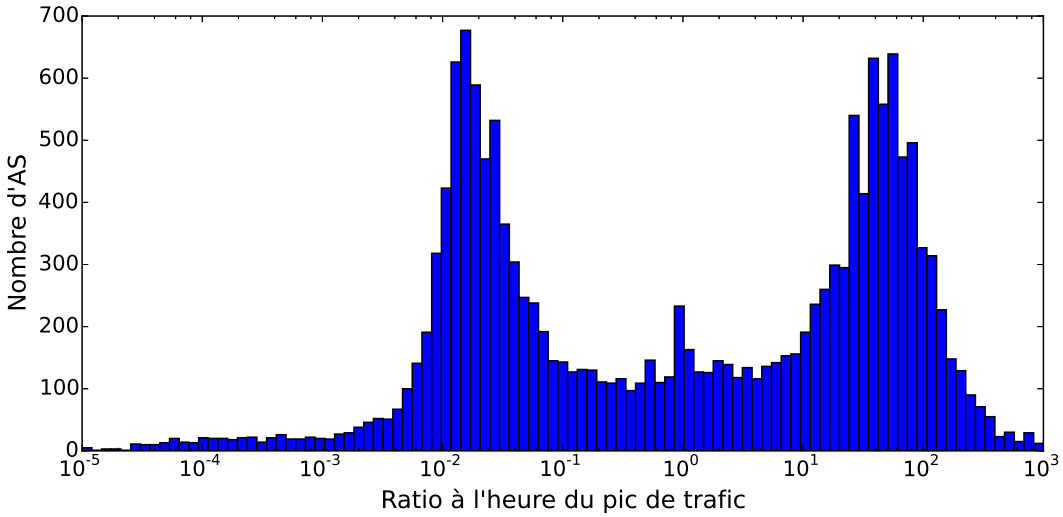


FIGURE 3.6 – Histogramme de ratio de tous les AS visibles à l'heure du pic de trafic

La figure 3.8 représente les trafics des AS Google et LU-CIX avant et après l'activation du cache de Google au sein de LU-CIX. L'activation du cache a eu lieu le 28 septembre 2015 et a amené une chute de trafic venant de Google et une augmentation de trafic venant de LU-CIX. Le trafic de l'AS15169 est passé de 14 Gbps à 6 Gbps en pic. La situation inverse est survenue à l'AS 49624. Si on ajoute le trafic des deux courbes (voir figure 3.8), on peut voir que le trafic dans son ensemble n'a pas été impacté par l'activation du cache. Cependant, une partie du trafic de Google n'est plus comptabilisé pour l'AS 15169 mais pour l'AS 49624 de LU-CIX. Sachant que LU-CIX propose plusieurs caches pour différents services, cela explique sa position dominante dans la table 3.1 peut aussi être expliquée.

Ces différents évènements peuvent amener à une congestion des liens. En effet, le trafic vers LU-CIX a augmenté de 8 Gbps dans l'exemple précédent. Ce cas était prévu et maîtrisé, mais des évènements externes imprévus peuvent avoir un impact non négligeable sur le réseau. C'est dans cette optique de détection d'anomalies que la suite de cette étude est menée.

3.4 Conclusion

Ce chapitre 3 a permis de mettre en évidence les résultats et les observations acquis lors de cette étude. Le trafic des différentes AS montre de grandes similitudes en terme de forme et de périodicité.

Cette double périodicité, quotidienne et hebdomadaire, va être utilisée dans la suite de l'étude. Nous avons aussi mis en évidence que des évènements externes ou internes peuvent influencer le trafic des AS.

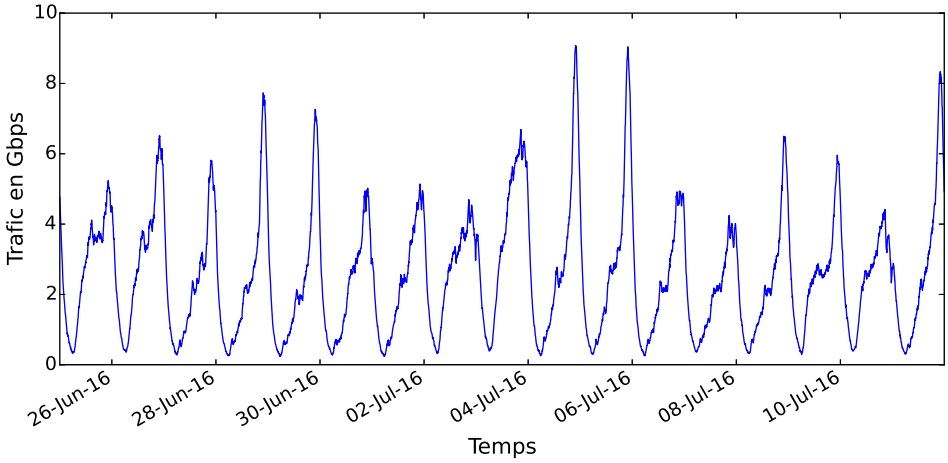


FIGURE 3.7 – Trafic de Netflix durant le championnat européen de football de l'été 2016

Nous allons nous servir de ces informations *a priori* dans le chapitre 4 afin de choisir et de paramétrer plus finement une méthode de modélisation adaptée au système dans le but de détecter ces phénomènes perturbateurs.

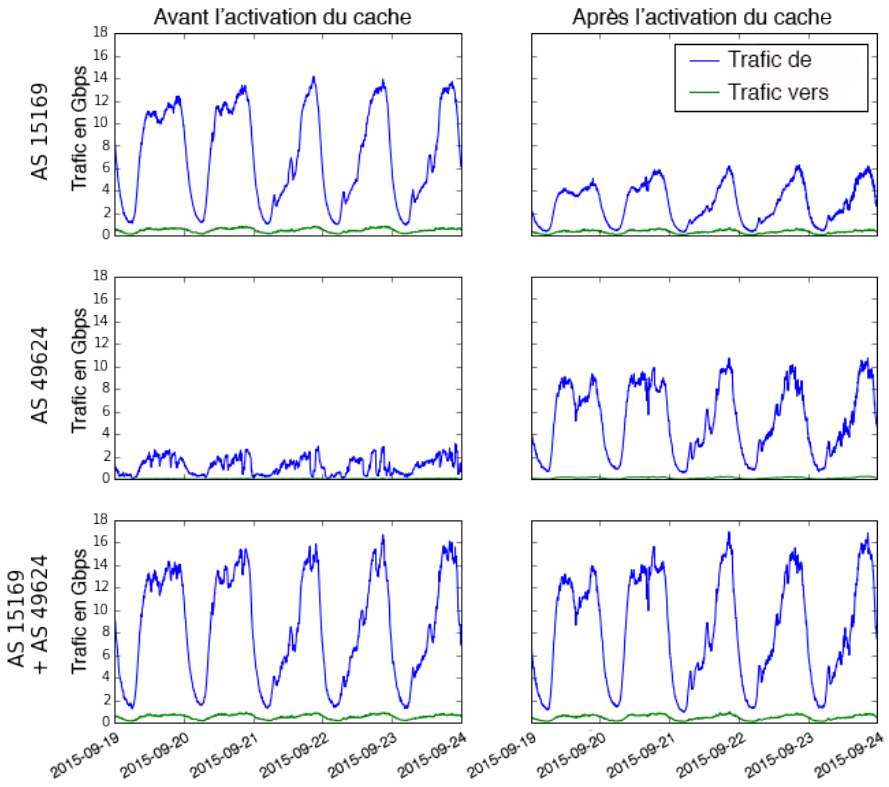


FIGURE 3.8 – Effet de l'activation du cache Google (AS 15169) au sein de LU-CIX (AS 49624) sur le trafic des deux AS

Chapitre 4

Modélisation du trafic

Sommaire

Dans les chapitres précédents nous avons observé le trafic interdomaine à la bordure du réseau de Post Luxembourg et nous en avons extrait des informations pertinentes concernant la forme des habitudes de trafic. De la même manière, nous avons découvert que de nombreux évènements, externes ou internes, impactant le réseau ou non, sont à l'origine de modifications des habitudes de trafic. Dans ce chapitre, le but sera d'obtenir une modélisation du comportement dit « normal » (ou sans anomalie) des AS les plus représentatifs du trafic de Post Luxembourg. Ces modèles serviront de bases, dans le chapitre 5, pour détecter des anomalies de trafic.

4.1 État de l'art

La modélisation de trafic réseau est un champ d'études complexe. En effet, il est très dépendant de l'échelle à laquelle se place l'étude.

Parmi les différents travaux portant sur la modélisation du trafic Internet, on relève l'étude comparative de Cortez *et al.* [38] mettant en avant les types d'approches dédiées à cette problématique : analyse en séries temporelles et réseaux de neurones. Ces méthodes sont appliquées à des données échantillonnées sur 5 minutes, une heure et un jour afin de déterminer l'approche la plus adaptée suivant le besoin. Ils remarquent que les méthodes classiques (modèles ARIMA) de séries temporelles ne sont pas adaptées à la modélisation de trafic réseau, du fait de la charge de calcul très importante de ce type de données. Il est à noter aussi que l'étude préconise l'utilisation des méthodes Holt-Winter ou réseaux de neurones pour des prédictions à moyen ou long terme, lorsque le nombre de données d'apprentissage est conséquent.

L'approche ARIMA est revue et dérivée sous une forme *seasonal ARIMA* par Moussas *et al.* [39] pour s'appliquer sur ces données Internet. Les limitations de la méthode ARIMA classique sont démontrées et l'utilisation d'une méthode SARIMA à simple périodicité est présentée. Elle permet d'obtenir de bons résultats sur des datasets de taille moyenne (semaine) et longue (mois). Une optimisation des paramètres de synthèse doit cependant se faire pour chaque dataset étudié, ce qui rend l'utilisation de cette méthode délicate.

Par ailleurs, les caractéristiques des données Internet reposent essentiellement sur la double périodicité des données, comme expliqué au chapitre 3. Cependant cette caractéristique n'est pas propre au domaine des réseaux informatiques mais se retrouve dans d'autres champs applicatifs, comme par exemple celui de la consommation électrique. Ainsi, Taylor *et al.* dans [40] étudient la modélisation de la consommation électrique afin de prédire l'utilisation future du réseau électrique. À cette fin, les méthodes classiques de type SARIMA sont utilisées mais ont dû être adaptées afin de prendre en compte une seconde périodicité. En effet, la méthode consiste en la multiplication de deux modèles ARIMA saisonniers (SARIMA) avec des périodicités s_1 et s_2 définies comme reflétant les

deux périodicités (ici, journalière et hebdomadaire).

$$\text{ARIMA}(p,d,q) \times (P_1, D_1, Q_1)_{s_1} \times (P_2, D_2, Q_2)_{s_2} \tag{4.1}$$

Il est intéressant de noter que dans ce cas d'étude, un seul dataset est utilisé pour une modélisation. Les paramètres du modèle sont ensuite adaptés au mieux pour ce dataset en particulier. Notre étude nécessitant de trouver un modèle assez générique pour être appliqué à différents datasets montrant des similarités, cette approche n'est pas la plus adaptée à notre problématique.

Papagiannaki *et al.* présente l'étude [41] de la modélisation réseau à une autre échelle temporelle. Les travaux précédemment présentés se basent sur des échantillonnages des données inférieurs à l'heure (entre 5 minutes et 30 minutes). Ici, les auteurs se fondent sur un dataset échantillonné toutes les 12 heures, mais sur une période de plusieurs années. Ainsi, ils utilisent un modèle ARIMA classique afin de prédire la tendance du trafic sur les 6 ou 12 prochains mois. Cette prédiction permet de définir quand et où une amélioration matérielle sera nécessaire. Cette approche, bien que très intéressante, n'est que peu applicable à notre étude puisque notre objectif est de détecter des anomalies réseaux à une granularité de 5 minutes.

L'approche proposée dans cette thèse repose sur l'analyse et les tests des techniques issues de l'automatique et de l'identification des systèmes au cas de la modélisation du trafic Internet.

Deux méthodes sont présentées dans ce chapitre. La première repose, comme celles présentées ci-dessus, sur une technique d'analyse de série temporelle issue du domaine de l'automatique mais revue pour l'application de trafic réseau. La deuxième méthode, issue du domaine de l'IA et du machine learning, présente l'intérêt de pouvoir gérer de grands sets de données. La modélisation par machine learning sera introduite dans la section 4.4.

4.2 Identification des systèmes

4.2.1 Procédure d'identification

L'élaboration de modèles de systèmes dynamiques représente un point clé en automatique et dans les sciences expérimentales. La procédure d'identification, qui suit les étapes détaillées dans la figure 4.1, consiste à rechercher un modèle mathématique adéquat pour un système donné à partir de données expérimentales et de connaissances disponibles *a priori* [42].

Les différentes étapes de la procédure d'identification sont résumées ci-après :

Étape 1 : *Développement de l'expérience et acquisition des données*

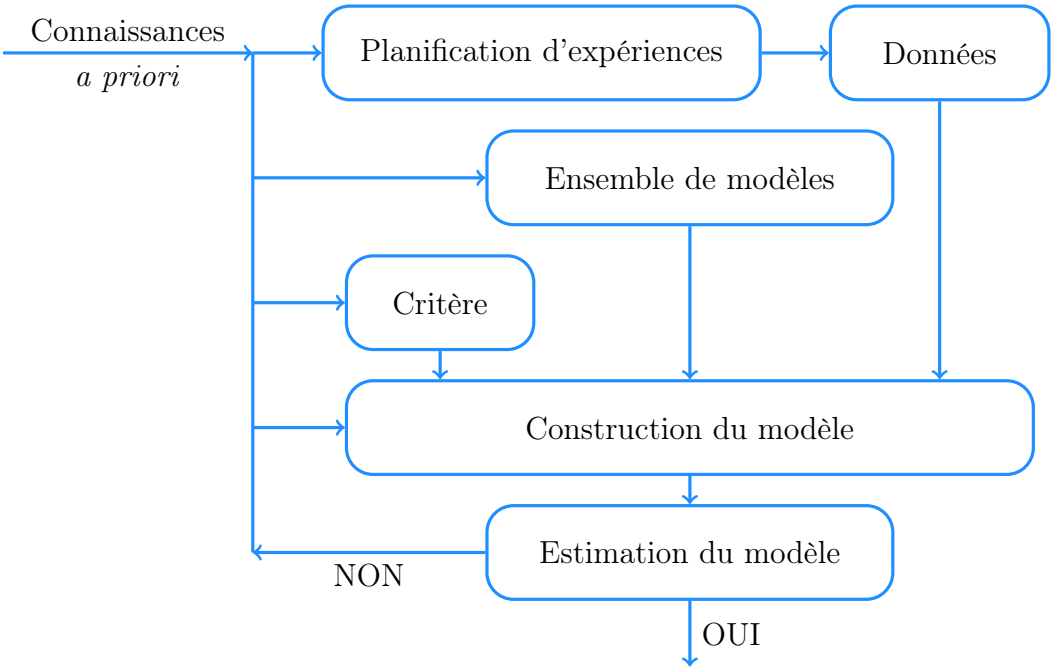


FIGURE 4.1 – Procédure d'identification

La première étape consiste à réaliser des expériences sur le système réel afin d'obtenir des données d'entrées/sorties. L'utilisateur, s'il le peut, choisit les valeurs d'entrées qui excitent au mieux le système afin d'en obtenir le plus d'information possible ;

Étape 2 : *Définition du modèle et de sa structure*
Le choix du modèle et de sa structure est le choix le plus important que fait l'utilisateur lors de cette procédure. En effet, même avec les meilleures données possibles, si le choix du modèle est mauvais, il n'est pas possible d'obtenir de bons résultats [43, 44]. La structure du modèle consiste à définir la relation entre les entrées et les sorties du systèmes ;

Étape 3 : *Estimation du modèle vis à vis du critère d'identification*
Afin d'estimer les paramètres du modèle, il est nécessaire de choisir un critère d'identification. Il sera ensuite possible d'optimiser ces paramètres vis à vis de ce critère afin d'obtenir le résultat optimal au sens du critère ;

Étape 4 : *Validation du modèle*
La validation du modèle revient à reconnaître le modèle comme suffisamment proche du système réel.

4.2.2 Contexte Post Luxembourg

Les méthodes classiques de l'identification des systèmes ne sont pas directement utilisables au contexte d'étude pour plusieurs raisons :

— perturbations non caractérisées ;
— absence de signal d'entrée ;
— données manquantes ;
— échantillonnage à pas variable ;
— impossibilité de définir un plan d'expériences : les données sont recueillies mais il est impossible d'agir ou d'exciter le système afin de pouvoir l'identifier.

Dans ce contexte, on se focalise sur deux approches. La première que nous allons présenter se fonde sur l'étude des séries temporelles. Elle est développée pour modéliser un système composé de tendances périodiques et variant dans le temps [45, 46]. Cette solution est généralement appliquée à des problématiques similaires, mais c'est la première fois qu'elle est utilisée pour étudier le trafic réseau interdomaine.

La seconde est une méthode venant du domaine de l'identification de type boîte noire et du machine learning. Elle a été principalement mise en œuvre dans le contexte de modélisation stochastique de systèmes non linéaires [47]. Même si cette méthode n'est pas directement dévolue à notre problématique, il est possible de la modifier et de l'adapter afin d'obtenir des résultats très satisfaisants.

Avant de détailler ces deux approches, ce chapitre suit la chronologie de la procédure d'identification. De ce fait, la section suivante présente les données utilisées pour la modélisation.

4.2.3 Données

Afin de comparer ces deux méthodes, il est nécessaire de poser un cadre précis aux données qui serviront dans ce chapitre. Le chapitre 3 a montré que la majorité du trafic est dû à une minorité d'AS et que de fait les évènements impactant ces AS ont des répercussions plus importantes sur le réseau et les utilisateurs finaux. Nous allons donc choisir trois des plus importants AS et utiliser la direction de trafic la plus élevée afin de tester nos modèles : Netflix, HINET et LU-CIX. Pour pouvoir tester nos modèles, ces jeux de données sont divisés en deux parties : les deux premiers mois servent à estimer le modèle et le dernier mois est dédié à la validation croisée du modèle.

Les jeux de données sont représentés sur la figure 4.2. Ils sont choisis afin de ne pas être parfaits, mais représentatifs des problèmes qui peuvent apparaître lors de l'utilisation en environnement de production. En effet, dans la partie des données qui est utilisée pour estimer le modèle, plusieurs phénomènes sont visibles :
— une interruption des mesures est visible le 17 décembre 2015 (celle-ci étant due à une maintenance sur le réseau de production) ;
— le jour de Noël et la Saint-Sylvestre causent des réductions importantes du trafic. Ce changement est particulièrement visible sur les données de l'AS Netflix ;
— plusieurs données aberrantes sont présentes dans le dataset.
Il est à noter que lors du dernier mois de cette étude, qui sera utilisé pour comparer

les prédictions du modèle, une montée de trafic est visible sur le dataset de Netflix (voir figure 4.2a). Il sera intéressant de vérifier, dans le cadre d'une validation croisée, si les méthodes de modélisation seront en mesure de prédire cette particularité ou non.

Chaque AS est considéré comme un système inconnu $\mathcal{S}_{\text{o}}$, les données obtenues peuvent être considérées comme N observations de chaque système à différents moments :

$$\mathcal{D}_{\text{N}} = \{(t_1, y_1), ..., (t_k, y_k), ..., (t_{\text{N}}, y_{\text{N}})\}, \tag{4.2}$$

Avec $t_k \in \mathbb{R}_{\geq 0}$ le temps de chaque mesure de trafic $y_k \in \mathbb{R} \, \forall \, k = 1, ..., \text{N}$.

Aucune entrée aux systèmes n'est définie ni mesurée, cependant, grâce aux informations détaillées dans le chapitre 3, nous pouvons remarquer plusieurs points d'intérêts :

— il existe une claire périodicité avec un motif se répétant à fréquence journalière et hebdomadaire (voir section 3.2.3) ;
— une tendance d'augmentation de trafic globale à faible pente.

Ces informations connues *a priori* vont être utilisées pour modéliser le trafic de chaque AS retenu.

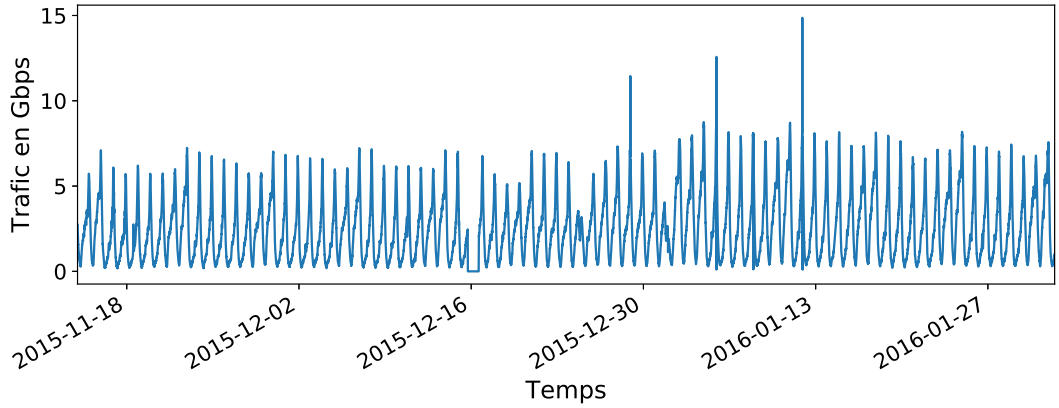
4.3 Méthode de séries temporelles

Lors d'essais préliminaires de modélisation des données, plusieurs méthodes spécialement dédiées aux séries temporelles ont été testées. Une modélisation par processus autoregressif (AR) n'a pas montré de résultats encourageants sur le moyen et long terme, et cela même en utilisant un ordre élevé (> 100). La prédiction du modèle s'effondre après quelques pas sur les données de validation. La principale raison est due à la non prise en compte des périodicités. De ce fait, il nous a paru pertinent de nous tourner vers des approches en séries temporelles permettant de prendre clairement en compte ces particularités, comme présentées dans la section suivante.

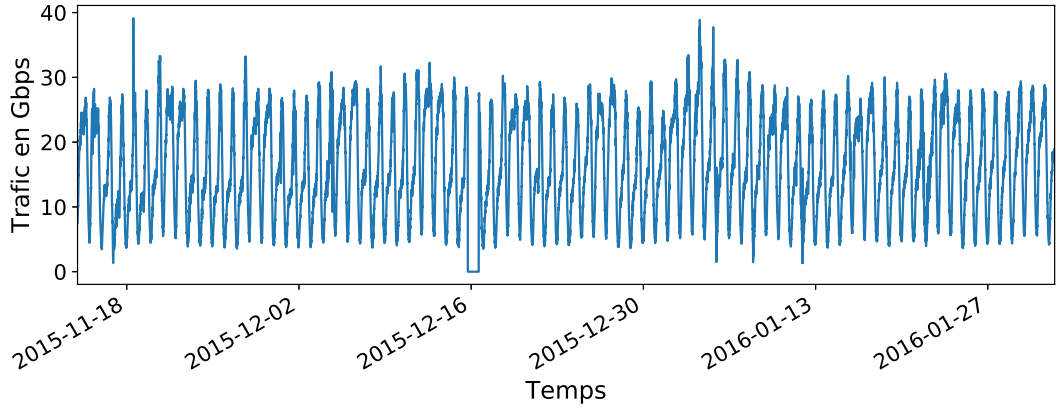
4.3.1 Présentation : Dynamic Harmonic Regression

Dans cette section, nous allons aborder la modélisation par une approche de séries temporelles. Ce champ d'études est dédié à l'estimation de modèles numériques de systèmes sans entrée et variants ou non [45]. De part ses caractéristiques, cette famille de méthodes est tout à fait adaptée à notre problématique de modélisation de trafic Internet.

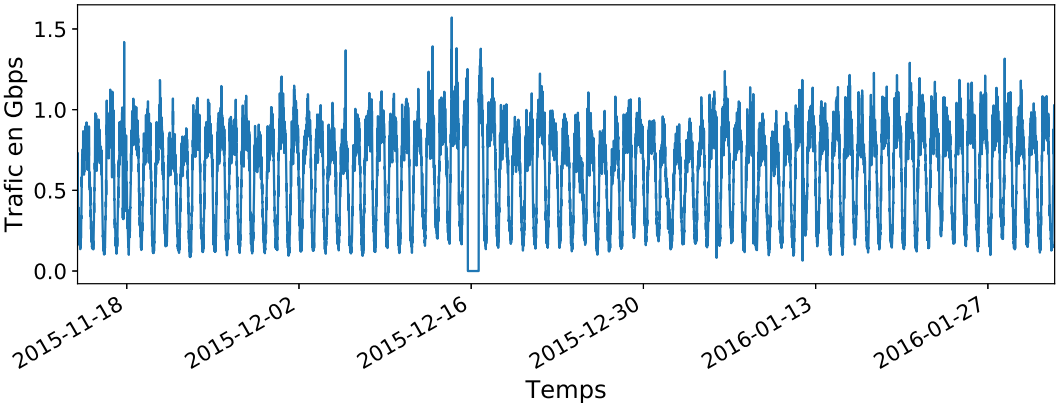
Au sein de cette famille, nous avons sélectionné une démarche particulière nommée *Dynamic Harmonic Regression* (DHR). DHR est une méthode spécifiquement développée pour modéliser des données périodiques et variables dans le temps [46] et est de ce fait une approche parfaitement appropriée pour modéliser le trafic inter-AS.



(a) Trafic venant de l'AS 2906 (Netflix)



(b) Trafic allant vers l'AS 3462 (HINET)



(c) Trafic venant de l'AS 49624 (LU-CIX)

FIGURE 4.2 – Dataset d'étude de Netflix (AS 2906) (a), LU-CIX (AS 49624) (c) et Hinet (AS 3462) (b)

Le modèle DHR se formule comme une combinaison de différents composants pério-

diques :

$$\mathcal{M}_{\text{DHR}} : \quad y_k = T_k + C_k + S_k + e_k, \quad e_k \sim \mathcal{N}(0, \sigma^2) \tag{4.3}$$

T_k **:** tendance basse fréquence

C_k **:** série de Fourier modélisant des variations *cycliques*

S_k **:** série de Fourier modélisant des variations *saisonnières*

e_k **:** bruit (supposé gaussien)

Les deux termes C_k et S_k sont définis par :

$$\begin{aligned} C_k &= \sum_{i=1}^{R_c} \{a_{i,k} cos(\omega_i k) + b_{i,k} sin(\omega_i k)\} \\ S_k &= \sum_{i=1}^{R_s} \{\alpha_{i,k} cos(f_i k) + \beta_{i,k} sin(f_i k)\} \end{aligned} \tag{4.4}$$

Cette approche peut être résumée en une décomposition spectrale en séries de Fourier des données d'entraînement, où les coefficients (représentant les paramètres du modèles) sont variables dans le temps, conformément au système. Même si l'utilisation de cette méthode est nouvelle pour une application sur des données de trafic réseau, elle a déjà été utilisée pour modéliser un grand nombre de systèmes [48, 49].

4.3.2 Implémentation

L'estimation du modèle ($\mathcal{M}_{\text{DHR}}$) requiert plusieurs pré-traitements avant de pouvoir utiliser la méthode d'estimation DHR. Ceux-ci sont détaillés dans le paragraphe ci-après. Par ailleurs, le modèle a été obtenu à l'aide de la **toolbox CAPTAIN** pour Matlab [50].

— **Prétraitement :** Afin d'améliorer la qualité du modèle, il est préférable de standardiser les observations avant l'entraînement. Cette étape est réalisée en enlevant la moyenne et en normalisant la variance des données. De plus, un sous-échantillonnage à 30 minutes peut aider à obtenir de meilleures prédictions sur le long terme en réduisant l'impact des termes correspondant aux hautes fréquences au sein du modèle. Après plusieurs tests, il a été choisi de combler les absences de données par des données équivalentes.

— **Étude Spectrale :** l'estimation de la base fréquentielle du système est réalisée par la fonction `arspec` de la toolbox. Cette fonction a pour but de trouver le spectre fréquentiel de modèle autorégressif (AR). Dans la pratique, il est plus pertinent d'estimer cette base par rapport à un modèle AR à ordre élevé, appliqué aux données. Ici, la modélisation AR du système est uniquement utilisée afin de définir le spectre fréquentiel.

— **Hyperparamètres :** Le modèle dépend de la configuration de certains paramètres de synthèse à déterminer par l'utilisateur :

— **Paramètres temporels (TVP) :** ces paramètres déterminent l'évolution dans le temps des paramètres du modèle. Ils peuvent être choisis grâce à des connaissances *a priori* du système ou, comme ici, en utilisant une méthode de validation croisée afin d'en choisir la configuration optimale [51].
— **Hyperparamètres du bruit :** les paramètres du bruit peuvent être déterminés en utilisant la fonction `dhropt` de la toolbox. Cette fonction permet l'estimation des hyperparamètres.
— **Estimation :** Une fois tous ces choix réalisés, nous obtenons une configuration optimale pour estimer le modèle $\mathcal{M}_{\text{DHR}}$ sur les données avec la fonction `dhr` de la toolbox.

4.3.3 Résultats

Le trafic des trois AS a été modélisé à l'aide de cette feuille de route. Les deux premiers mois du dataset ont été utilisés pour l'estimation du modèle par la méthode DHR et le troisième mois sert à la validation du caractère prédictif du modèle obtenu. Les figures 4.3, 4.4 et 4.5 représentent les résultats obtenus pour les trois AS Netflix, HINET et LU-CIX.

L'erreur relative est utilisée pour visualiser l'erreur du modèle. Elle est définie par la valeur absolue de la différence entre la valeur réelle et la prédiction à un moment précis, normalisée par le maximum du trafic sur le dataset entier :

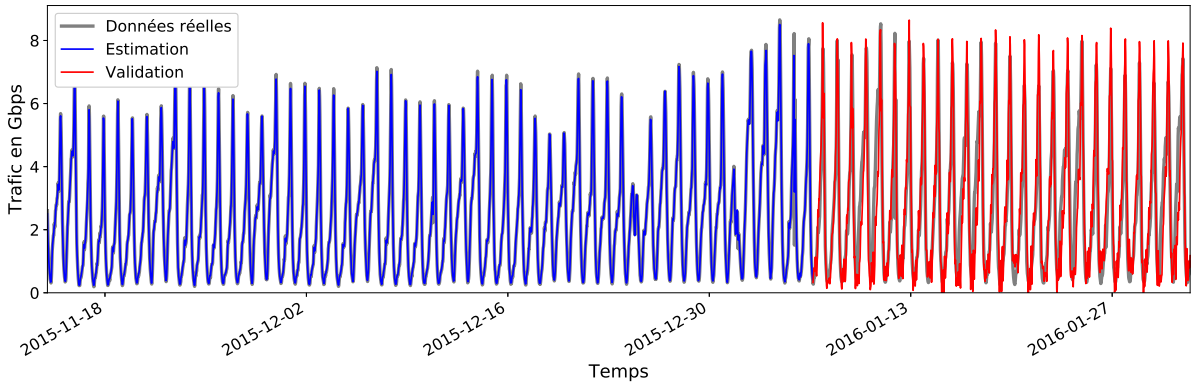
$$\text{Erreur}(t) = \frac{|y(t) - \hat{y}(t)|}{max(y)} \tag{4.5}$$

AS	Netflix (AS 2906)	Hinet (AS 3462)	LU-CIX (AS 16509)
Ordre AR	97	69	51
TVP	Trig.	Trig.	Trig.
Précision	80,8 %	74,9 %	83,9 %

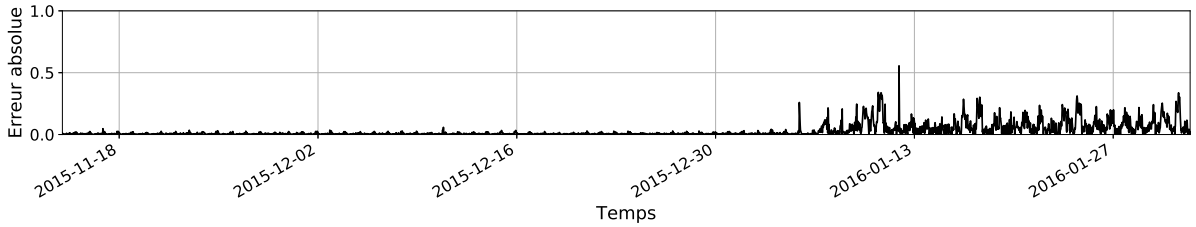
Table 4.1 – Précision des prédictions du modèle $\mathcal{M}_{\text{DHR}}$ pour différents AS à une période d'échantillonnage de 30 min

Les résultats obtenus avec le modèle DHR, pour les trois AS sélectionnés, sont détaillés dans le tableau 4.1 où la précision du modèle, définie par (4.6), est calculée en utilisant la partie des données n'ayant pas été utilisée pour l'estimation.

$$\text{Précision} = 100\left(1 - \frac{||\hat{y} - y||^2}{||y - \bar{y}||^2}\right)\% \tag{4.6}$$



(a) Prédiction du modèle DHR sur les données de Netflix



(b) Erreur absolue du modèle DHR sur les données de Netflix

FIGURE 4.3 – Résultat de modélisation DHR sur les données de Netflix échantillonnées à 30 min

Nous pouvons voir, dans la table 4.1, que la méthode DHR fournit des résultats acceptables avec une précision comprise entre 75 % et 84 % suivant les AS. Dans chacun des cas, pour un échantillonnage des données à 30 minutes, le temps de calcul total, pour l'estimation et la prédiction, est d'environ une minute.

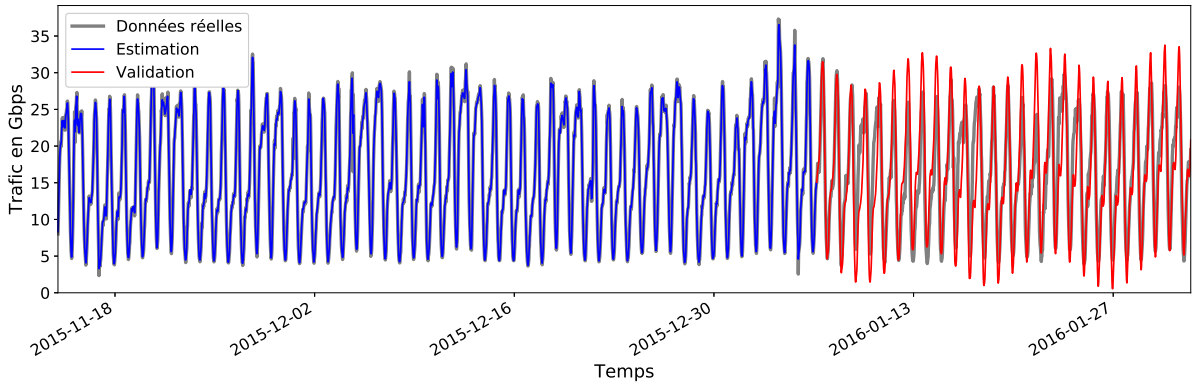
4.4 Méthode de machine learning

Ces méthodes se différencient des méthodes classiques d'identification des systèmes en particulier dans les modélisations dites *boite noire* des systèmes. Dans le domaine du machine learning, il existe deux types d'apprentissage : supervisé et non supervisé. Le machine learning supervisé correspond à l'apprentissage d'un dataset contenant des entrées et des sorties du système. Cet apprentissage peut être de différentes catégories qui dépendent de l'objectif final du modèle : régression ou classification [47].

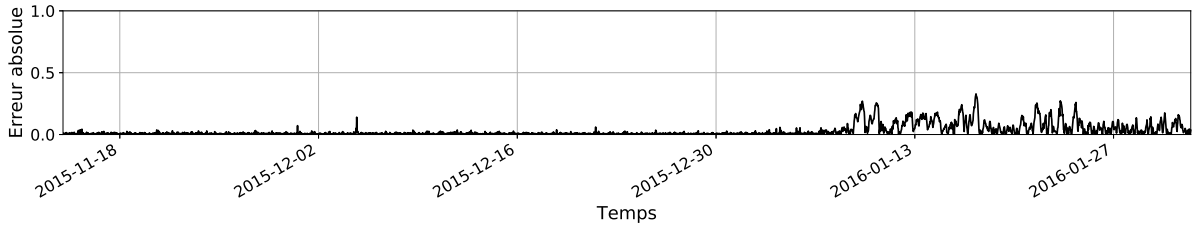
Régression : Prédire les sorties continues du système. Exemple d'application : analyse de séries temporelles ;

Classification : Prédire les sorties discrètes du système. Exemple d'application : reconnaissance de caractères ou de formes.

Différentes méthodes de machine learning sont étudiées dans [52], dans le cadre de prédiction de séries temporelles. La conclusion de cette étude montre que deux méthodes



(a) Prédiction du modèle DHR sur les données de Hinet



(b) Erreur absolue du modèle DHR sur les données de Hinet

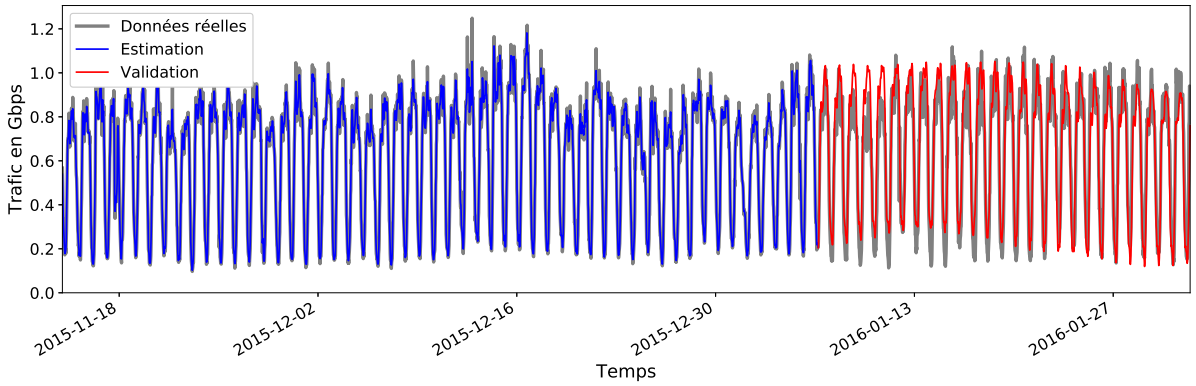
FIGURE 4.4 – Résultat de modélisation DHR sur les données de Hinet échantillonnées à 30 min

sortent du lot dans ce cas d'usage : les réseaux de neurones et les processus gaussiens. Ces deux approches fournissent des résultats encourageants. Celle fondée sur les processus gaussiens n'a pas encore reçu une attention approfondie par les équipes de recherche du domaine et de ce fait mérite d'être regardée et étudiée plus précisément. Il reste en effet encore beaucoup de pistes d'améliorations possibles sur cette méthode. Il nous a donc paru intéressant de proposer une approche fondée sur ces processus gaussiens (ou « Gaussian Processes [12] ») pour la modélisation du trafic Internet. Un des intérêts de cette méthode est de permettre la prise en compte de connaissances *a priori* sur les AS précédemment étudiés (voir chapitre 3).

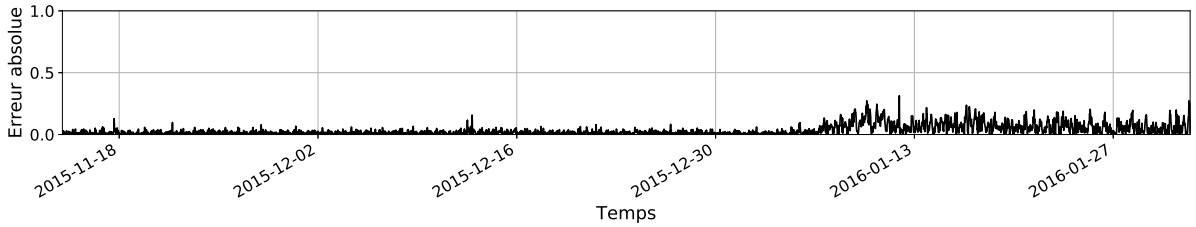
4.4.1 Présentation : Gaussian Process Regression

Le machine learning est une discipline de recherche bien établie et représente aujourd'hui une grande activité et attractivité scientifique que ce soit dans l'industrie ou le milieu académique. Bien que traditionnellement orienté sur l'analyse de données informatiques [53], son rapide développement conduit à son utilisation dans d'autres domaines.

12. Nous utiliserons la notation « GP » dans la suite du document



(a) Prédiction du modèle DHR sur les données de LU-CIX



(b) Erreur absolue du modèle DHR sur les données de LU-CIX

FIGURE 4.5 – Résultat de modélisation DHR sur les données de LU-CIX échantillonnées à 30 min

Dans le cadre d'une régression linéaire définie par :

$$\mathcal{M}_{\text{LIN}} : \quad y = \theta_0 + \theta_1 x + e \tag{4.7}$$

l'objectif de l'identification est d'estimer les paramètres θ_0 et θ_1, afin de représenter au mieux le jeu de données. Une approche bayésienne consisterait à supposer que les paramètres θ_n ne sont plus définis en tant que variable fixe, mais en tant que distribution de probabilité mise à jour avec chaque nouveau point du dataset. L'approche fondée sur les processus gaussiens quant à elle est une méthode non paramétrique. L'objectif n'est plus l'estimation des valeurs ou de la distribution des paramètres θ_n mais de la distribution de chaque fonction $\mathcal{F}(x)$ possible pour expliquer les valeurs observées.

Dans ce domaine, la méthode fondée sur le *Gaussian Process Regression* (ou GPR) est un paradigme bien connu en machine learning [54]. Elle peut être considérée comme une branche au sein des méthodes à noyaux non paramétriques et peut modéliser de nombreuses catégories de systèmes linéaires ou non linéaires.

Ainsi, pour une régression fondée sur des noyaux déterministes, une fonction de noyau est utilisée afin de définir une relation entre l'entrée et un espace de redescription (« feature space »). Dans le cas présent, GPR est une méthode probabiliste dans laquelle un noyau correspond à une fonction de covariance d'un processus gaussien stochastique :

$$\mathcal{F}(.) \sim \mathcal{GP}(0, \mathcal{K}(.,.)) \tag{4.8}$$

avec $\mathcal{F}(.)$ définissant un modèle tel que :

$$\mathcal{M}_{\text{GPR}} : \quad y_k = \mathcal{F}(x_k) + e_k, \quad e_k \sim \mathcal{N}(0, \sigma^2) \tag{4.9}$$

Dans cette expression, x_k est l'entrée du système à un temps k, y_k est la sortie et e_k est un bruit caractérisé par un bruit gaussien centré sur 0.

La modélisation de trafic Internet est un domaine différent de ce qui est réalisé habituellement dans la littérature, notamment parce qu'aucune entrée x_k du système n'est disponible. Cependant, grâce aux particularités relevées précédemment (périodicité et le caractère lisse du trafic) et à une comparaison des exemples similaires dans la littérature (voir section 5.3.2 de [47]), une solution est proposée dans la suite de ce chapitre.

4.4.2 Fonction de covariance et choix des hyperparamètres

Les deux étapes cruciales à la formulation d'un modèle GPR sont :
— la définition de la fonction de noyau (correspondant à la covariance du processus) ;
— la sélection des hyperparamètres s'y référant (la configuration du noyau).
Ces deux étapes sont présentées dans la littérature, par exemple dans [55].

De part nos connaissances sur le trafic interdomaine obtenues dans le chapitre 3, nous savons que le trafic est périodique suivant deux fréquences : journalière et hebdomadaire. Mais aussi que le trafic augmente progressivement. Afin de prendre en compte ces différentes connaissances *a priori*, nous proposons la formulation suivante pour la fonction de noyau :

$$\mathcal{K} = k_{\text{tend}} + (k_{\text{jour}} * k_{\text{sem}}) + k_{\text{bruit}} \tag{4.10}$$

où chaque partie du noyau reflète une particularité du système :

k_{tend} **:** tendance à long terme

k_{jour} **:** composant de périodicité journalière

k_{sem} **:** composant de périodicité hebdomadaire

k_{bruit} **:** bruit blanc

Ces composants peuvent être définis grâce à un choix particulier de fonction de noyau et de leurs hyperparamètres. Pour simplifier le problème et ainsi faciliter la reproduction postérieure à d'autres AS, les hyperparamètres sont réglés manuellement grâce à la connaissance du système. Des améliorations pourraient être faites en optimisant ces paramètres (par exemple en utilisant une validation croisée, comme pour la méthode DHR, ou maximum de vraisemblance). Cependant, en considérant les résultats, il peut être suffisant de garder ces paramètres manuellement définis dans un premier temps.

La tendance à long terme du système peut être modélisée en utilisant un noyau appelé *square exponential kernel*. Ce noyau prend comme paramètre une largeur l que nous

définissons comme étant la taille totale du dataset d'entraînement :

$$k_{\text{tend}} : \; k_{\text{se}}(t,t') = \exp\left(-\frac{(t-t')^2}{2l^2}\right), \; 2l^2 \approx 8\,\text{semaines} \tag{4.11}$$

Les composants périodiques du systèmes peuvent, quand à eux, être modélisés en utilisant un noyau de type *periodic kernel*, avec comme hyperparamètres p, définissant la période du noyau :

$$\begin{aligned} k_{\text{jour}} : \quad k_{\text{per}}(t,t') &= \exp\left(-\frac{2\sin^2\left(\frac{\pi|t-t'|}{p}\right)}{l^2}\right) \\ p &= 1\,\text{jour} \\ l^2 &= 0.05 \\ k_{\text{sem}} : \quad k_{\text{per}}(t,t') &= \exp\left(-\frac{2\sin^2\left(\frac{\pi|t-t'|}{p}\right)}{l^2}\right) \\ p &= 1\,\text{semaine} \\ l^2 &= 1. \end{aligned} \tag{4.12}$$

Le paramètre l est défini plus court pour le composant journalier afin de pouvoir modéliser les différents types de comportement durant la journée (exemples : heures du déjeuner, heures travaillées et non travaillées). Il est au contraire défini plus large pour le composant hebdomadaire afin que chaque jour puisse influencer les jours adjacents.

Pour finir, le composant du bruit est défini comme un noyau de type *white kernel*. Le rôle de ce noyau est similaire à celui du terme de régularisation dans d'autres méthodes de modélisation :

$$k_{\text{bruit}} : \quad k_{\text{wn}}(t,t') = \sigma_{\text{N}}^2\,\delta_{t,t'} \tag{4.13}$$

Où $\delta_{t,t'}$ est le symbole de Kronecker, σ_{N}^2 est un hyperparamètre devant être configuré pour refléter le niveau du bruit du système. Ce dernier paramètre est le moins trivial à définir manuellement. En utilisant une méthode d'essai-erreur nous obtenons des résultats satisfaisants avec $\sigma_{\text{N}}^2 = 0.05$ (voir plus loin, dans la section 4.6)

4.4.3 Implémentation

Comme pour la méthode DHR, un bref descriptif de la mise en œuvre de la modélisation GPR est développée dans cette section. Nous n'allons pas utiliser Matlab ici, bien que

la méthode soit disponible dans la toolbox GMPL [47], mais le langage de programmation Python et la librairie Scikit-learn [56] puisque l'objectif final est l'utilisation du modèle en production chez Post.

— **Prétraitement :** contrairement à la méthodologie précédente, les données ne sont pas traitées en amont de la modélisation. La fonction `learn` se chargera de standardiser les données automatiquement par la suite.
— **Choix des fonctions de covariance :** la sélection des fonctions de covariance est expliquée dans la section 4.4.2. La configuration est réalisée à l'aide de la fonction `GaussianProcessRegressor` de la librairie Scikit-learn. Cette fonction retourne un objet de type *GaussianProcess* qui sera utilisé par la suite.
— **Estimation :** la fonction `learn` de l'objet *GaussianProcess* est utilisée sur les données d'apprentissage afin d'estimer le modèle.
— **Prédiction :** la fonction `predict` est appelée avec un vecteur de temps t_{pred} afin d'en prédire les valeurs de sortie. Celle-ci peut aussi retourner l'écart-type de la prédiction si on lui précise (avec la valeur `return_std` définie à vrai).

L'implémentation en Python est détaillée dans la section 4.7.

4.4.4 Résultats

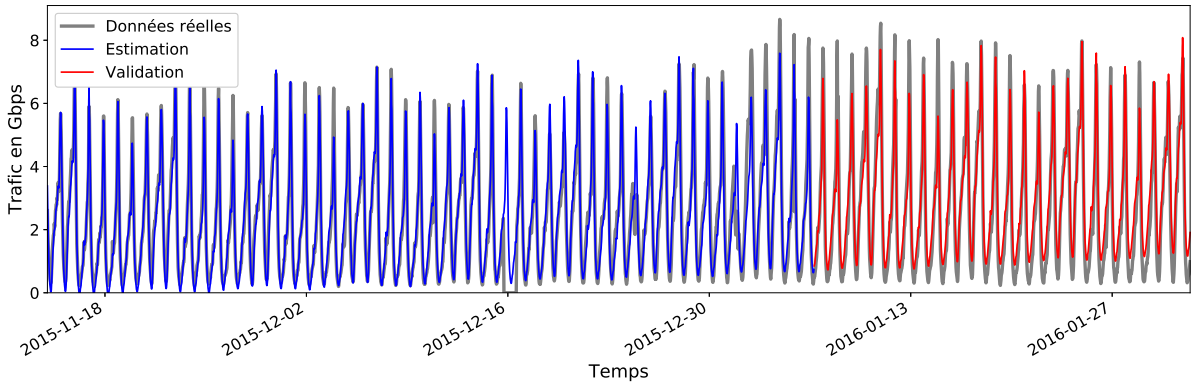
Les données utilisées ici sont identiques à celles utilisées pour la modélisation pour la méthode des séries temporelles (DHR) et reprennent la même différenciation des données d'estimation et de validation. Les résultats de cette modélisation sont représentés aux figures 4.6, 4.7 et 4.8 pour les AS Netflix, HINET et LU-CIX respectivement.

Les résultats de la prédiction (sur les données n'ayant pas servies à l'apprentissage du modèle) pour les trois AS étudiés sont données dans le tableau 4.2, pour différentes périodes d'échantillonnages (5, 15, 30 et 60 minutes). Le comportement prédictif des modèles, quel que soit l'AS, est de très bonne qualité pour toutes les périodes d'échantillonnage représentées (de 5 minutes à 60 minutes).

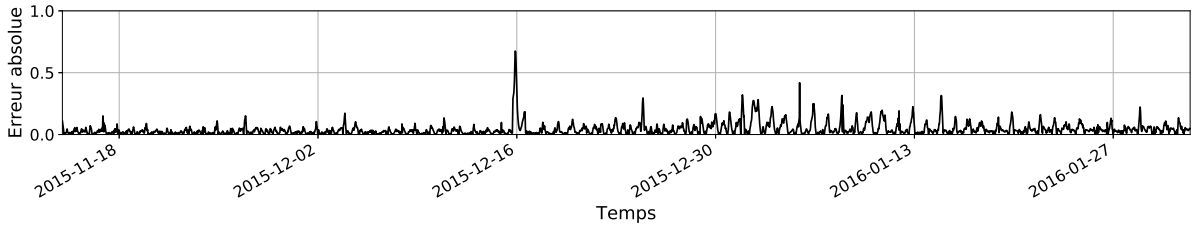
Échantillonnage de $\mathcal{D}_N$	Netflix (AS 2906)	Hinet (AS 3462)	LU-CIX (AS 16509)
5 min	91,7 %	93,2 %	88,5 %
15 min	92,7 %	94,2 %	89,6 %
30 min	92,6 %	94,7 %	90,4 %
60 min	91,9 %	94,8 %	91,1 %

TABLE 4.2 – Précision des prédictions du modèle $\mathcal{M}_{GPR}$ pour différents AS et différents échantillonnages de $\mathcal{D}_N$

On peut toutefois remarquer que le modèle semble de meilleure qualité pour une période d'échantillonnage de 60 minutes par rapport à celle de 5 minutes (en particulier



(a) Prédiction du modèle GPR sur les données de Netflix



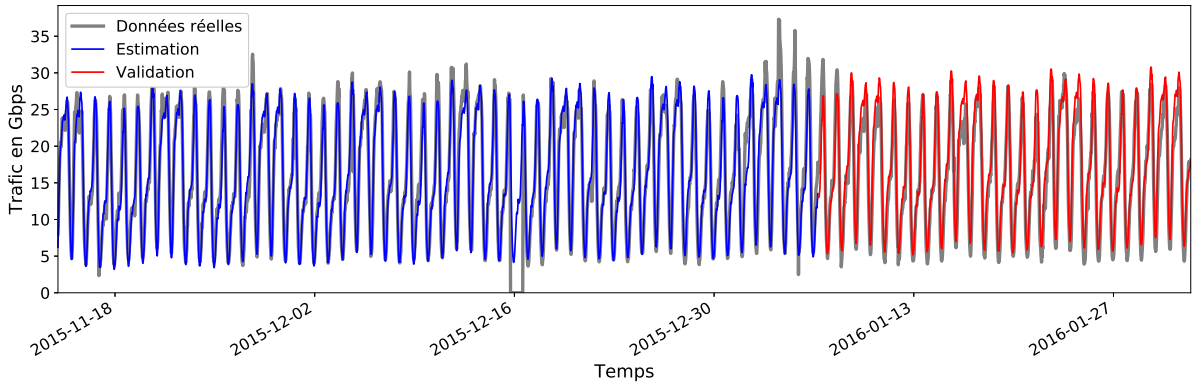
(b) Erreur absolue du modèle GPR sur les données de Netflix

FIGURE 4.6 – Résultat de modélisation GPR sur les données de Netflix échantillonnées à 30 min

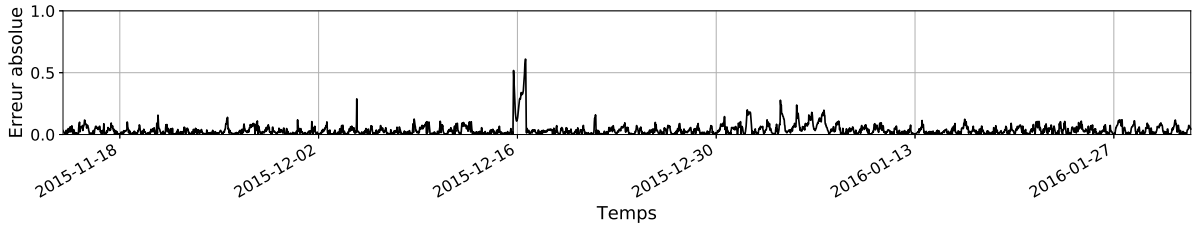
pour les AS Lucix et Hinet). Ceci s'explique facilement par l'analyse des données aux différentes périodes d'échantillonnage. Les figures 4.9a et 4.9b représentent la prédiction du modèle pour l'AS Hinet pour respectivement 5 minutes et 60 minutes de période d'échantillonnage. On remarque l'effet lissage de l'échantillonnage à 60 minutes, quasi inexistant pour l'échantillonnage à 5 minutes. De ce fait, les perturbations sont nettement plus présentes à 5 minutes qu'à 60 minutes. Ainsi le modèle à 5 minutes est plus éloigné des mesures que celui à 60 minutes.

Dans notre cas d'étude, le choix de la période d'échantillonnage des données ne se fait donc pas par rapport à la qualité de la prédiction mais bien par rapport à l'utilisation finale du modèle. L'objectif étant de détecter des anomalies, il est important d'observer les données en quasi continu, de ce fait, on choisira dans la suite de l'étude un échantillonnage rapide, de 5 minutes afin de ne pas passer à côté de certains évènements.

D'autre part, un autre élément de décision important (pour une mise en production future) est le temps de calcul de ces modèles, ceux-ci sont indiqués dans la table 4.3). Il est judicieux de réduire la taille du dataset d'estimation si l'on souhaite utiliser un échantillonnage de 5 minutes. C'est pourquoi nous utiliserons des jeux de données d'estimation de quatre semaines dans l'application de cette méthode.



(a) Prédiction du modèle GPR sur les données de Hinet



(b) Erreur absolue du modèle GPR sur les données de Hinet

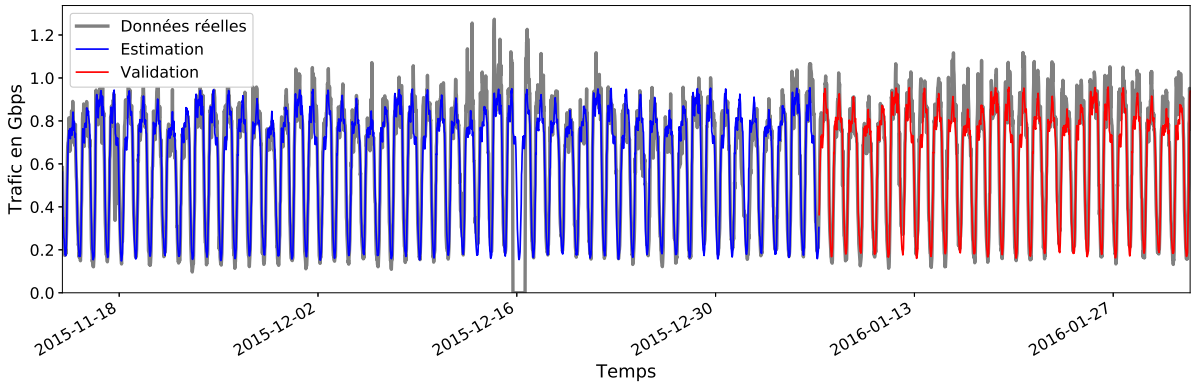
FIGURE 4.7 – Résultat de modélisation GPR sur les données de Hinet échantillonnées à 30 min

4.5 Comparaison : DHR vs GPR

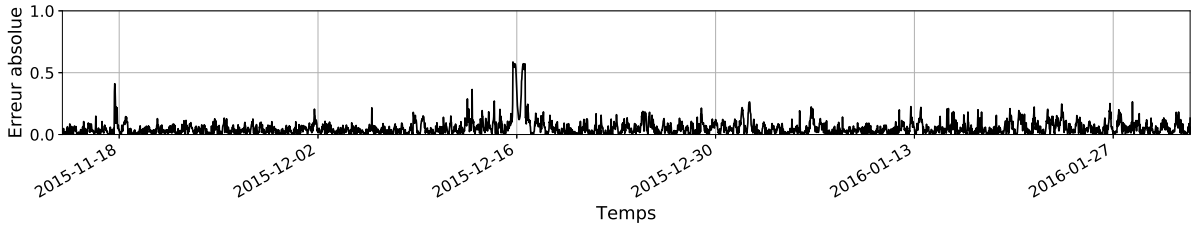
Pour commencer, il est intéressant de faire des commentaires sur la précision des modèles obtenus par les deux méthodes. Elle est calculée sur la base de la prédiction du modèle estimé sur la partie non utilisée pour l'apprentissage et qui n'est pas non plus exempte de défauts. Plus particulièrement, certains points sont manquants et il existe des artefacts de trafic dans cette partie du jeu de données. Ces artefacts sont facilement reconnaissables sur les graphiques montrant l'erreur des modèles (voir figures 4.3b et 4.6b), ils apparaissent sous forme de pics.

Échantillonnage de $\mathcal{D}_N$	Taille de $\mathcal{D}_N$	Temps d'estimation	Temps de prédiction
5 min	23000	5 min	4 min
15 min	7650	20 s	10s
30 min	3800	3 s	1s
60 min	1900	< 1 s	< 1 s

TABLE 4.3 – Temps de calcul du modèle $\mathcal{M}_{\text{GPR}}$ pour différents échantillonnages de $\mathcal{D}_N$



(a) Prédiction du modèle GPR sur les données de LU-CIX



(b) Erreur absolue du modèle GPR sur les données de LU-CIX

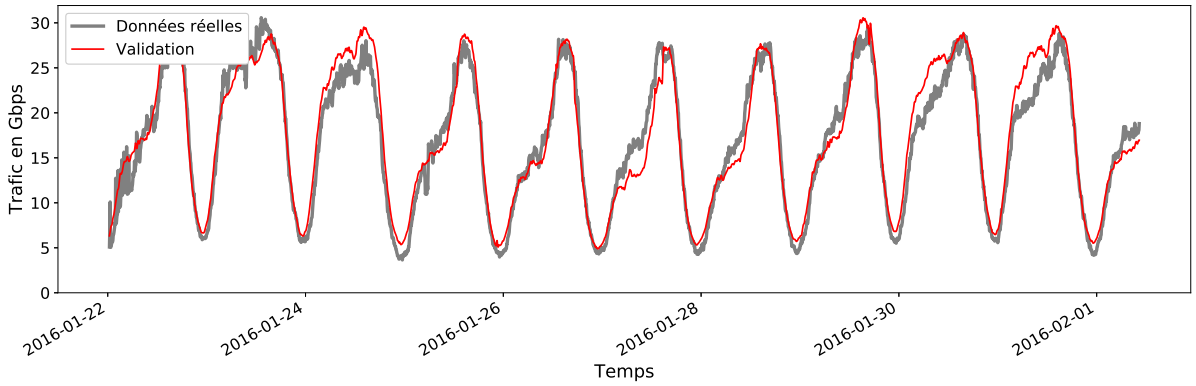
FIGURE 4.8 – Résultat de modélisation GPR sur les données de LU-CIX échantillonnées à 30 min

Une différence notable à préciser avant de continuer plus en détails dans la comparaison des deux méthodes est que le modèle DHR a été entraîné sur des données nettoyées et préparées, contrairement au modèle GPR qui inclut un filtrage des données intrinsèques à la méthode.

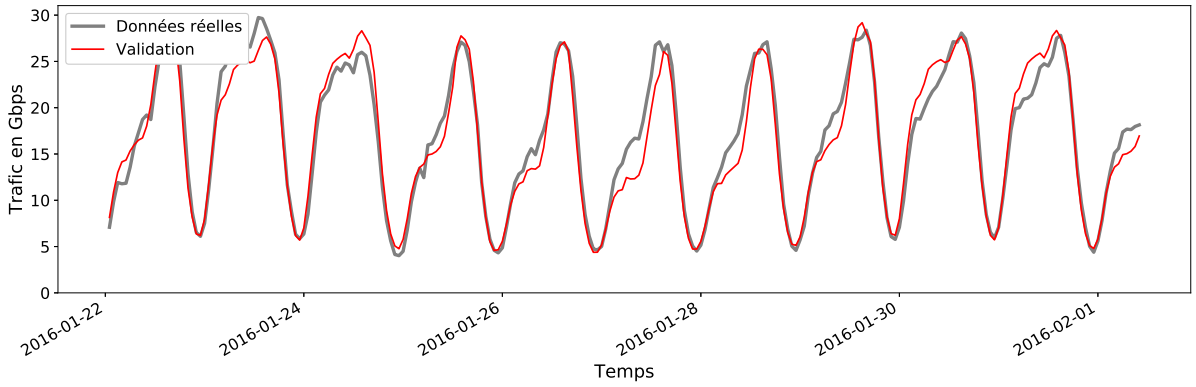
4.5.1 Qualité de l'estimation

Nous allons dans un premier temps nous intéresser à la validation du modèle sur la partie des données ayant servi à l'entraîner afin de s'assurer de l'estimation correcte du processus. Une différence entre les deux méthodes est déjà présente aux figures 4.3 et 4.6 ; on observe que le modèle DHR arrive à prendre en compte et à intégrer les différences de trafic lors des jours de Noël et de la Saint-Sylvestre (voir section 3.2.3). En comparaison, l'autre méthode essaie d'appliquer le même trafic que les autres jours sans prendre en compte les variations.

De la même manière, sur la dernière partie du dataset de validation, on observe une augmentation du trafic de Netflix. Comme précédemment, le modèle de séries temporelles, contrairement au modèle GRP, arrive à intégrer ce changement. Cependant, cela engendre un décalage sur la suite de la prédiction et entraîne une erreur non négligeable.



(a) Prédiction du modèle GPR sur les données de Hinet avec un échantillonnage à 5 min



(b) Prédiction du modèle GPR sur les données de Hinet avec un échantillonnage à 60 min

FIGURE 4.9 – Prédiction du modèle GPR sur les données de Hinet avec un échantillonnage à 5 (a) et 60 min (b)

Ceci met en évidence une différence notable entre les deux méthodes, le modèle GPR semble moins perturbé par des variations de courte durée et présente donc un caractère prédictif de meilleure qualité. Ceci est un point très positif quant à une utilisation de détection d'anomalies. En effet, cela permet d'éviter une perturbation de la prédiction si des évènements sont parvenus dans la partie des données servant à entraîner le modèle.

4.5.2 Qualité de prédiction

Qu'il s'agisse de l'une ou de l'autre méthode, les résultats obtenus sont très encourageants (voir tables 4.1 et 4.2). Cependant, la méthode GPR semble montrer de meilleurs résultats dans l'ensemble pour un même échantillonnage de sortie de 30 minutes. En effet, là où la méthode DHR approche les données avec une précision d'entre 75 % et 84 %, le modèle GPR fait mieux avec entre 90 % et 95 % de précision.

On remarque dans la table 4.2 que la précision peut encore augmenter si l'échantillonnage de $\mathcal{D}_N$ est plus proche de la réalité. Cependant, plus l'échantillonnage se rapproche

des données brutes, plus le nombre de points augmente et donc plus le temps de calcul est important. En effet, que ce soit l'une ou l'autre méthode, le temps de calcul n'est pas négligeable et doit être pris en compte dans l'optique d'une utilisation en environnement de production.

De plus, afin de faciliter le déploiement et la maintenabilité en production, l'utilisation d'un langage répandu et ouvert est un plus. La méthode GPR utilisant le langage de développement Python et la librairie open source Scikit-learn [56] est un avantage comparé à l'utilisation de Matlab et de la toolbox CAPTAIN [50] pour la méthode DHR.

Un comparatif des avantages et inconvénients des deux méthodes est dressé dans la table 4.4.

Modèle	$\mathcal{M}_{\text{DHR}}$	$\mathcal{M}_{\text{GPR}}$
Précision	++	+++
Temps de calcul	++	++
Facilité d'implémentation	+	+++

TABLE 4.4 – Comparaison des deux modèles $\mathcal{M}_{\text{DHR}}$ et $\mathcal{M}_{\text{GPR}}$

4.6 Application à la détection d'anomalies

Il est désormais nécessaire de poser des bases et de mesurer l'efficacité de la méthode quant à l'utilisation de ces modèles pour une détection d'anomalies de trafic. Pour cela, nous allons nous fonder sur des données réelles, brutes de tout prétraitement. Il s'agit du trafic du système autonome de Apple (AS 6185) où une explosion du trafic a été remarquée le 13 septembre 2016. En effet, lors de ce jour, Apple a déployé une mise à jour pour la plupart de ses systèmes (MacOS, iOS, AppleTV...) : chaque utilisateur de système Apple s'est connecté au réseau pour télécharger ces mises à jours. Ceci a eu un impact non négligeable sur le trafic. Celui-ci variant autour de 2 Gbps en pic en temps normal, s'est retrouvé au dessus de 6 Gbps pendant plusieurs jours, comme le montre la figure 4.10.

Un évènement de ce type peut mener à des congestions sur le réseau et donc perturber les autres trafics passant sur les mêmes liens. Il est donc important de pouvoir les détecter au plus tôt afin de prendre des mesures pour contourner les effets négatifs de ceux-ci.

Nous avons appliqué la méthode GPR développée dans la section 4.4 sur les données d'Apple, en utilisant comme jeu d'entraînement les données précédant l'évènement réseau et en calculant des prédictions pour toute la plage de données. Les résultats sont illustrés à la figure 4.11a. On peut observer plusieurs points importants :

— le modèle parvient à capturer l'effet périodique hebdomadaire où le trafic semble suivre une vague ;

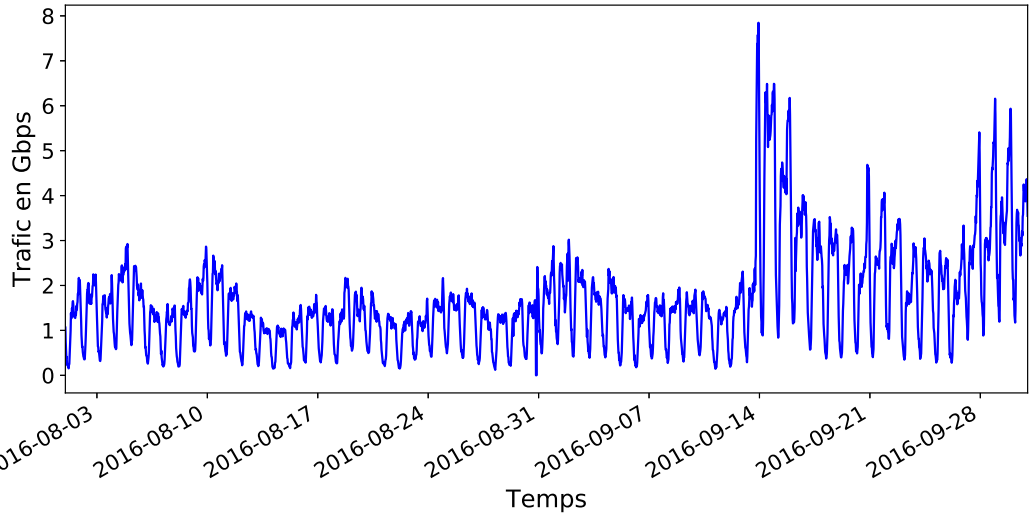


FIGURE 4.10 – Trafic d'Apple avant et après le déploiement de plusieurs mises à jours

— la prédiction du modèle n'est pas perturbée par des valeurs aberrantes (« outliers ») comme celle observée le 30 août, même sans prétraitement ;
— la prédiction du modèle sur les données de tests (n'ayant pas servi à entraîner le modèle) semble stable sur la période observée.

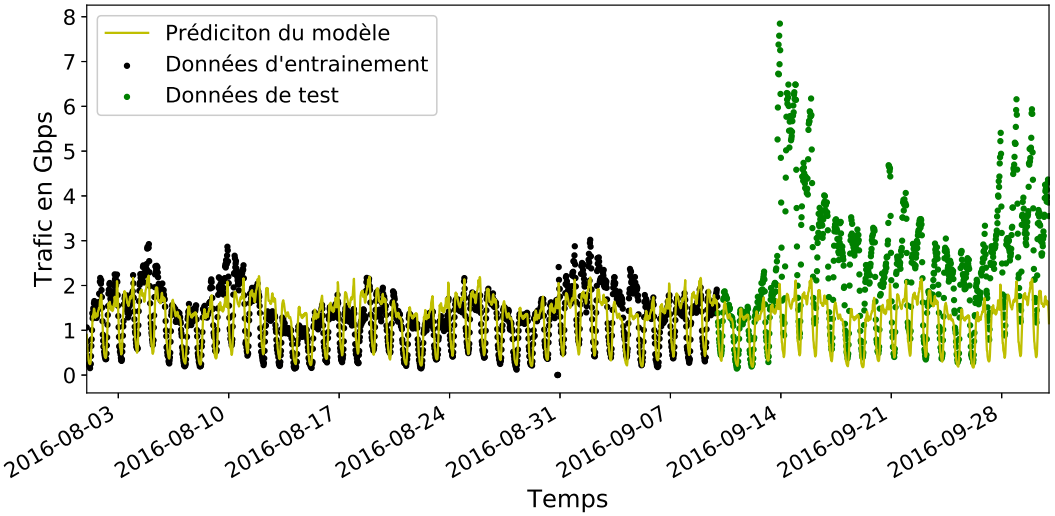
Les résultats obtenus sont très intéressants et permettent l'utilisation du modèle à des fins de détection. Pour ce faire, il est possible dans un premier temps, d'obtenir l'écart-type de la prédiction à l'aide de la fonction `predict`. L'écart-type est largement dépendant de la valeur estimée pour la fonction de covariance du bruit blanc. Cette dernière est estimée à partir d'une campagne de tests par essai-erreur [13]. L'écart-type et l'erreur (équation (4.14)) de la prédiction sont représentés dans la figure 4.11b. Il est donc aisé de comparer l'erreur du modèle avec des valeurs seuils (ici 1, 2 ou 3 écart-types représentés par les courbes en pointillés rouges sur la figure 4.11b) et de définir une anomalie réseau par le dépassement de ces niveaux.

$$\text{Erreur}_k = \frac{abs(y_k - y_{pred_k})}{max(y)} \tag{4.14}$$

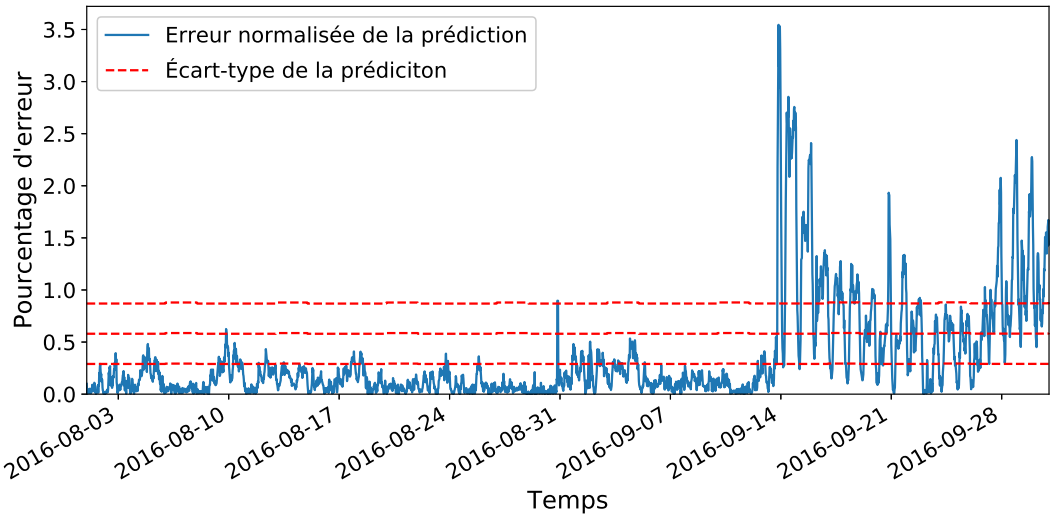
L'étape la plus délicate de cette étude est de définir ces valeurs qui serviront de seuils à nos détections. Plusieurs méthodes sont développées dans la littérature et pourraient être appliquées à ce cas.

Cependant, dans un premier temps, on peut remarquer que la détection d'anomalies fondée sur le dépassement d'un modèle de comportement dit normal (modèle prédictif plus écart-type) fournit de bons résultats. En effet, dans le cas d'Apple, bien que le

13. D'autres méthodes peuvent être utilisées pour affiner cette estimation, comme celles du maximum de vraisemblance par exemple



(a) Modélisation du trafic venant de l'AS 6185 (Apple)



(b) Erreur normalisée de la prédiction

FIGURE 4.11 – Application de la méthode GPR à la détection d'anomalies sur le trafic de l'AS 6185 (Apple)

trafic global reste relativement faible (autour de 3 Gbps), le comportement du trafic sort de la zone acceptable définie par le modèle et déclenche une alarme à l'équipe en charge du réseau. Elle est alors en mesure de prendre des actions sur le réseau pour remédier au problème issu de l'évènement en cours. Dans ce cas, il s'agirait d'absorber le trafic supplémentaire en reroutant une partie du trafic sur une autre interface d'entrée du réseau. Cet exemple illustre le fait que cette approche peut être utilisée dans un contexte de détection d'anomalies.

4.7 Implémentation en Python de la méthode GPR

Cette section présente la procédure mise en œuvre utilisant la librairie Scikit-learn [56] afin de modéliser le trafic, grâce à la méthode des processus gaussiens [47]. Les outils suivants doivent être installés au préalable :

— Python (version 3.x) ;
— `SciPy` [57] (dont `NumPy` et `Pandas`) ;
— `Scikit-learn` [56].

4.7.1 Importation des librairies

Afin de traiter les données, les librairies `Pandas` et `Numpy` incluses dans `SciPy` [57] sont utilisées. Ce sont de puissants outils de gestion de matrices et de données. Nous importons aussi la partie des processus gaussiens qui servira par la suite.

```
import pandas as pd
import numpy as np

from sklearn.gaussian_process import GaussianProcessRegressor
from sklearn.gaussian_process.kernels \\
    import RBF, ExpSineSquared, WhiteKernel
```

4.7.2 Importation des données en Python

Les données échantillonnées à T_e doivent être enregistrées dans un fichier de type `csv` ayant pour colonnes : timestamp, qui servira d'index, et données. Un dataframe `Pandas` est une structure de données à deux dimensions et sert de base pour la suite de l'étude.

```
data = pd.read_csv('data.csv',
                   index_col=0,
                   names=['timestamp', 'données'],
                   parse_dates=True)
```

4.7.3 Mise en forme des données

Les données sont ici en bits par période d'échantillonnage T_e, nous allons les transformer en Gbps avant la phase de modélisation. De la même manière, afin de créer le vecteur d'entrée x, nous allons transformer les dates en entiers représentants des secondes. Pour

cela, une nouvelle colonne est créée (appelée 'ts_sec') en convertissant les timestamps de type datetime en entier. Par défaut, la méthode retourne des valeurs en nanosecondes, c'est pourquoi une remise à l'échelle en secondes est réalisée. Dans la suite de l'étude, nous souhaitons utiliser la première moitié des données pour l'estimation et la seconde moitié pour la validation.

```
# Transformation des dates en secondes
data['ts_sec'] = data.index.values.astype(np.int64) // 10**9

# Définition de Te à 5 minutes = 300 secondes
Te = 300

# Transformation des bits par Te en Gbps
data['Gbps'] = data['données'] / (Te * 10**9)

# Détermination de la limite des deux datasets
lim_estimation = int(len(data)/2)
```

4.7.4 Configuration de la fonction de covariance

Afin de configurer le noyau du modèle, nous allons nous appuyer sur la méthode d'implémentation développée dans la section 4.4. Pour cela, nous créons quatre noyaux :

- un noyau RBF à longue échelle de temps (*lenght_scale*) servant de tendance au modèle ;
- un premier noyau représentant la périodicité journalière ;
- un second noyau pour la périodicité hebdomadaire ;
- un noyau blanc paramétrant le bruit du système.

Une combinaison de ces quatre noyaux est ensuite réalisée avant de créer notre modèle. Le paramètre *alpha* correspondant à la modélisation du bruit est forcé à zéro car le bruit est déjà pris en compte par le noyau k_4. De même, les hyperparamètres ayant été définis et expliqués plus tôt, nous désactivons l'optimisation des paramètres proposée par la librairie.

```
# Création de la nomenclature du modèle
## Tendance sur une année
## (365 jours * 24 heures * 60 minutes * 60 secondes)
k1 = RBF(length_scale=365*24*60*60)
## Périodicité journalière
## (24 heures * 60 minutes * 60 secondes)
k2 = ExpSineSquared(length_scale=0.05, periodicity=24*60*60)
## Périodicité hebdomadaire
## (7 jours * 24 heures * 60 minutes * 60 secondes)
```

```
k3 = ExpSineSquared(length_scale=1, periodicity=288*300*7)
## Bruit blanc
k4 = WhiteKernel(noise_level=0.01)

kernel_gpml = k1 + k2 * k3 + k4

gp = GaussianProcessRegressor(kernel=kernel_gpml,
                              alpha=0,
                              optimizer=None,
                              normalize_y=True)
```

4.7.5 Estimation du modèle

Afin d'estimer le modèle sur les données d'entraînement, nous allons utiliser la méthode `fit` de l'objet `GaussianProcessRegressor` avec la partie des données définie précédemment.

```
# Estimation du modèle
gp.fit(data['ts_sec'].values.reshape(-1, 1)[:lim_estimation],
       data['Gbps'].values.reshape(-1,1)[:lim_estimation])
```

4.7.6 Prédiction et validation

Pour réaliser une prédiction du modèle précédemment estimé, nous utilisons la fonction PREDICT. Celle-ci retourne deux vecteurs, le premier est la prédiction en elle-même. Le second est l'écart-type de la prédiction, calculé et renvoyé seulement si le paramètre RETUR_STD est positionné à VRAI.

```
data['Gbps pred'], data['pred std'] = \\
    gp.predict(data['ts_sec'].values.reshape(-1, 1), return_std=True)
```

Nous disposons maintenant, dans notre dataframe, de différentes colonnes contenant les valeurs réelles, la prédiction et l'écart-type du modèle.

4.8 Conclusion

Ce chapitre 4 montre l'application de deux méthodes différentes de modélisation sur les données mesurées sur différents AS du réseau de Post Luxembourg. La première est une

solution développée et utilisée principalement dans le milieu de l'automatique, particulièrement dédiée à l'étude de systèmes périodiques. La seconde est une méthode de machine learning par processus gaussiens qui a été configurée afin de prendre en considération la périodicité du système.

Ces deux approches ont été comparées et l'une d'entre elles, celle des processus gaussiens, a été retenue pour la suite de l'étude pour sa facilité de mise en œuvre et ses résultats encourageants. De plus, le modèle GPR a montré qu'il était utilisable dans un contexte de détection d'anomalies réseau. C'est pourquoi cette méthode sera utilisée pour un développement en production d'une application de détection d'anomalies de trafic. Le chapitre 5 présente cette application.

Chapitre 5

ANODE : ANOmaly DEtection

Sommaire

Ce chapitre est l'aboutissement des différents travaux de la thèse. En effet, les éléments présentés dans les chapitres précédents sont utilisés pour le développement d'un logiciel de détection d'anomalies.

5.1 Présentation

La détection d'anomalies réseau est un domaine de recherche établi et déjà exploité dans l'industrie par différentes solutions de protection contre des attaques par déni de service notamment (par exemple les outils des sociétés Arbor Networks et Huawei). Cependant celles-ci ne sont développées que pour protéger le réseau contre des attaques en volume pouvant saturer l'infrastructure. En effet, ces solutions se fondent sur des volumes seuils visant à détecter des évènements qui génèrent plus de trafic que la normale, et qui risquent donc de saturer le réseau. Ces techniques sont fondées sur l'exploitation,

comme ici, de statistiques Netflow et d'informations récupérées par le protocole SNMP (voir section 2.1.2).

De ce fait, ces applications, appelées protections *anti-DDoS*, ne détectent pas les évènements réduisant le trafic. De plus, bien souvent, les détections ne se font que pour des seuils fixes et, par conséquent, des attaques se produisant en période de faible trafic peuvent passer inaperçues. Par ailleurs, l'observation (et par la même, la protection) est souvent centrée sur le réseau opérateur ou sur un client. De fait, l'observation de la qualité des connexions vers un AS distant n'est pas prise en compte.

De par toutes ces limitations, ces solutions ne sont pas pleinement exploitables pour explorer le trafic interdomaine et en détecter les anomalies, hors attaques en débit. C'est pourquoi, nous avons choisi de développer notre solution du point de vue du cœur de réseau, afin d'être capable de détecter les évènements au sein du trafic interdomaine de Post Luxembourg.

La détection classique d'attaques ne prend pas non plus en compte les congestions qui ne sont pas locales. En effet, il n'est pas rare qu'un client constatant un souci en se connectant à un service remonte l'information à l'opérateur. En investiguant le problème, on observe que le point bloquant se situe en dehors de notre réseau : une panne sur une liaison entre deux Tier-1, une congestion ... Et de fait, la qualité d'expérience (*QoE*) du client se voit réduite sans que l'opérateur puisse intervenir facilement sur la panne. La seule solution dans ces cas particuliers est de rerouter le trafic par un autre chemin, mais pour ce faire, il est nécessaire de pouvoir détecter ces points bloquants.

La solution logicielle, ANODE (pour *ANOmaly DEtection*) développée dans cette thèse, a pour but d'être utilisée par l'équipe en charge de la surveillance du réseau afin de réagir et de prendre des mesures correspondant aux problèmes qu'elle détecte. Elle utilise ce qui a été présenté dans les précédents chapitres de cette thèse, à savoir :

— l'utilisation de données de trafic récoltées via Netflow ;
— une modélisation de comportement de trafic des AS les plus importants par la technique GPR ;
— une détection d'anomalies de trafic réel vis à vis des prédictions des modèles de comportement.

Le dernier point n'a pas été analysé en détail jusqu'à présent et une première approche est proposée dans ce chapitre.

Plusieurs études présentées dans la littérature donnent des pistes intéressantes pour améliorer la détection d'anomalies dans un réseau.

Parmi celles-ci, Cormode *et al.* [58] proposent une méthode afin de faire ressortir des informations pertinentes du trafic. Cette étude est fondée sur l'agrégation des flux IP sur une heure, et ces objets, appelés *deltoid*, sont soit : une différence absolue d'une heure à la suivante ; une différence relative d'une heure à la suivante ; ou une variance au cours du temps. Cette étude permet de poser un cadre pour définir ce qui est intéressant à

envoyer à l'équipe en charge du réseau et ce qui ne l'est pas afin de ne pas la surcharger d'informations.

Au sein de l'étude [59], les données réelles sont abandonnées au profit de *sketch*, représentations probabilistes des jeux de données. Sur ceux-ci, différentes méthodes de séries temporelles sont appliquées pour en détecter, comme ici, les variations.

De la même façon, Salem *et al.* proposent aussi dans [60] une étude basée sur des *sketch* pour détecter des anomalies de trafic. Une fois l'anomalie détectée la clé de stockage, représentant la cible ou l'attaquant, est retrouvée grâce à un algorithme spécifique.

Lakhina *et al.* étudient dans [61] l'entropie des couples adresses et ports des flux afin de détecter des anomalies. Mazel dresse un état des différentes méthodes de détection d'anomalies dans [62] tout en proposant sa propre approche fondée sur du machine learning non supervisé.

Brutlag propose dans [63] une approche très similaire à celle proposée dans la suite de ce document en modélisant le trafic réseau à l'aide de méthodes de séries temporelles. La partie détection d'anomalies se fait ensuite en appliquant un intervalle de confiance autour de la prédiction. La détection est réalisée sur la base de plusieurs valeurs aberrantes, sur une fenêtre glissante.

En se fondant sur les différents travaux précédemment cités, nous avons proposé une solution originale adaptée au cas de l'analyse du trafic de Post Luxembourg.

Nous avons remarqué dans le chapitre 3 que l'ensemble des AS avaient un comportement reproductible dans le temps. De ce postulat, nous avons décidé d'estimer des modèles de comportement afin de déterminer le comportement normal de chaque AS. Ayant accès aux données de trafic en temps réel, nous pouvons alors détecter une anomalie lorsque cette valeur sort d'une zone définie comme normale de comportement.

La solution logicielle est dédiée à l'observation et l'analyse du trafic par AS, mais dans un souci d'évolutivité, nous avons choisi de coder une analyse par « Monitored Object » (abrégé en MO) plutôt que par AS ; un MO pouvant représenter un AS, un groupement d'AS ou d'autres éléments représentatifs du trafic réseau. Ceci permet d'étendre l'étude à d'autres objets qui pourraient nous sembler pertinents. Ces autres objets seront détaillés plus loin dans ce chapitre.

5.2 Développement

5.2.1 Architecture

Lors du développement d'une application, il est important de poser le cadre de celle-ci en prenant en compte plusieurs critères :

— quel est le but de cette application ;
— qui utilisera cette application ;
— qui maintiendra cette application dans le futur ;
— quel est l'avenir de cette application.

Des différentes réponses qui découlent de ce raisonnement, plusieurs choix devront être réalisés. Un point très important à prendre en compte lors du développement d'une application est l'utilisateur final de cette application, à savoir ici, l'équipe en charge du réseau. Celle-ci n'ayant pas les connaissances nécessaires pour l'estimation des modèles, l'application doit être totalement autonome sur ce point. De ce fait, les paramètres de synthèse des méthodes d'estimation des différents modèles seront fixés au préalable. Cependant, il doit être possible d'augmenter ou de diminuer la base de MO à observer facilement.

Il a aussi été décidé de séparer l'application en deux parties. La première sert à récupérer les données et à estimer les modèles de comportement. Quant à la seconde, elle est dédiée à la comparaison entre le modèle et les données réelles afin de détecter des anomalies.

Plus précisément, l'architecture de l'application est détaillée par la figure 5.1 :

— les statistiques Netflow de trafic venant du réseau sont acheminées vers un collecteur Netflow ;
— ces données sont ensuite utilisées par la partie « learning » de l'application afin de créer les modèles de comportement pour chacun des MO configurés ;
— ces modèles sont ensuite stockés (sous forme de fichiers csv) afin d'être disponibles pour la seconde partie de l'application ;
— la détection se fait dans un second temps en comparant les données réelles du collecteur Netflow avec les prédictions de comportement réalisées en amont ;
— en cas d'anomalie, une alarme est envoyée via le serveur à l'équipe en charge du réseau.

Ce choix de séparation de l'application en deux parties distinctes a été réalisé afin de la rendre plus légère en terme de consommation de processeur. En effet, la détection d'anomalies réalisée à fréquence élevée (toutes les 30 minutes ici) n'a pas besoin de recréer un modèle à chaque exécution. Celui-ci pouvant être calculé à une fréquence inférieure à la journée.

Pour assurer le maintien de l'application par une équipe de développement interne, il a aussi été décidé de travailler avec le langage de programmation Python, déjà utilisé pour d'autres outils internes. La méthode GPR présentée au chapitre 4 est codée dans ce langage grâce à la libraire `Scikit-learn` [56]. L'utilisation du langage Python et de librairies open-source sont un avantage pour cette application.

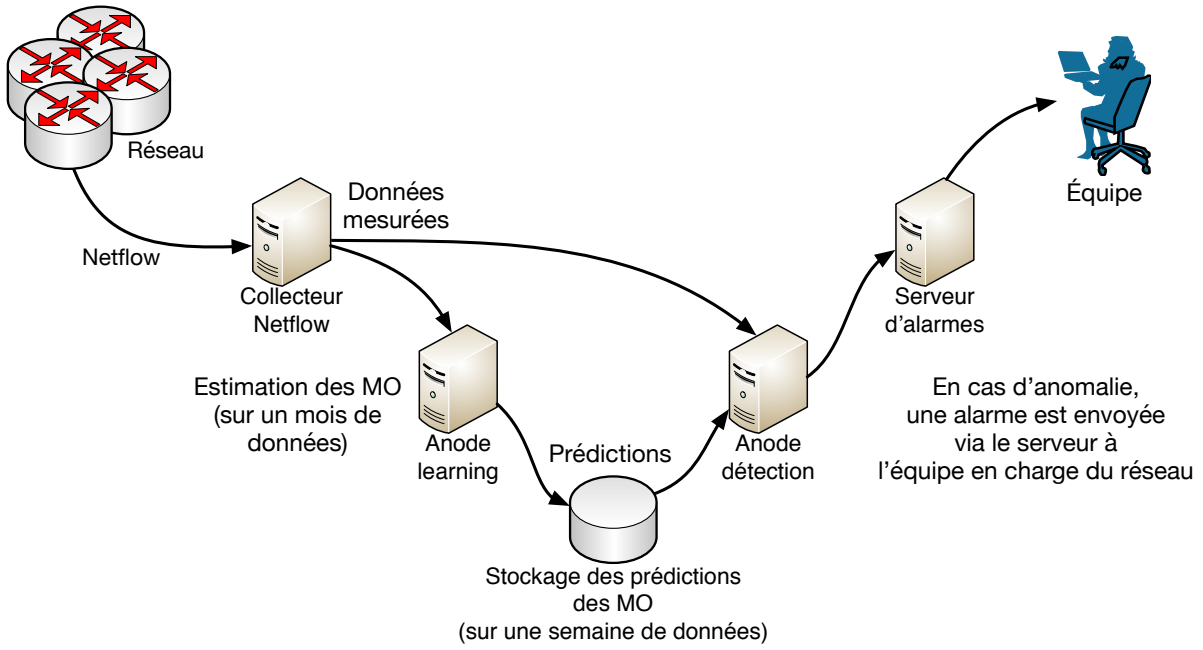


FIGURE 5.1 – Architecture de l'application Anode

5.2.2 Récolte des données

Les données réelles de trafic sont obtenues par un collecteur Netflow déjà utilisé dans l'environnement de production dans le cadre d'une solution de détections d'attaques. Il a été préféré de mutualiser la collecte afin de réduire le besoin de ressources et de maintenance. Nous récupérons les statistiques Netflow de trafic pour un MO configuré au préalable par le biais d'un appel sur une API web. Le serveur renvoie un tableau de séries temporelles sur la période demandée.

5.2.3 Estimation des modèles de comportement

L'application met en œuvre la méthode GPR présentée dans le chapitre 4 avec une période d'échantillonnage des données d'entrée réglée sur 5 minutes. En effet, le but de cette application n'est pas de prévoir l'état du trafic à long terme mais de détecter des anomalies de courtes durées. C'est pourquoi nous avons fait le choix d'utiliser une fenêtre glissante d'un mois de données pour estimer le modèle afin de prédire une semaine de comportement.

5.2.4 Remontée d'alarmes

Au sein de l'environnement de production, l'équipe utilise une solution de gestion des alarmes gérée par deux applications : Logstash et Kibana. Cette infrastructure permet

une centralisation des alarmes de tous les systèmes vers un serveur Logstash qui filtre et interprète chacun des messages avant de les stocker. Un serveur Kibana est utilisé pour visualiser ces messages systèmes de façon claire. Il a donc fallu développer l'application en pensant à l'intégration avec ce système d'alarme.

Une discussion avec l'équipe a permis de définir quelles informations étaient importantes à afficher dans l'alarme afin d'amener une compréhension de l'anomalie ayant cours. Ces informations sont recensées dans la table 5.1. Plusieurs graphiques de trafic sont créés afin de donner des vues à la fois globales et centrées du problème.

Information	Description
MO	ID et nom du MO
data_type	Type de données (bps, pps...)
direction	Direction des données (vers, de)
type	Indique si le trafic est au-dessus ou en-dessous de la prédiction
timestamp_start	Heure à laquelle la détection a commencé
timestamp_end	Heure à laquelle la détection s'est terminée
shift	Différence entre le trafic réel et la prédiction
shift_perc	Différence en pourcentage entre le trafic réel et la prédiction
occurence	Nombre d'anomalies détectées sur la période de détection
description	Texte descriptif de l'anomalie
graph	Graphique montrant le trafic réel et la prédiction

TABLE 5.1 – Liste des informations disponibles dans un rapport d'anomalie

5.2.5 Algorithmes de l'application

Les différentes étapes constituantes de ces deux parties de l'application sont détaillées pour la partie learning par la figure 5.2 ainsi que par la figure 5.3 pour le module de détection. L'estimation des modèles de MO est réalisée une fois par semaine glissante à partir de la collecte d'un mois de données. Puis ce modèle est utilisé pour prédire le comportement normal du MO, toutes les semaines. Ensuite l'algorithme de détection d'anomalies est lancé toutes les 30 minutes.

5.3 Utilisation du logiciel ANODE

La configuration de l'application pour un *Monitored Object* est détaillée. Ensuite, une détection en utilisation réelle est présentée.

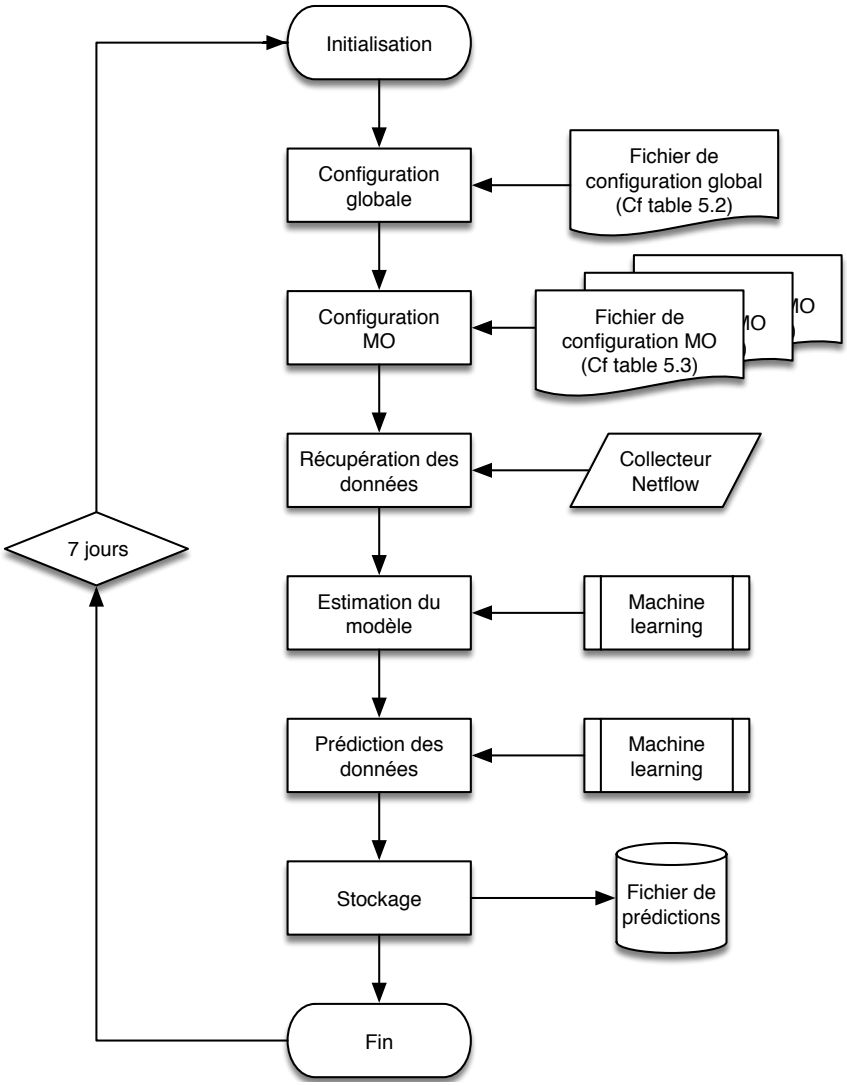


FIGURE 5.2 – Algorithme de l'application de learning pour l'estimation et la prédiction

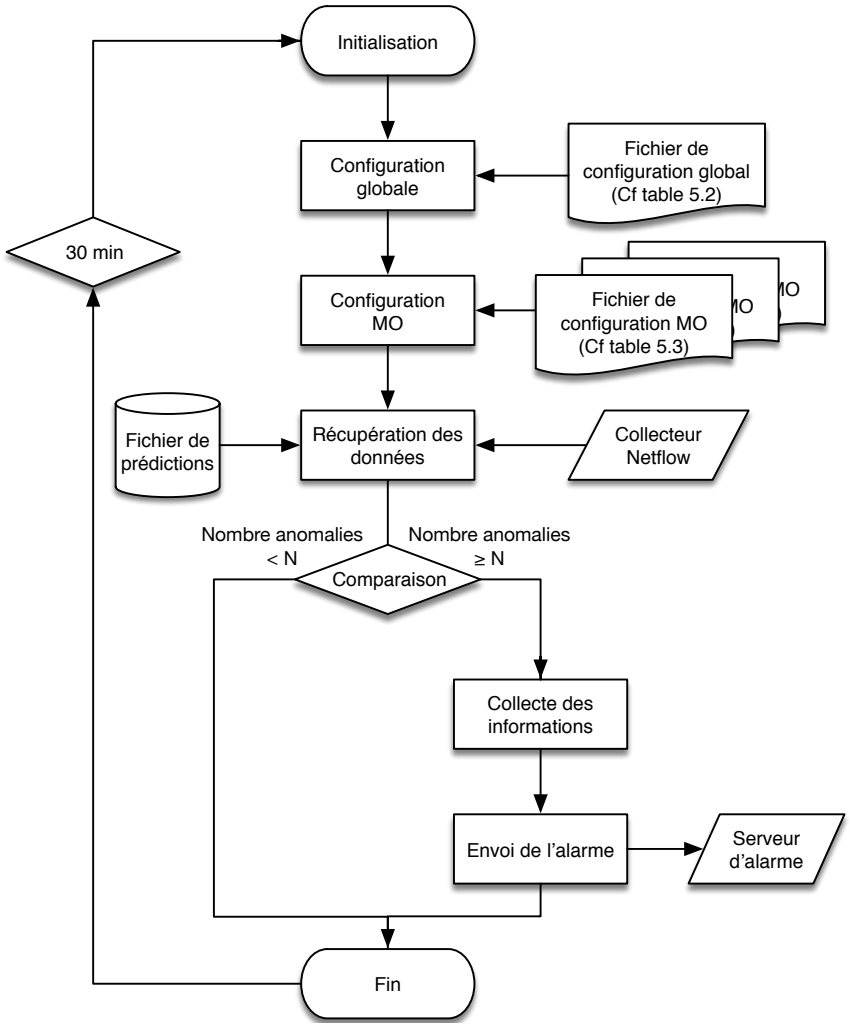


FIGURE 5.3 – Algorithme de l'application de détection

5.3.1 Configuration

La configuration de l'application se fait à plusieurs niveaux. Tout d'abord, il s'agit de régler les paramètres de l'application en elle-même, puis de configurer chaque *Monitored Object* avec ses propres paramètres.

Dans un premier temps, il est donc nécessaire de définir des paramètres de base pour l'application afin qu'elle puisse fonctionner (voir table 5.2 et algorithme à la figure 5.2 et 5.3). C'est ici que nous définissons le lieu où les données de trafic sont disponibles. En l'occurrence nous utiliserons ici un collecteur Netflow.

Catégorie	Paramètre	Description
Source	type	Source des données (fichier ou serveur)
	path	Dossier où sont disponibles les données
	host	Lien du serveur de données
	user	Nom d'utilisateur pour se connecter au serveur de données
	password	Mot de passe pour se connecter au serveur de données
Sortie des modèles	path	Lien du dossier où stocker les prédictions du modèle
	log	Activation du journal d'évènements (*logging*)
	log_path	Lien vers lequel stocker le journal d'évènements
Sortie des alarmes	type	Destination des alarmes (fichier ou serveur)
	path	Lien du fichier où enregistrer les alarmes
	host	Lien du serveur où envoyer les alarmes

TABLE 5.2 – Paramètres globaux de l'application

Du fait de son infrastructure en deux parties, les prédictions de chaque modèle doivent être stockées sur le disque avant d'être utilisées par la partie détection de l'application. Le choix de ce chemin de stockage est disponible via un paramètre. De la même manière, si une anomalie est détectée, il est possible de choisir la destination de l'alarme.

Une fois l'application paramétrée pour fonctionner normalement, il est nécessaire de lui fournir une liste de *Monitored Object* (ou « MO ») à observer. Pour chacun de ceux-ci, un fichier de configuration est rempli afin de définir leur paramètre propre et indépendant. Ces paramètres sont définis à la table 5.3. Au sein du serveur de collection Netflow, il est possible de définir des MO en fonction de deux types : voisin ou sous-réseau.

Voisin : les voisins sont des réseaux externes pouvant être définis par un ou plusieurs numéro(s) d'AS. Ils permettent de récupérer des informations sur la quantité de trafic allant vers et venant de la liste d'AS fournie, en bits par seconde et en paquets par seconde. Exemples de voisins : Netflix, Apple...

Sous-réseau : les sous-réseaux sont des sous-parties du réseau interne. Ils peuvent être définis avec plusieurs valeurs (numéro d'AS, préfixes IP, BGP AS path, BGP

Catégorie	Paramètre	Description
Monitored Object	name	Nom du MO
	id	ID du MO sur le serveur de données
	type	Type de MO : voisin ou sous-réseau
	cols	Colonnes de données à utiliser par l'application (ex : 'from bps', 'to pps'...)
Modèle	model_name	Nom du modèle à utiliser (ici : 'GPR')
	learn_wind_pts	Nombre de points à utiliser pour l'entraînement
	fc_wind_pts	Nombre de points à prédire
Paramètre GPR	return_std	Configurer le modèle pour retourner l'écart-type de la prédiction
Détection	last_pts	Nombre de données à utiliser pour la détection
	pts_trigg	Nombre d'anomalies avant de déclencher une alarme
	overwrite_wind	Écraser la valeur d'écart type avec des nouvelles valeurs
	window_perc	Définition de la fenêtre d'erreur en pourcentage

TABLE 5.3 – Paramètres de chaque *Monitored Object*

community). Contrairement aux voisins, il est possible de récupérer, en plus des données de trafic en bits et en paquets par seconde, le nombre d'IP actives à l'intérieur du sous-réseau et le nombre d'IP avec lesquelles ce sous-réseau communique. Exemples de sous-réseaux : les clients résidentiels, le réseau mobile, un opérateur tier-3 à qui Post propose du transit...

Après configuration, chaque MO obtient un identifiant unique. C'est par celui-ci qu'ils seront identifiés dans l'application, le nom pouvant être modifié.

On peut donc voir que le nombre de colonnes disponibles pour la surveillance de comportement varie suivant le type de MO. Il est aussi possible de ne choisir que certaines colonnes et de ne pas réaliser la détection d'anomalies sur l'ensemble du jeu de données.

Pour l'instant, un seul modèle est utilisable, mais l'application a été pensée pour être évolutive. C'est pourquoi il est possible de spécifier un nom de modèle à utiliser, qui renvoie à un algorithme disponible dans l'application à utiliser pour la modélisation.

La catégorie de paramètre suivante concerne la configuration du modèle GPR. Il est possible ou non de retourner l'écart-type de la prédiction du modèle. Il est à noter que cet écart-type est majoritairement dépendant de la valeur de l'hyperparamètre σ_N^2 de la fonction de covariance du bruit (voir section 4.4). La prédiction du modèle (et si demandé, son écart-type) est stockée sur le disque sous forme d'un fichier csv.

La dernière catégorie de paramètre pour chaque MO correspond à la configuration de la détection d'anomalie. Le nombre de points à prendre en compte dans le passé jusqu'à

l'instant où la détection commence est paramétrable. Nous avons choisi d'observer les anomalies sur des fenêtres de 30 minutes, ce paramètre est donc configuré dans notre cas à la valeur de 6 T_e (T_e étant de 5 minutes). Il est aussi possible, au moment de la lecture du fichier csv de la prédiction, de changer la fenêtre d'erreur (définie précédemment par l'écart-type) par valeur définie par un pourcentage de la valeur de la prédiction. On peut également définir le nombre d'anomalies de trafic nécessaire dans l'observation pour déclencher une alarme. Ceci dans le but d'éviter de détecter des faux positifs dus à des artefacts de mesures.

5.3.2 Exécution

Chaque partie de l'application (modélisation et détection) est lancée comme tâche récurrente. La partie modélisation pouvant être gourmande en temps de calcul, il est possible de ne la lancer qu'une fois par semaine et en prédisant le trafic sur une semaine. De l'autre côté, l'application de détection est appelée toutes les 30 minutes (la détection observant les 30 dernières minutes de trafic).

Si la partie détection observe un nombre d'anomalies supérieur à la valeur définie dans la configuration (par le paramètre « pts_trigg ») dans le dataset, elle crée un rapport contenant plusieurs valeurs qui serviront à l'équipe en charge du réseau pour la compréhension du problème (voir table 5.1).

5.4 Exemple d'anomalie détectée

Différents *Monitored Object* (AS voisins et sous réseaux) ont été configurés et observés pendant plusieurs semaines. La figure 5.4 montre les différentes alarmes survenues pendant une semaine sur les différents MO configurés. La figure du haut représente un graphique cumulé des alarmes en bits par seconde (bps) et celui du dessous en paquets par seconde (pps). On remarque que beaucoup d'alarmes sont levées la nuit, il s'agit souvent de faux positifs. Cela vient du mode de détection, loin d'être optimal, qui compare le trafic réel avec la prédiction ± un pourcentage de celle-ci. On peut remarquer cependant que les alarmes levées pour une anomalie en bits par seconde se retrouvent souvent en paquets par seconde. L'intérêt d'observer les deux types de mesures (bps et pps) réside dans le fait que certaines attaques ou problèmes ne génèrent que des paquets de petites tailles, mais en grand nombre. Ceux-ci pourraient ne pas être détecter en bps, mais le seront en pps.

On remarque sur la figure 5.4 la présence de nombreuses alarmes sur le MO Apple (correspondant à l'AS 6185) entre le 18 et le 21 janvier. En regardant de plus près la première alarme de la période, on observe une augmentation du trafic de l'AS à partir de 19h50 le 18 janvier. Ces résultats sont particulièrement visibles sur le graphique, de la figure 5.5, extrait de cette alarme. Sur le graphique de la partie supérieure, on observe le

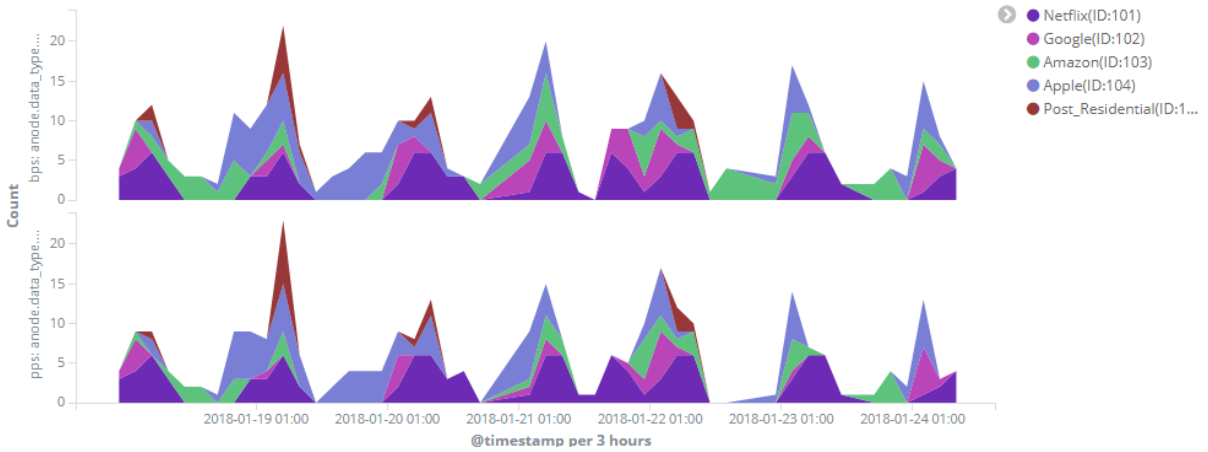


FIGURE 5.4 – Alarmes levées par l'application sur une semaine

trafic réel des trois dernières semaines pour cet AS (en noir) puis la prédiction du modèle pour les 48 prochaines heures (en rouge). On observe que sur la période de détection (bleue sur le graphique de la partie inférieure), le trafic réel double.

L'application ANODE a détecté cette anomalie et a envoyé une alarme accompagnée de la liste d'informations, contenues dans la table 5.4 ainsi que dans la figure 5.5, dans l'objectif d'aider à comprendre l'évènement en cours. L'application envoie aussi une brève description de l'alarme : « 2018-01-18 20:05:00 - 2018-01-18 20:30:00: Traffic bps for Apple (ID:104) is anormal. It should be 4,761,497,935 +- 1,428,449,381 and it is 7,561,084,672 »

Information	Valeur
MO	Apple
data_type	bps
direction	from (trafic venant du MO)
type	overflow (trafic plus important que prévu)
timestamp_start	2018-01-18 20:05:00
timestamp_end	2018-01-18 20:25:00
occurence	6
shift	2 799 586 737 (bps)
shift_perc	159%

TABLE 5.4 – Informations contenues dans l'alarme

Cette anomalie a bien été détectée par l'application ANODE, en revanche elle n'a pas été signalée par les autres systèmes de détection implantés chez Post Luxembourg. Cela provient du fait que le trafic généré est peu élevé comparé aux seuils de détection de ces systèmes.

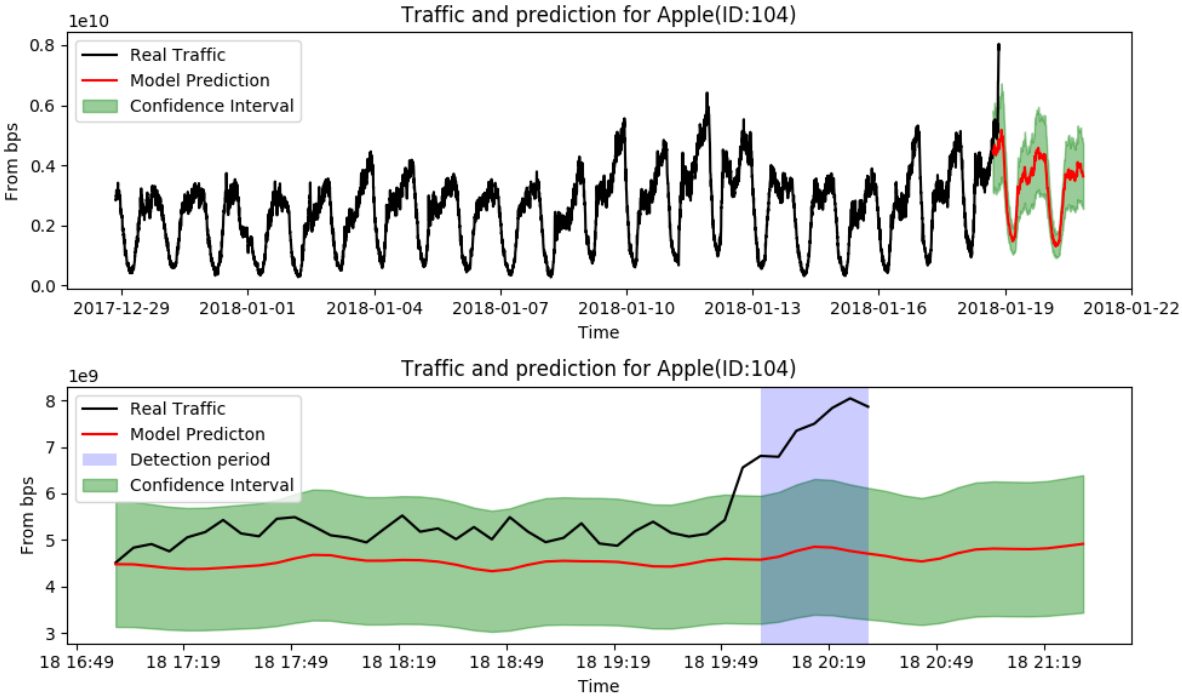


FIGURE 5.5 – Exemple d'anomalie détectée sur le trafic de l'AS Apple

5.5 Conclusion

Ce chapitre 5 est la clé de voûte de cette thèse : les développements présentés dans les chapitres précédents (sélection des informations pertinentes à analyser, mesures des données, estimation du modèle, prédiction et détection d'anomalies) sont utilisés pour proposer une architecture de solution logicielle pour la détection d'anomalies de trafic interdomaine.

Celle-ci se décompose essentiellement en deux parties. Une première dédiée à la récupération de données et à la modélisation du trafic ; la deuxième à la détection d'anomalies. Un des intérêts de cette solution logicielle est de fonder la détection d'anomalies sur un modèle de comportement et non uniquement sur le dépassement d'une valeur seuil de trafic.

Cette solution est désormais en production et utilisée par Post Luxembourg. L'application proposée a été développée dans l'objectif d'être facilement utilisable par l'équipe en charge du réseau mais aussi aisément évolutive suivant les besoins du cœur de réseau.

Chapitre 6

Conclusion

Sommaire

6.1 Conclusion

Lors de ces travaux en collaboration entre le Centre de Recherche en Automatique de Nancy (CRAN) et Post Luxembourg, nous avons développé une chaîne de traitement des données de trafic de bout en bout.

Après une introduction générale sur le domaine des réseaux et le contexte de l'étude, dans le chapitre 2 ont été présentées et comparées différentes méthodes d'observation de trafic réseau. Parce qu'il est le plus adapté à notre étude, nous avons choisi d'utiliser le protocole Netflow qui semble être devenu le standard pour l'analyse de trafic.

Le chapitre 3 est consacré à l'exploration des données de trafic sur plusieurs mois, à la recherche d'informations pouvant nous permettre de caractériser le trafic de certains AS. La double périodicité du trafic a été mise en évidence ainsi que d'autres indicateurs pouvant servir à la classification des AS tel que : l'heure de pic de trafic, le ratio au pic de trafic ou encore la répartition au cours de la journée. Ces caractéristiques montrent que le choix d'avoir agrégé les statistiques par AS source et destination se révèle être un choix judicieux. De plus, cela permet de réduire les quantités de données à traiter tout en conservant une représentation pertinente.

Dans le chapitre 4, ces statistiques et les caractéristiques nous ont permis, après plusieurs essais infructueux, de choisir et tester deux méthodes de modélisation du domaine de l'automatique. La première, de la catégorie des séries temporelles, nous a permis de

mieux appréhender les problématiques de modélisation. La seconde, issue du machine learning par processus gaussiens, paramétrée grâce aux informations recueillies dans le chapitre 3, nous a donné pleinement satisfaction et a été choisie comme base pour la suite de l'étude.

Dans le dernier chapitre, un état des solutions de détection d'anomalies à un niveau opérateur a été dressé. Il nous a paru intéressant d'appliquer ce que nous avons montré dans ces travaux afin d'améliorer les différentes approches actuelles. C'est dans ce contexte que nous avons proposé et développé ANODE, une solution complète de traitement de données, de modélisation, de prédiction et de détection d'anomalies de trafic. Afin d'être le plus simplement mise en œuvre au sein de Post Luxembourg, elle est fondée sur le langage Python et la librairie Scikit-learn.

Cette application est actuellement utilisée en production au sein de Post Luxembourg, en complément des solutions classiques à seuil, afin de mettre en évidence des phénomènes qui jusqu'à présent n'étaient pas mis en lumière par les solutions traditionnelles.

Ces travaux m'ont permis, à partir d'une feuille blanche, de développer une solution de bout en bout, tout en devant prendre à chaque étape des décisions (choix de la méthode de récolte de statistiques, exploration des données, méthode de modélisation...). Ceci a été très enrichissant car j'ai été sans cesse poussé en dehors de ma zone de confort pour acquérir de nouvelles compétences. Cette thèse m'a aussi permis de découvrir le monde de la recherche ainsi que tous les rouages qui le compose, mais aussi de m'immerger dans la vie d'un projet de trois ans en entreprise au sein de Post Luxembourg.

J'ai notamment eu *carte blanche* afin de réaliser mes travaux, c'est pourquoi je tiens encore à remercier l'équipe avec laquelle j'ai partagé ces trois ans.

6.2 Apport de la thèse

Plusieurs points importants ont été levés lors de cette thèse et ont conduit à des propositions innovantes dans le domaine de l'analyse du trafic réseau :

— solution de mesure du trafic réseau à l'intérieur d'un cœur de réseau (chapitre 2 et publication ICNC17) ;
— réduction du problème de données massives (big data) engendrées par la proposition de l'agrégation des mesures par systèmes autonomes (AS) (chapitre 2 et publication ICNC 2017) ;
— proposition de technique de modélisation dédiées à l'analyse du trafic de cœur de réseau fondées sur les méthodes de machine learning (chapitre 3 et 4 et publications ICC 2017 et World IFAC 2017) ;
— proposition et développement d'une solution logicielle ANODE mise en production chez Post Luxembourg et permettant l'analyse de bout en bout du trafic de cœur de réseau : mesure de données, modélisation, prédiction et détection d'anomalies

(chapitre 5, en cours de publication).

6.3 Perspectives

Un travail de thèse n'est pas une fin en soi et les études menées pendant ces trois années peuvent être poursuivies dans plusieurs directions. Parmi celles-ci, on peut en citer plusieurs.

De nombreuses méthodes dédiées aux séries temporelles existent dans la littérature, au milieu de celles-ci se trouve peut être une solution encore plus adaptée à notre système. Il serait aussi possible de continuer à développer des solutions basées sur la méthode GPR utilisée ici. La solution logicielle a été développée de telle sorte à ce que de nouvelles méthodes de modélisation puissent être facilement utilisées.

Dans le domaine du machine learning, les méthodes de régularisation fondées sur les dérivées et favorisant les modèles globalement lisses mais comportant des sauts (jour/nuit et semaine/week-end) peuvent être une piste à explorer. L'apprentissage simultané d'un ensemble de sous-modèles lisses correspondant aux différents modes de fonctionnement combiné à un classifieur sélectionnant le sous-modèle actif à partir de l'entrée est une autre piste possible afin de proposer de nouvelles méthodes de modélisation.

Cette thèse propose une première approche de détection d'anomalies fondées sur le dépassement d'un certain nombre d'occurrences sur une fenêtre glissante. Tout comme pour la méthode de modélisation, l'étude de nouvelles techniques de détection permettrait d'encore améliorer l'application, celle-ci étant ouverte à l'utilisation de nouvelles méthodes de détections.

Lors de ces travaux, afin de détecter des anomalies sur des AS distants, nous nous sommes intéressés à l'étude de trafic. Hors il pourrait être tout aussi pertinent d'explorer également les évènements BGP entre Post Luxembourg et Internet. De nombreux évènements peuvent aussi être découverts par ce biais et par l'association des deux méthodes [64]. Durant la thèse, une solution se basant sur l'étude des routes à été testée grâce à l'option AddPath, permettant de recueillir plusieurs routes au lieu de la meilleure. Il pourrait être intéressant de continuer en ce sens ou d'utiliser le protocole BMP (pour « BGP Monitoring Protocol ») qui devrait être déployée sous peu dans les prochaines mises à jours des équipements réseaux.

Il est aussi possible d'imaginer de continuer le développement de l'application en y intégrant une interface graphique afin de pouvoir directement observer les différents MO. De la même façon, avoir des prédictions à long et moyen terme (utilisant chacune la méthode de modélisation la plus adaptée) permettrait d'aider l'équipe en charge du réseau afin de prévoir les futurs déploiement du réseau.

Liste des publications

Q. Grandemange, O. Ferveur, M. Gilson et E. Gnaedinger. *A live network AS-level traffic characterization*. 2017 IEEE International Conference on Computing, Networking and Communications (ICNC). Silicon Valley, USA, Janvier 2017. Taux d'acceptation : 29 %.

Q. Grandemange, Y. Bhujwalla, M. Gilson, O. Ferveur et E. Gnaedinger. *An as-level approach to network traffic analysis and modelling*. 2017 IEEE International Conference on Communications (ICC). Paris, France, Mai 2017. Taux d'acceptation : 38 %.

Q. Grandemange, Y. Bhujwalla, M. Gilson, O. Ferveur et E. Gnaedinger. *Analysing and modelling a network AS-level traffic*. 20th IFAC World Congress. Toulouse, France, Juillet 2017. Taux d'acceptation : 65 %.

Y. Bhujwalla, Q. Grandemange, M. Gilson, V. Laurain, E. Gnaedinger. *How We Spend Our Time Online : Predicting Network Traffic Using System Identification*. 20th IFAC World Congress. Toulouse, France, Juillet 2017. Taux d'acceptation : 65 %.

Annexe A

Configuration Netflow v9

A.1 Configuration de routeur

A.1.1 Configuration Cisco IOS-XR

Détails de configuration Netflow v9 sur routeur Cisco IOS-XR. Testé sur SR IOS XR v5.1.3.

```
routeur# configure terminal

! configuration de l'exportation des statistiques vers le collecteur
routeur(config)# flow exporter-map fem
routeur(config-fem)# destination [ip du collecteur netflow]
routeur(config-fem)# source [interface de sortie, i.e. "loopback 0"]
routeur(config-fem)# transport udp [port d'écoute du collecteur]
routeur(config-fem)# version v9
routeur(config-fem-ver)# option sampler-table timeout 2000
routeur(config-fem-ver)# template data timeout 10000
routeur(config-fem-ver)# exit

! configuration du sampler de paquets
routeur(config)# sampler-map [nom du sampler-map, i.e. "smap"]
routeur(config-sm)# random 1 out-of [valeur de sampling, i.e. "1000"]
routeur(config-sm)# exit

! configuration du choix des templates de statistiques
routeur(config)# flow monitor-map [nom du monitor-map, i.e. "fmm"]
routeur(config-fmm)# record mpls ipv4-fields
routeur(config-fmm)# exporter fem
```

```
routeur(config-fmm)# exit

! application à une interface
routeur(config)# interface [nom de l'interface]
routeur(config-if)# flow mpls monitor fmm sampler smap ingress|egress
routeur(config-if)exit
```

A.1.2 Configuration Nokia SR-OS

Détails de configuration Netflow v9 sur routeur Nokia, ou Alcatel-Lucent, SR-OS. Testé sur SR-OS v12r9.

```
# configuration de l'exportation des statistiques
routeur# configure cflowd
routeur>config>cflowd# rate [valeur de sampling, i.e. "1000"]
routeur>config>cflowd# template-retransmit 60
routeur>config>cflowd# collector [ip]:[port] version 9
routeur>config>cflowd>collector# template-set mpls-ip
routeur>config>cflowd>collector# exit

# application à une interface
routeur# configure router interface [nom de l'interface]
# ou une interface d'un service
routeur# configure service [type] [id] interface [nom de l'interface]
routeur>interface# cflowd interface ingress|egress|both
```

A.2 Configuration de `nfdump` et `nfsen`

Cette configuration a été testée sur Ubuntu LTS14.04 et avec les versions 1.6.13 de `nfdump` et 1.3.6p1 de `nfsen`.

Dans un premier, nous devons mettre à jour notre système et installer quelques prérequis.

```
apt-get update
apt-get upgrade
apt-get install gcc flex librrd-dev make
apt-get install apache2 libapache2-mod-php5 php5-common
apt-get install libmailtools-perl rrdtool librrds-perl
```

Nous installons maintenant `nfdump`.

```
cd /usr/src/
wget http://downloads.sourceforge.net/project/nfdump/
        stable/nfdump-1.6.13/nfdump-1.6.13.tar.gz
tar zxvf nfdump-1.6.13.tar.gz
cd nfdump-1.6.13
./configure --enable-nfprofile
make
make install
```

Ainsi que `nfsen`.

```
cd /usr/src/
wget http://sourceforge.net/projects/nfsen/files/
        stable/nfsen-1.3.6p1/nfsen-1.3.6p1.tar.gz
tar zxvf nfsen-1.3.6p1.tar.gz
cd nfsen-1.3.6p1
perl -MCPAN -e 'install Socket6'

sudo useradd -M www-data
sudo passwd www-data
sudo usermod -G www-data www-data
sudo mkdir /var/www/html/nfsen
sudo chown -R www-data:www-data /var/www/html/nfsen
sudo mkdir -p /data/nfsen
sudo chown -R www-data:www-data /data/nfsen

cp etc/nfsen-dist.conf /etc/nfsen.conf
```

Ensuite, il faut modifier les lignes du fichier de configuration "/etc/nfsen.conf" suivantes.

```
$HTMLDIR = "/var/www/html/nfsen/";
$USER = "www-data";
$WWWUSER = "www-data";
$WWWGROUP = "www-data";
```

Il est désormais possible d'installer `nfsen`.

```
cd /usr/src/nfsen-1.3.6p1
sudo ./install.pl /etc/nfsen.conf
cd /data/nfsen/bin
./nfsen start
```

Pour lancer `nfsen` comme un service, nous pouvons créer la commande.

```
sudo ln -s /data/nfsen/bin/nfsen /etc/init.d/nfsen
sudo update-rc.d nfsen defaults 20
```

Suite à cela il est possible d'ajouter des sources dans le fichier de configuration situé maintenant à l'adresse "/data/nfsen/etc/nfsen.conf".

```
sources = (
    <nom>' => { 'IP' => '<IP du routeur>', 'port' => '<port d'écoute>',
        'col' => '#0000ff', 'type' => 'netflow', 'optarg' => '-T all' },
);
```

Bibliographie

[1] Cisco, "The zettabyte era : Trends and analysis," White Paper, 2015.

[2] L. Kleinrock, *Queueing systems, volume 2 : Computer applications*. wiley New York, 1976, vol. 66.

[3] D. Yang and Z. Rong, "Evolution of the internet at the autonomous system level," in *34th Chinese Control Conference (CCC)*, Hangzhou, China, July 28-30 2015, pp. 1313–1317.

[4] P. Faratin, D. D. Clarck, S. Bauer, W. Lehr, P. W. Gilmore, and A. Berger, "The growing complexity of internet interconnection," *Communications and Strategies*, no. 72, p. 51, 2008.

[5] B. Ager, B. Chatzis, A. Feldmann, N. Sarrar, S. Uhlig, and W. Willinger, "Anatomy of a large european ixp," in *SIGCOMM'12*, Helsinki, Finland, August 13-17 2012, pp. 163–174.

[6] R. Caceres, N. Duffield, A. Feldmann, J. D. Friedmann, A. Greenberg, R. Greer, T. Johnson, C. R. Kalmanek, B. Krishnamurthy, D. Lavelle *et al.*, "Measurement and analysis of ip network usage and behavior," *IEEE Communications Magazine*, vol. 38, no. 5, pp. 144–151, 2000.

[7] C. Fraleigh, S. Moon, B. Lyles, C. Cotton, M. Khan, D. Moll, R. Rockell, T. Seely, and S. C. Diot, "Packet-level traffic measurements from the sprint ip backbone," *IEEE network*, vol. 17, no. 6, pp. 6–16, 2003.

[8] W. John, S. Tafvelin, and T. Olovsson, "Passive internet measurement : Overview and guidelines based on experiences," *Computer Communications*, vol. 33, no. 5, pp. 533 – 550, 2010.

[9] P. Owezarski and N. Larrieu, "Techniques et outils de métrologie pour l'internet et son trafic," *Techniques de l'ingénieur*, vol. 1090, pp. 1090–22, 2006.

[10] H. Wang, L. Song, H. Chen, and L. Yan, "Study on network measurement technologies and performance evaluation methods," in *2013 Cross Strait Quad-Regional Radio Science and Wireless Technology Conference*, July 2013, pp. 377–380.

[11] MEF, "Service oam fault management implementation agreement," Metro Ethernet Forum, Recommendation, January 2011. [Online]. Available : https://mef.net/PDF_Documents/technical-specifications/MEF_30.pdf

[12] ——, "Service oam performance monitoring implementation agreement," Metro Ethernet Forum, Recommendation, April 2012. [Online]. Available : https://mef.net/PDF_Documents/technical-specifications/MEF_35.pdf

[13] IEEE, "Connectivity fault management," IEEE Standards Association, STD, December 2007.

[14] ITU-T, "Recommendation itu-t g.8013/y.1731 operation, administration and maintenance (oam) functions and mechanisms for ethernet-based networks," ITU-T, Recommendation, August 2015. [Online]. Available : https://www.itu.int/rec/T-REC-G.8013-201508-I/en

[15] J. Postel, "Internet control message protocol," Internet Requests for Comments, RFC Editor, STD 5, September 1981. [Online]. Available : http://www.rfc-editor.org/rfc/rfc792.txt

[16] E. Rosen, A. Viswanathan, and R. Callon, "Multiprotocol label switching architecture," Internet Requests for Comments, RFC Editor, RFC, January 2001. [Online]. Available : http://www.rfc-editor.org/rfc/rfc3031.txt

[17] R. Hofstede, I. Drago, G. Moura, and A. Pras, "Carrier ethernet oam : an overview and comparison to ip oam," *Managing the Dynamics of Networks and Services*, pp. 112–123, 2011.

[18] J.-C. Bolot, "End-to-end packet delay and loss behavior in the internet," in *ACM SIGCOMM Computer Communication Review*, vol. 23, no. 4. ACM, 1993, pp. 289–298.

[19] R. Prasad, C. Dovrolis, M. Murray, and K. Claffy, "Bandwidth estimation : metrics, measurement techniques, and tools," *IEEE network*, vol. 17, no. 6, pp. 27–35, 2003.

[20] M. Zink, K. Suh, Y. Gu, and J. Kurose, "Characteristics of youtube network traffic at a campus network–measurements, models, and implications," *Computer networks*, vol. 53, no. 4, pp. 501–514, 2009.

[21] C. Barakat, P. Thiran, G. Iannaccone, C. Diot, and P. Owezarski, "A flow-based model for internet backbone traffic," in *Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurment*. ACM, 2002, pp. 35–47.

[22] J. D. Case, M. Fedor, M. L. Schoffstall, and J. R. Davin, "Simple network management protocol (snmp)," Internet Requests for Comments, RFC Editor, STD 15, May 1990. [Online]. Available : http://www.rfc-editor.org/rfc/rfc1157.txt

[23] S. Waldbusser, "Remote network monitoring management information base version 2," Internet Requests for Comments, RFC Editor, RFC 4502, May 2006.

[24] B. Claise, "Cisco systems netflow services export version 9," Internet Requests for Comments, RFC Editor, RFC 3954, October 2004, http://www.rfc-editor.org/rfc/rfc3954.txt. [Online]. Available : http://www.rfc-editor.org/rfc/rfc3954.txt

[25] R. Hofstede, P. Celeda, B. Trammell, I. Drago, R. Sadre, A. Sperotto, and A. Pras, "Flow monitoring explained : From packet capture to data analysis with netflow and ipfix," *IEEE Communications Surveys Tutorials*, vol. 16, no. 4, pp. 2037–2064, 2014.

[26] A. Feldmann, A. Greenberg, C. Lund, N. Reingold, J. Rexford, and F. True, "Deriving traffic demands for operational ip networks : Methodology and experience," *IEEE/ACM Trans. Netw.*, vol. 9, no. 3, pp. 265–280, Jun. 2001.

[27] C. Labovitz, S. Iekel-Johnson, D. McPherson, J. Oberheide, and F. Jahanian, "Internet inter-domain traffic," in *SIGCOMM'10*, New Delhi, India, August 30 – September 3 2010, pp. 75–86.

[28] M. Roughan, W. Willinger, O. Maennel, D. Perouli, and R. Bush, "10 lessons from 10 years of measuring and modeling the internet's autonomous systems," *IEEE Journal on Selected Areas in Communications*, vol. 29, no. 9, pp. 1810–1821, 10 2011.

[29] C. Labovitz, A. Ahuja, and F. Jahanian, "Experimental study of internet stability and backbone failures," in *Digest of Papers. Twenty-Ninth Annual International Symposium on Fault-Tolerant Computing (Cat. No.99CB36352)*, June 1999, pp. 278–285.

[30] W. Fang and L. Peterson, "Inter-as traffic patterns and their implications," in *Global Telecommunications Conference, 1999. GLOBECOM '99*, vol. 3, 12 1999, pp. 1859–1868.

[31] K. Cho, K. Fukuda, H. Esaki, and A. Kato, "The impact and implications of the growth in residential user-to-user traffic," in *Proceedings of the 2006 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, ser. SIGCOMM '06. New York, NY, USA : ACM, 2006, pp. 207–218.

[32] M. Kihl, P. Ödling, C. Lagerstedt, and A. Aurelius, "Traffic analysis and characterization of internet user behavior," in *International Congress on Ultra Modern Telecommunications and Control Systems*, Oct 2010, pp. 224–231.

[33] M. Feknous, T. Houdoin, B. L. Guyader, J. D. Biasio, A. Gravey, and J. A. T. Gijón, "Internet traffic analysis : A case study from two major european operators," in *2014 IEEE Symposium on Computers and Communications (ISCC)*, June 2014, pp. 1–7.

[34] M.-S. Kim, Y. J. Won, and J. W. Hong, "Characteristic analysis of internet traffic from the perspective of flows," *Computer Communications*, vol. 29, no. 10, pp. 1639 – 1652, 2006, monitoring and Measurements of IP Networks.

[35] S. Alvisi, M. Franchini, and A. Marinelli, "A short-term, pattern-based model for water-demand forecasting," *Journal of Hydroinformatics*, vol. 9, no. 1, pp. 39–50, 2007. [Online]. Available : http://jh.iwaponline.com/content/9/1/39

[36] H. Jiang, Z. Ge, S. Jin, and J. Wang, "Network prefix-level traffic profiling : Characterizing, modeling, and evaluation," *Computer Networks*, vol. 54, no. 18, pp. 3327 – 3340, 12 2010.

[37] A. Dhamdhere and C. Dovrolis, "Twelve years in the evolution of the internet ecosystem," *IEEE/ACM Transactions on Networking*, vol. 19, no. 5, pp. 1420–1433, 10 2011.

[38] P. Cortez, M. Rio, M. Rocha, and P. Sousa, "Multi-scale internet traffic forecasting using neural networks and time series methods," *Expert Systems*, vol. 2, pp. 143–155, 2012.

[39] V. C. Moussas, M. Daglis, and E. Kolega, "Network traffic modeling and prediction using multiplicative seasonal arima models," in *Proceedings of the 1st International Conference on Experiments/Process/System Modeling/Simulation/Optimization, Athens*, 2005, pp. 6–9.

[40] J. W. Taylor, "Short-term electricity demand forecasting using double seasonal exponential smoothing," *Journal of the Operational Research Society*, vol. 54, no. 8, pp. 799–805, 2003.

[41] K. Papagiannaki, N. Taft, Z.-L. Zhang, and C. Diot, "Long-term forecasting of internet backbone traffic," *IEEE transactions on neural networks*, vol. 16, no. 5, pp. 1110–1124, 2005.

[42] T. Söderström and P. Stoica, *System identification*. Prentice-Hall, Inc., 1989.

[43] L. Ljung, *System Identification : Theory For The User*, 2nd ed. Prentice Hall, Upper Saddle River, 1999.

[44] R. Tóth, *Modeling and identification of linear parameter-varying systems*. Springer, 2010, vol. 403.

[45] P. C. Young, *Recursive Estimation and Time-Series Analysis : An Introduction for the Student and Practitioner*. Springer-Verlag Berlin Heidelberg, 2011.

[46] P. Young, D. Pedregal, and W. Tych, "Dynamic harmonic regression," *Journal of Forecasting*, vol. 18, pp. 369–394, 1999.

[47] C. E. Rasmussen, *Gaussian processes for machine learning*. MIT Press, 2006.

[48] P. Gonzàlez and P. Moral, "Comments on 'an analysis of the international tourism demands in spain'," *International Journal of Forecasting*, vol. 13, pp. 551–556, 1997.

[49] W. Tych, D. Pedregal, P. Young, and J. Davies, "An unobserved component model for multi-rate forecasting of telephone call demand : the design of a forecasting support system," *International Journal of Forecasting*, vol. 18, pp. 673–695, 2002.

[50] P. Young and C. J. Taylor, "Recent developments in the captain toolbox for matlab," in *16th IFAC Symposium on System Identification*, Bruxelles, Belgium, 2012.

[51] P. Stoica, P. Eykhoff, P. Janssen, and T. Söderström, "Model-structure selection by cross-validation," *International Journal of Control*, vol. 43, no. 6, pp. 1841–1878, 1986.

[52] N. K. Ahmed, A. F. Atiya, N. E. Gayar, and H. El-Shishiny, "An empirical comparison of machine learning models for time series forecasting," *Econometric Reviews*, vol. 29, no. 5-6, pp. 594–621, 2010.

[53] C. M. Bishop, *Pattern recognition and machine learning*. springer, 2006.

[54] C. K. Williams and C. E. Rasmussen, "Gaussian processes for regression," *Advances in neural information processing systems*, pp. 514–520, 1996.

[55] C. E. Rasmussen and H. Nickisch, "Gaussian processes for machine learning (gpml) toolbox," *Journal of Machine Learning Research*, vol. 11, no. Nov, pp. 3011–3015, 2010.

[56] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn : Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[57] E. Jones, T. Oliphant, P. Peterson *et al.*, "SciPy : Open source scientific tools for Python," 2001–, [Online ; accessed <today>]. [Online]. Available : http://www.scipy.org/

[58] G. Cormode and S. Muthukrishnan, "What's new : Finding significant differences in network data streams," *IEEE/ACM Transactions on Networking (TON)*, vol. 13, no. 6, pp. 1219–1232, 2005.

[59] B. Krishnamurthy, S. Sen, Y. Zhang, and Y. Chen, "Sketch-based change detection : methods, evaluation, and applications," in *Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement*. ACM, 2003, pp. 234–247.

[60] O. Salem, S. Vaton, and A. Gravey, "An efficient online anomalies detection mechanism for high-speed networks," in *MonAM 2007 : Second IEEE Workshop on Monitoring, Attack Detection and Mitigation*, 2007, pp. 1–6.

[61] A. Lakhina, M. Crovella, and C. Diot, "Mining anomalies using traffic feature distributions," in *ACM SIGCOMM Computer Communication Review*, vol. 35, no. 4. ACM, 2005, pp. 217–228.

[62] J. Mazel, "Unsupervised network anomaly detection," Ph.D. dissertation, INSA de Toulouse, 2011.

[63] J. D. Brutlag, "Aberrant behavior detection in time series for network monitoring." in *LISA*, vol. 14, no. 2000, 2000, pp. 139–146.

[64] M. Roughan, T. Griffin, M. Mao, A. Greenberg, and B. Freeman, "Combining routing and traffic data for detection of ip forwarding anomalies," *ACM SIGMETRICS Performance Evaluation Review*, vol. 32, no. 1, pp. 416–417, 2004.

Résumé

Grâce au partenariat avec l'entreprise luxembourgeoise Post Luxembourg, nous avons pu tester différentes méthodes pour mesurer le trafic interdomaine à la bordure de leur réseau avec Internet. Le choix s'est porté sur une technologie existante : Netflow. Avec ces données nous avons pu réaliser diverses analyses afin de comprendre l'évolution du trafic en fonction de différents paramètres comme l'heure de la journée, le jour de la semaine...

D'après ces analyses, plusieurs solutions ont été envisagées pour modéliser le trafic. Deux méthodes ont été proposées et testées sur des données réelles : une méthode d'analyse de séries temporelles et une méthode de machine learning reposant sur les processus gaussiens. Ces techniques ont été comparées sur différents systèmes autonomes. Les résultats sont satisfaisants pour les deux méthodes avec un avantage pour la méthode des processus gaussiens.

Cette thèse propose le développement d'une solution logicielle ANODE mise en production chez Post Luxembourg et permettant l'analyse de bout en bout du trafic de cœur de réseau : mesure de données, modélisation, prédiction et détection d'anomalies.

Mots-clés: Système autonome, Modélisation, Machine learning, Routage interdomaine, Détection d'anomalies

Abstract

Inter-domain routing statistics are not usually publicly available but with the partnership with Post Luxembourg, we deployed a network wide measurements of Internet traffic. Those statistics show clear daily and weekly pattern and several points of interest.

From all the information gathered, two modelling approach were chosen : the first one from the time series domain and the second one from the machine learning approach. Both were tested on several dataset of autonomous systems and the second one, Gaussian Process, was kept for the next steps.

The proposal of this study is the development of a software solution called ANODE, which is used at Post Luxembourg, allowing the analysis of backbone traffic : measurments, modelling, forecasting and anomaly detection.

Keywords: Autonomous System, Modelling, Machine learning, Inter-domain routing, Anomaly detection