Faculté
des Sciences
Département de l'Informatique

UMONS
Université de Mons

Titre

Mémoire réalisé par Prénom NOM
pour l'obtention du diplôme de Master en Sciences Informatiques

Année académique 2009–2010

Directeur : Nom du directeur

Service : Service dans lequel vous avez fait votre mémoire

Faculté des Sciences • Université de Mons • Place du Parc 20 • B-7000 Mons

Table des matières

Introduction générale

Actuellement, plus de $10,300$ [1] sondes Atlas sont déployées dans le monde pour effectuer des mesures réseaux comme le DNS, Ping, Traceroute, etc. Ces sondes sont maintenues par le RIPE NCC (Réseaux IP Européens - Network Coordination Centre). Les données collectées par ces mesures sont stockées et sont disponibles en accès libre [2]. Quotidiennement, plus de 18732 mesures sont planifiées au départ de ces sondes vers de multiples autres destinations. En moyenne, 33 Go de données [3] sont collectées chaque jour pour tous les types de mesures.

Le besoin en stockage des données est en croissance continue avec la quantité de données générées par les transactions des clients, les réseaux sociaux, l'Internet des objets qui collectent constamment les données, etc. Les solutions traditionnelles en terme de stockage, de calcul et de visualisation ne répondent pas aux besoins, surtout des grandes organisations. Ce qui les a encouragé à créer des solutions permettant de répondre aux nouveaux besoins, c'est le Big Data.

L'objectif du présent mémoire est de montrer la capacité des nouvelles technologies du Big Data à fournir des solutions efficaces capables d'assurer le stockage des données massives et d'effectuer des tâches de traitement sur ces quantités de données. Dans notre cas, ce sont des données collectées par les sondes Atlas. Ces données apportant des informations utiles et pertinentes que nous recueillons sur l'état du réseau.

Les articles, les travaux publiés par RIPE Atlas et les présentations durant les rencontres organisées par RIPE NCC permettent d'avoir une idée générale sur les sujets à traiter en vue d'exploiter les données collectées par les sondes Atlas. Plus généralement, les sujets abordés sont : la visualisation de certains indicateurs sur les performances d'un réseau, l'analyse des censures appliquées au niveau de certains pays, l'analyse des détours que subit un trafic local et l'étude des performances d'un réseau, par exemple : le temps de la latence, la perte des paquets,

1. A la date de l'accès à la source `https://atlas.ripe.net/results/maps/network-coverage/`, le $14/08/2018$.

2. Les données des derniers 30 jours, les données des autres périodes sont accessibles avec une API REST.

3. Au format compressé.

l'asymétrie du trafic et la congestion des routeurs.

Les utilitaires ping et traceroute font partie des outils d'analyse de l'état du réseau et de résolution des problèmes dans les réseaux fortement utilisés. En particulier, l'utilitaire traceroute fournit des informations de l'aller et du retour entre une adresse IP source et une adresse IP destination sur un réseau. Il fournit les sauts impliqués tout au long du chemin entre la source et la destination ainsi que le temps requis pour les atteindre. Les détails fournis par traceroute permettent d'avoir des informations sur les réseaux traversés, la latence, etc.

Les traceroutes effectués par toutes les sondes, durant une heure, génèrent des données dont la taille est d'environ 8 Go. La manipulation de cette quantité de données nécessite des ressources de hautes performances. On ne peut pas compter sur les ressources traditionnelles comme les bases de données relationnelles pour le stockage, les processeurs des machines ordinaires [4] pour traiter les données après la récupération de celles-ci.

L'analyse des données collectées par les sondes Atlas a prouvé l'utilité de ces données. Plusieurs cas d'utilisation sont régulièrement publiés. Nous nous intéresserons au sujet du délai d'un lien réseau (lien topologique), car traceroute fournit les sauts impliqués dans un chemin, entre une adresse IP source et une adresse IP destination, avec les informations de la latence. Nous allons étudier la capacité des technologies Big Data à gérer la quantité de données générées par les sondes Atlas. La gestion des données porte sur le stockage, le traitement et la visualisation.

Ce document est structuré comme suit : le premier chapitre reprend une présentation du projet RIPE Atlas où nous allons présenter les sondes Atlas, leur fonctionnement et quelques cas d'utilisation spécifiques. Le deuxième chapitre introduit le terme Big Data, puis, il reprend les disciplines impliquées dans le Big Data, pour ensuite parcourir un ensemble d'outils Big Data. Pour conclure le deuxième chapitre, un choix sera énoncé concernant les outils à utiliser dans l'analyse des traceroutes. En ce qui concerne le troisième chapitre, il reprendra les étapes de l'analyse de données des traceroutes suivant un processus particulier. Depuis la collecte des données Atlas jusqu'à la génération des résultats de l'analyse.

4. Machines avec une mémoire RAM de 4 ou de 8 Go.

Chapitre I

RIPE Atlas

I.1 Introduction

Le présent chapitre commence par une présentation détaillée du projet RIPE Atlas mené par l'organisme RIPE NCC. Ce projet a introduit l'utilisation des sondes pour effectuer des mesures des réseaux dans le monde. Ensuite, ce chapitre reprend une liste non exhaustive de quelques outils similaires aux sondes Atlas en matière d'objectifs. Enfin, expose quelques limites du système RIPE Atlas. La dernière section reprend brièvement quelques travaux basés sur le projet RIPE Atlas.

I.2 A propos RIPE NCC

Le RIPE NCC est un organisme qui alloue les blocs d'adresses IP et des numéros des Systèmes Autonomes dans l'Europe et une partie de l'Asie, notamment au Moyen-Orient.

> Un ***Système Autonome***, appelé AS, est un ensemble de réseaux et de routeurs sous la responsabilité d'une même autorité administrative. Chaque Système Autonome est identifié par un code sur 16 bits uniques. Les protocoles qui tournent au sein d'un Système Autonome peuvent être différents.

RIPE NCC assure différents services relatifs à la gestion des réseaux informatiques. Il maintient multiples projets pour un nombre de protocoles comme DNS (DNSMON), BGP (Routing Information Service ou RIS) et d'autres projets et services. En particulier, nous sommes intéressés par projet RIPE Atlas géré aussi par RIPE NCC. L'objectif du projet RIPE Atlas est de déployer des dispositifs dans le monde, capables de collecter des données réseaux. Nous allons le

détailler dans la section I.3.

I.3 Présentation du projet RIPE Atlas

RIPE NCC a créé le projet RIPE Atlas en 2010. Le nombre de sondes déployées est en augmentation constante, sachant qu'elles sont déployées par des volontaires.

I.3.1 Les mesures actives et passives de l'Internet

Il existe plusieurs approches pour analyser l'état d'un réseau. Les deux approches les plus répandues sont : active et passive. L'approche passive fait référence au processus de mesure d'un réseau, sans créer ou modifier le trafic sur ce réseau. L'approche active repose sur l'injection des paquets sur le réseau et surveiller le flux de ce trafic. Cette injection a pour objectif la collecte des données relatives aux performances du réseau en question. Par exemple, la mesure du temps de réponse, le suivi du chemin des paquets, etc.

Les données collectées permettent de surveiller les réseaux pour ensuite proposer des améliorations de l'Internet. Le projet RIPE Atlas est un des outils s'inscrivant dans l'approche active. Ce sont des dispositifs, appelés sondes, hébergés par des volontaires, ils sont distribués et maintenus par RIPE NCC. Les données collectées par ces dispositifs sont disponibles au public [4].

Actuellement, plus de $10,000$ sondes Atlas sont actives, ces dernières produisent environ 450 millions de mesures par jour, ce qui correspond à $5,000$ résultats par seconde [22].

I.3.2 Généralités sur les sondes Atlas

- Les sondes Atlas mesurent les performances de la couche IP. Une sonde envoie des paquets réels et observe la réponse en temps réel indépendamment des applications en dessus de la couche IP.

- Les sondes Atlas ne sont pas des observatrices des données comme le trafic du routage BGP, ainsi, elles n'observent pas le trafic de leurs hébergeurs.

- Les sondes Atlas se situent dans différents emplacements dans le monde, cette répartition permet de diversifier les mesures (voir les sections des mesures I.3.9 et I.3.11).

- Les sondes Atlas sont déployées volontairement dans une maison, un bureau, un entrepôt de données, etc.

– Les mesures peuvent être lancées à tout moment et pour n'importe quelle période [1].

– La participation au projet RIPE Atlas est ouverte à toute personne qui s'y intéresse, cela inclut les résultats de mesures, les outils d'analyse, l'hébergement des sondes elles-mêmes, les travaux, etc.

– RIPE Atlas simule le comportement de la couche IP. Par exemple, avec RIPE Atlas, il est possible de :

  – Suivre l'accessibilité d'une destination [2] depuis différents emplacements dans le monde et depuis différents réseaux. Car les sondes Atlas sont réparties dans plusieurs pays et déployées dans différents réseaux.

  – Étudier des problèmes du réseau remontés en effectuant des vérifications de connectivité ad-hoc via les mesures effectuées par les sondes Atlas.

  – Tester la connectivité IPv6.

  – Vérifier l'infrastructure DNS.

La section I.5 reprend quelques cas d'utilisation du système RIPE Atlas et les sujets qu'on peut étudier.

I.3.3 Les générations des sondes Atlas

Depuis leur création en 2010, les sondes Atlas ont connu trois générations du matériel. Le tableau I.1 reprend quelques caractérisations de ces trois générations des sondes Atlas et la figure I.4 montre le matériel utilisé dans chaque génération.

1. Si le nombre de crédits (voir la section des crédits I.3.10) disponibles le permet et qu'il n'y a pas de dépassement du nombre de mesures autorisé.
2. Une destination représente une adresse IP dans le présent contexte.

	v1	**v**2	**v**3
Matériel informatique	Lantronix XPort Pro [8]	Lantronix XPort Pro [8]	tp-link tl-mr3020
Début d'utilisation	2010	2011	2013
Mémoire RAM	8 Mo	16 Mo	32 Mo
Mémoire Flash	16 Mo	16 Mo	4 Mo
CPU	32-bit	32-bit	32-bit
Support du Wi-Fi	Non	Non	oui
Support du NAT	oui	oui	oui
Vitesses supportées	10 Mbit/s et 100 Mbit/s	10 Mbit/s et 100 Mbit/s	10 Mbit/s et 100 Mbit/s

TABLE I.1 – Les caractéristiques des trois générations des sondes Atlas

FIGURE I.1 – Génération 1

FIGURE I.2 – Génération 2

FIGURE I.3 – Génération 3

FIGURE I.4 – Les trois générations des sondes Atlas

Source : `https://atlas.ripe.net/docs/`, consultée le 05/08/2018.

Pour précision, les générations 1 et 2 présentent une très faible consommation d'énergie, cependant, elles ont un temps de redémarrage et coûts de production élevés.

En 2015, plusieurs utilisateurs des sondes Atlas ont montré un intérêt aux sondes virtuelles. Ces sondes virtuelles présentent des avantages et aussi des inconvénients. Parmi les avantages, la conception des sondes virtuelles permet d'explorer des emplacements qui sont difficilement accessibles. En effet, cela permet d'étendre le réseau des sondes Atlas. D'autre part, les sondes virtuelles peuvent être installées sans contraintes physiques ou organisationnelles. Parmi les inconvénients, une complexité sera ajoutée au système RIPE Atlas, plus de ressources seront demandées. Ensuite, il y a le problème de la qualité des données ; le manque

de données peut faire référence à une perte de paquets ou bien la machine qui héberge la sonde n'est plus disponible pour continuer les mesures.

I.3.4 La connexion des sondes Atlas à Internet

Les génération 1 et 2 des sondes Atlas ont une interface Ethernet (RJ-45). La génération 3 dispose techniquement des capacités Wi-Fi. Cependant, ces sondes ne sont pas suffisamment prêtes au niveau logiciel pour supporter le Wi-Fi. L'objectif était de garder l'indépendance des sondes Atlas du trafic de celui qui les héberge.

Une fois la sonde se connecte au port d'Ethernet, elle acquiert une adresse IPv4, un résolveur DNS en utilisant DHCP et la configuration IPv6 via *Router Advertisement*. Ensuite, elle essaie de rejoindre l'infrastructure du RIPE Atlas. Pour ce faire, elle utilise le résolveur DNS et se connecte à l'infrastructure à travers SSH sur le port TCP de sortie 443 comme il est illustré dans la figure I.5. L'architecture du système RIPE Atlas est détaillée dans la section I.3.5.

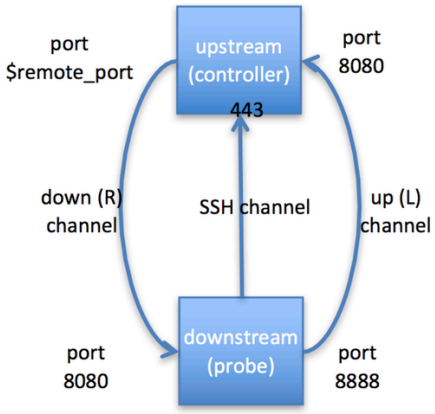


FIGURE I.5 – La connexion des sondes Atlas à l'infrastructure RIPE Atlas [21]

I.3.5 Architecture du système RIPE Atlas

Il existe deux catégories d'outils de surveillance du réseau : des outils matériels et d'autres logiciels. Les sondes Atlas sont parmi les outils matériels. le choix d'utilisation d'un outil matériel au lieu d'un outil logiciel dépend de plusieurs facteurs, par exemple l'indépendance du système d'exploitation, la facilité de déploiement, la disponibilité des sondes tout le temps (au lieu d'être dépendante de la machine qui l'héberge) et d'autres facteurs liés à la sécurité.

Le système RIPE Atlas est conçu pour qu'il soit opérationnel de façon distribuée. La plupart des composantes ont assez de connaissances pour remplir leurs rôles, sans nécessairement avoir besoin de connaître les états des autres composantes du système. Cela assure que le système soit capable d'assurer la plupart des fonctionnalités en cas d'un problème temporaire. Par exemple, si une sonde est déconnectée de l'infrastructure, elle continue les mesures planifiées et les données sont renvoyées dès sa reconnexion au système.

La figure I.6 montre une vue d'ensemble de l'architecture du RIPE Atlas.

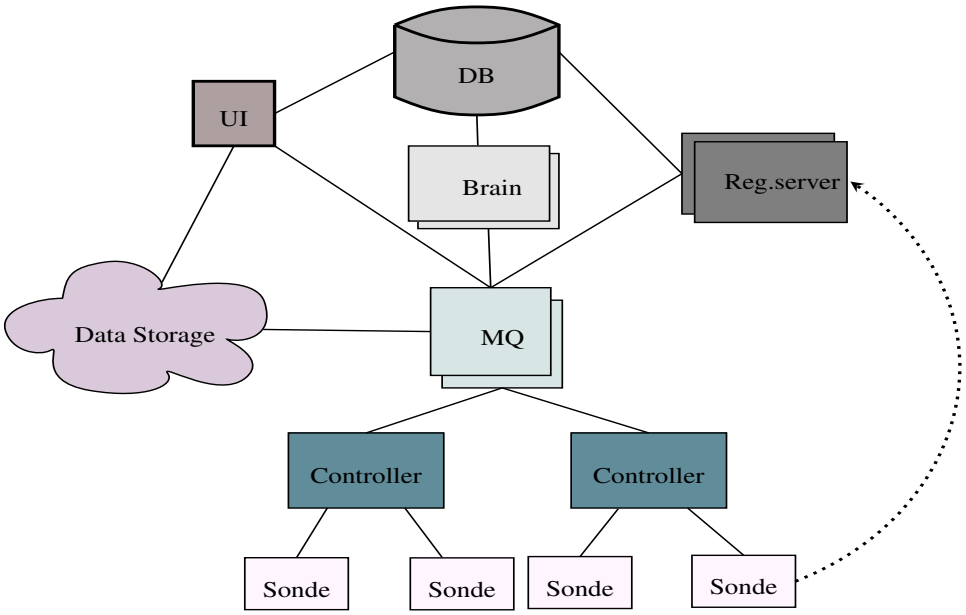


FIGURE I.6 – Architecture du système RIPE Atlas [22]

On distingue les composantes suivantes :

Registration server (Reg.server) : c'est le seul point d'entrée de confiance pour les sondes Atlas. Son rôle est de recevoir toutes les sondes désirant se connecter au système RIPE Atlas. Ensuite, il redirige chaque sonde vers le contrôleur adéquat, celui le plus proche de la sonde et que ce contrôleur soit suffisamment non occupé. Le serveur d'enregistrement a un aperçu de haut niveau du système.

Controller : les contrôleurs acceptent d'établir une connexion avec une sonde parmi les sondes dont ils ont reçu leurs clés du serveur d'enregistrement (Reg.server). Une fois la connexion est établie entre une sonde et un contrôleur, ce dernier garde cette connexion active pour recevoir les résultats et

prévenir la sonde des mesures à effectuer. Le rôle du contrôleur est de communiquer avec les sondes, associer les mesures aux sondes en se basant sur la disponibilité de la sonde et autres critères, enfin, collecter les résultats intermédiaires des mesures.

Message Queue (MQ) : Tout d'abord définissons MQ :

> « ***Message Queue ou file d'attente de message*** *: est une technique de programmation utilisée pour la communication interprocessus ou la communication de serveur-à-serveur. Les files d'attente de message permettent le fonctionnement des liaisons asynchrones normalisées entre deux serveurs, c'est-à-dire de canaux de communications tels que l'expéditeur et le récepteur du message ne sont pas contraints de s'attendre l'un l'autre, mais poursuivent chacun l'exécution de leurs tâches*[a]*.* »
>
> *a.* Source : `https://fr.wikipedia.org/wiki/File_d'attente_de_message`, consultée le 05/08/2018.

Un cluster de serveurs MQ agit comme un système nerveux central au sein de l'architecture du RIPE Atlas. Il gère la connectivité entre les composantes de l'infrastructure et assure l'échange de messages avec un délai minimal. C'est cette composante qui élimine le besoin que les autres composantes de l'infrastructure soient au courant des états des autres composantes de l'infrastructure. En plus, chaque composante peut être ajoutée ou retirée sans devoir synchroniser cette information à l'infrastructure entière. Si c'est le cas d'une déconnexion d'une composante, les messages seront sauvegardés sur différents niveaux jusqu'au moment de la reconnexion.

UI (User Interface) : elle s'occupe des interactions de l'utilisateur. Elle sert les pages pour l'interface graphique de mesures [3]. Elle traite les appels en provenance de l'API [3] et sert les demandes de téléchargement en provenance de l'API.

Brain : il effectue des tâches de haut niveau dans le système, notamment la planification des mesures. Cette planification est basée sur les demandes reçues via l'interface graphique web de mesures (UI) ou bien via l'API. La planification passe par la présélection des sondes Atlas et la négociation avec les contrôleurs pour voir la disponibilité des sondes Atlas.

3. Source : `https://atlas.ripe.net/docs/api/v2/manual/`, consultée le 05/08/2018.

DB : c'est une base de données SQL contenant toutes les informations du système RIPE Atlas : les informations sur les sondes et leurs propriétés, les meta-data des mesures, les utilisateurs, les crédits, etc.

Data Storage : c'est un cluster Hadoop/HBase pour le stockage à long terme de tous les résultats . Cette technologie permet aussi d'effectuer des calculs d'agrégation périodiques et d'autres tâches.

> ***Hadoop MapReduce*** est un modèle de programmation qui permet de traiter les données massives suivant une architecture distribuée dans un cluster.
> ***HBase*** est une base de données non relationnelle et distribuée. Elle est adaptée au stockage de données massives.

La figure I.7 présente les étapes d'établissement de la connexion entre une sonde Atlas et l'infrastructure RIPE Atlas.

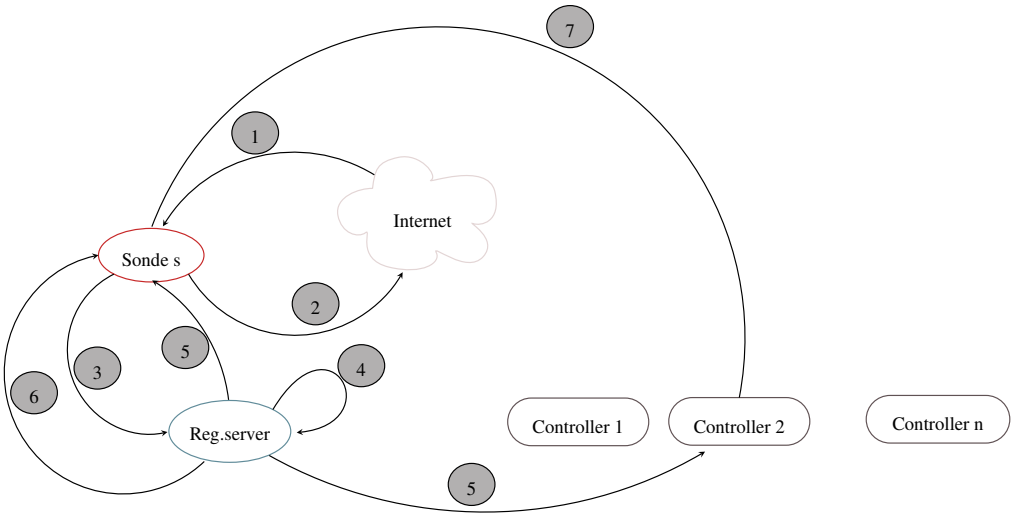


FIGURE I.7 – Les étapes d'établissent d'une connexion entre la sonde Atlas et l'architecture RIPE Atlas

Les étapes suivantes illustrent le déroulement de la connexion d'une sonde Atlas s à l'infrastructure RIPE Atlas.

- La sonde Atlas se connecte à Internet via le câble Ethernet $RJ45$ (1).
- La sonde Atlas acquiert différentes informations : une adresse IPv4, une adresse IPv6 via Router Advertisement et les informations du résolveur DNS via DHCP (2).

- Les informations précédemment acquises permettent à la sonde Atlas de se connecter au serveur d'enregistrement (Reg.server). C'est la première entrée vers l'infrastructure (3).

- En se basant sur la géolocalisation de la sonde Atlas, la charge des différents contrôleurs et d'autres options, le serveur d'enregistrement décide le contrôleur qui va être associé à la sonde Atlas (4).

- Suite à la décision du serveur d'enregistrement, le contrôleur reçoit l'identifiant de la sonde Atlas à gérer et la sonde Atlas reçoit l'identifiant du contrôleur avec à qui elle sera associée (5).

- Une fois l'association entre la sonde Atlas et le contrôleur est faite, la sonde Atlas se déconnecte du serveur d'enregistrement (6).

- La connexion entre la sonde Atlas et le contrôleur est maintenue le plus longtemps possible. Les contrôleurs gardent le contact avec les autres composantes via Message Queue. Dans le cas où une des composantes se déconnecte de l'architecture, les événements sont conservés jusqu'au moment où la connexion est restaurée (7).

La connexion précédemment établie permet aux sondes Atlas d'envoyer leurs rapports de mesures aux serveurs de stockage. C'est la même connexion qui permet de passer les commandes aux sondes pour qu'elles puissent effectuer les mesures et les mises à jour de leur firmware.

I.3.6 Les sondes Atlas et la vie privée

La sonde Atlas n'a pas l'accès au trafic de son hébergeur. Elle maintient sa connexion avec l'infrastructure centrale et elle exécute les mesures planifiées vers les destinations publiques sur Internet.

Les sondes Atlas peuvent révéler l'adresse IP de leur hébergeur. Bien que, les informations personnelles telles que les adresses MAC et les adresses e-mail ne seront jamais affichées. Cependant, l'adresse IPv6 peut exposer l'adresse MAC.

I.3.7 La sécurité dans RIPE Atlas

La connexion entre les composantes de l'infrastructure RIPE Atlas est maintenue le plus longtemps possible comme c'est décrit dans la section I.3.5. De ce fait, la sécurité des différentes connexions est primordiale. Afin de réduire la surface d'attaque contre ces sondes, les précautions suivantes sont prises :

- Les hébergeurs des sondes Atlas ne disposent d'aucun service qui leur permet de se connecter aux sondes (dans le sens de TCP/IP).
- Les sondes Atlas n'échangent aucune clé d'authentification entre elles. En effet, chaque sonde dispose de sa clé qu'elle l'utilise pour se connecter à l'infrastructure.
- Comme les sondes Atlas sont chez les hébergeurs, il est impossible qu'elles soient résilientes au démontage. Cependant, si c'était le cas, cela ne devrait pas affecter les autres sondes Atlas.
- Toutes les communications au sein de l'infrastructure RIPE Atlas se font d'une manière sécurisée. Les connexions entre les composantes sont maintenues grâce aux *secure channels* avec *mutual authentication*.
- Le logiciel qui tourne dans les sondes Atlas peut être facilement mis à niveau ; la sonde Atlas est capable de vérifier l'authenticité d'une nouvelle version du firmware et cela via les signatures cryptographiques.

Le système RIPE Atlas est un système comme les autres, il n'est pas résilient à 100 % aux attaques. Cependant, l'équipe RIPE Atlas propose régulièrement des améliorations et des fixations de bugs surmontées par la communauté RIPE Atlas.

I.3.8 Les ancres VS sondes Atlas

Les ancres Atlas sont des dispositifs agissant comme cibles aux différentes mesures lancées par les sondes Atlas. Il est possible de planifier des mesures entre les ancres RIPE Atlas, ces mesures permettent de vérifier l'état des réseaux qui hébergent ces ancres. Les ancres Atlas peuvent être considérées comme cibles aux mesures suivantes :

- Ping.
- Traceroute.
- DNS : les ancres ont été configurées avec BIND pour qu'elles agissent en tant que serveur DNS faisant autorité.
- HTTP et HTTPS : l'ancre fait tourner un serveur Web, ce dernier utilise un gestionnaire de réponses personnalisé aux requêtes HTTP(S) ayant comme seule option la taille du payload. Cette taille peut prendre une valeur maximale de 4096 et la réponse est fournie sous format JSON. L'exemple d'une requête HTTP avec une taille de 536 depuis une sonde Atlas vers une ancre Atlas est :

http ://nl-ams-as3333.anchors.atlas.ripe.net/536

Les ancres sont configurées avec un certificat SSL auto-signé en utilisant une clé de 2048 bit et un temps d'expiration de 100 ans. Le tableau I.2 reprend une comparaison de certaines caractéristiques communes entre les sondes et les ancres Atlas.

	Sonde Atlas	**Ancre Atlas**
Mesures originaires de	oui	oui
Mesures à destination de	— [4]	ping, traceroute, DNS, HTTP(S).
Nomination	—	structurée [5]
Crédit gagnés	N	$10 * N$
Besoin en bande passante	léger	important
Coût : gratuite	oui	non [6]

TABLE I.2 – Comparaison entre sondes et ancres RIPE Atlas

I.3.9 Les mesures intégrées : Built-in

Une fois une sonde Atlas connectée, elle lance automatiquement un ensemble de mesures prédéfinies, appelées *Built-in Measurements*. Les mesures personnalisées sont détaillées dans la section I.3.11. Le choix du mode IPv4, IPv6 ou les deux, dépend de la capacité du réseau qui héberge la sonde Atlas.

Il existe deux types de mesures : les mesures *One-Off*, ce sont les mesures qui s'exécutent une seule fois. Pour le deuxièmes type, ce sont les mesures qui s'exécutent périodiquement, à chaque intervalle de temps.

De base, les sondes Atlas assurent les mesures intégrées suivantes :

- Les informations sur la configuration du réseau dans lequel la sonde Atlas est déployée.
- L'historique de la disponibilité de la sonde Atlas.
- Les mesures du RTT (Round Trip Time) par traceroute.
- Les mesures ping vers un nombre de destinations prédéfinies.

4. — : Non disponible.
5. Exemple de *de-mai-as2857.anchors.atlas.ripe.net* avec la structure suivante : *pays-ville-ASN.anchors.atlas.ripe.net*.
6. Le matériel est au frais de l'hébergeur.

- Les mesures traceroute vers un nombre de destinations prédéfinies.
- Les requêtes vers les instances des serveurs DNS (Domain Name System) racines.
- Les requêtes SSL/TLS (Secure Socket Layer/Transport Layer Security) vers un nombre de destinations prédéfinies.
- Les requêtes NTP (Network Time Protocol).

Chaque mesure a un identifiant ID unique. Cet identifiant indique le type de la mesure, s'il s'agit du ping, traceroute ou autres. Plus de détails sur la signification des identifiants des mesures sont disponibles dans la section **??** dans l'annexe A.

En plus des mesures intégrées, les sondes Atlas peuvent effectuer des mesures personnalisées. Ces mesures peuvent être lancées via l'interface web [3] ou bien via HTTP REST API. Toutefois, la planification des mesures personnalisées nécessite l'acquisition de ce qu'on appelle les "crédits" au sens RIPE Atlas.

I.3.10 Le système de crédits Atlas

Le système de crédits RIPE Atlas est une sorte de reconnaissance de la contribution des participants à ce projet. Un hébergeur d'une sonde Atlas reçoit un nombre de crédits en contrepartie de la durée pendant laquelle sa sonde reste connectée. D'autre part, il gagne d'autres crédits suivant les résultats de mesures générés par cette sonde. Les crédits gagnés peuvent être utilisés dans la création des mesures personnalisées, appelées *User Defined Measurements* (voir la section I.3.11). Les personnes ayant gagné des crédits peuvent les transférer vers une autre personne ayant besoin de ces crédits. Les crédits peuvent être obtenus via :

- L'hébergement d'une sonde Atlas ; à chaque utilisation d'une sonde, son hébergeur reçoit un nombre de crédits. La connexion d'une sonde Atlas au système durant une minute apporte 15 crédits.
- L'hébergement d'une ancre Atlas [7].
- La recommandation à une personne d'héberger une sonde Atlas.
- En étant un sponsor du RIPE NCC. Le parrainage des sondes Atlas est disponible pour les organisations et les individus. Le sponsor reçoit le même nombre de crédits que les hébergeurs de ces sondes.
- En étant un registre Internet régional (Local Internet Registry).

7. Les ancres Atlas sont décrites dans la section I.3.8.

– La réception des crédits d'une autre personne via un transfert de crédits.

Le lancement des mesures personnalisées exploite les ressources de l'infrastructure RIPE Atlas d'une part, du réseau hébergeur de la sonde d'autre part. Par conséquent, les mesures sont organisées afin d'éviter toute surcharge du système. Le coût d'une mesure dépend du type de la mesure et des options spécifiées. Le système calcule le nombre de crédits nécessaires pour effectuer une mesure donnée. Le nombre de crédits est déduit à chaque résultat reçu. Ci-dessous le coût unitaire des différents types de mesures.

Ping et ping6 :

$$\text{Coût unitaire} = N \times (\lfloor \tfrac{S}{1500} \rfloor + 1)$$

Où N est le nombre de paquets dans le train (par défaut 3) et S est la taille du paquet (par défaut : 48 octets).

DNS et DNS6 :

Coût unitaire pour UDP : 10 crédits/résultat
Coût unitaire pour TCP : 20 crédits/résultat

Traceroute et traceroute6 :

$$\text{Coût unitaire} = 10 \times N \times (\lfloor \tfrac{S}{1500} \rfloor) + 1)$$

Où N est le nombre de paquets dans le train (par défaut 3) et S est la taille du paquet (par défaut : 40 octets).

SSLCert et SSLCert6 :

Coût unitaire = 10 crédits/résultat.

Exemple :
La planification d'une mesure ayant les caractéristiques suivantes nécessite $14,400$ crédits.

La fréquence	: deux fois par heure
La durée	: deux jours (48 heures)
Le nombre de sondes	: 5
Type de mesure	: *traceroute*

Tel que :

$$5 \times 2 \text{ mesures/heure} \times 48 = 480 \text{ ligne résultat}$$
$$30 \text{ credits/result} \times 480 \text{ results} = 14,400 \text{ crédits}$$

I.3.11 Les mesures personnalisées : User Defined mesurement

En plus des mesures intégrées, par défaut, dans une sonde Atlas, il est possible de planifier des mesures personnalisées. Ce sont les mêmes types de mesures : ping, traceroute, HTTP Get, SSLCert , DNS, NTP et TLS. Cette planification coûte des crédits, en effet, il faut avoir assez de crédits pour lancer des mesures. L'interface web dédiée à la création d'une nouvelle mesure offre toute les possibilités comme la précision des éléments suivants :

- Le type de la mesure.
- La sélection des sondes Atlas réalisant la mesure.
- La fréquence de la mesure et sa durée.

Chaque mesure est suivie via son état. Plusieurs états à distinguer : *specified*, *scheduled*, *ongoing*, *stopped*, *Forced to stop*, *no suitale probes* et enfin *failed*.

I.3.12 La sélection des sondes Atlas

La sélection des sondes Atlas pour effectuer une des mesures repose un des critères suivants :

- Numéro d'AS.
- Zone géographique via l'attitude et la longitude.
- Pays (ou zone géographique comme Europe).
- Préfixe IP.
- Manuellement, avec les identifiants des sondes Atlas.
- Reprendre celles d'une mesure précédente.

Il existe une autre manière de regrouper les sondes avec des étiquettes. Le système d'étiquettes sert comme indicateur des propriétés, des capacités, de la topologie du réseau ou d'autres classifications. On distingue les étiquettes système et utilisateur. Chaque nom d'étiquette est lisible par un humain.

Les étiquettes utilisateurs sont associées à une sonde librement par son hébergeur. Les étiquettes système sont attribuées uniquement par l'équipe RIPE Atlas et sont mises à jour périodiquement, à priori chaque 4 heures. Des exemples d'étiquettes système sont présentés dans la section **??** de l'annexe A.

I.3.13 Les sources de données Atlas

Les sondes Atlas génèrent trois types de données : leurs détails de connexions d'un jour donné, leurs résultats des mesures intégrées et personnalisées et les descriptions des mesures effectuées (meta-data).

Premièrement on trouve les données sur les sondes Atlas par jour. Les détails sur les sondes reprennent les informations des connexions, des réseaux et autres. A priori, les détails des connexions des sondes sont disponibles pour la période du 13 mars 2014 jusqu'à ce jour [8], un fichier JSON par jour (voir un exemple dans la section **??** de l'annexe A). Les données de certains jours sont manquantes. La totalité des archives se trouve dans [5]. La taille d'une seule archive est entre 120 Ko et 921 Ko.

Deuxièmement, les résultats des mesures sont aussi archivés dans un serveur FTP. Seules les données des derniers 30 jours sont conservées en archives [9]. Les fichiers ont été nommés jusqu'au 15 mars 2018 de façon structurée comme suit :

$TYPE-$IPV-$SUBTYPE-$DATE.bz2

- $TYPE peut être traceroute, ping, dns, ntp, http, sslcert.
- $IPV version du protocole IP v4 ou v6.
- $DATE date au format YEAR-MONTH-DAY. (etc. 2017-06-13)
- $SUBTYPE type de mesure builtin ou udm.

En considérant toutes les possibilités des types, la quantité de données générées quotidiennement est environ 25 Go [10] et la taille des archives est entre 281M et 3.2G.

Depuis 15 mars 2018, les résultats des mesures ont été regroupés différemment. 24 archives par jour, une seule archive pour chaque heure et type de mesure.

8. 15/08/2018.
9. Source : https://data-store.ripe.net/datasets/atlas-daily-dumps/, consultée le 05/04/2018.
10. Source : https://ftp.ripe.net/ripe/atlas/data/README, consultée le 26/03/2018.

L'archive ne distingue pas entre mesures IPv4 et IPv6, entre mesures intégrées et personnalisées. Il existe un attribut **"af"** qui distingue entre IPv4 et IPv6 et l'identifiant de la mesure pour distinguer les mesures intégrées et celles personnalisées (identifiant > $1,000,000$).

Streaming API propose un service de récupération des résultats de mesures en temps réel, depuis les sondes publiques. Ainsi, elle fournit continuellement de nouveaux résultats en temps réel, obtenus par les sondes Atlas publiques, via une connexion de type HTTPS web-socket active tout le temps.

Troisièmement, on trouve des archives sauvegardées chaque semaine, elles décrivent les méta-datas des mesures. Une ligne objet JSON pour chaque mesure publique. Au moment de la consultation, la taille de chaque archive était entre 124 Mo et 1.5 Go. L'accès à ce jeu de données se fait de deux façons, via le téléchargement direct depuis un serveur FTP ou bien via streaming API. Les noms des archives sont bien structurés.

I.3.14 Les versions du firmware des sondes Atlas

En principe, toutes les sondes Atlas collectent la même information, indépendamment de leur version du firmware. On trouve les mêmes attributs [11] dans toutes les versions sauf de léger changements : ajout d'un ou de plusieurs attributs, la modification des noms des attributs, etc. Pour la simplification, nous donnons un identifiant entier pour chaque version, entre les parenthèses. Cet identifiant sera utilisé dans la suite de ce document.

Il existe plusieurs versions du firmware :

- La version 1 est identifié par 1 (1).
- La version 4400 est identifiée par une valeur entre 4400 et 4459 (2).
- La version 4460 est identifiée par une valeur entre 4460 et 4539 (3).
- La version 4540 est identifiée par une valeur entre 4540 et 4569 (4).
- La version 4570 est identifiée par une valeur entre 4570 et 4609 (5).
- La dernière version du firmeware [12] est 4610 (6).

I.3.15 Les limitations du RIPE Atlas

De nombreux travaux ayant exploité les données générées par les sondes Atlas. Néanmoins, ce système connaît des bugs et des limitations. Les membres

11. Attribut dans le sens du JSON : chaque résultat de mesure est enregistré comme étant un objet JSON.
12. A la date de consultation $25/01/2018$.

de la communauté RIPE Atlas s'engagent à remonter les bogues liées aux sondes Atlas. Tous les bogues sont répertoriés sous une rubrique dédiée [6].

RIPE Atlas connaît des limitations liées à la visualisation. Actuellement, RIPE Atlas supporte la visualisation des mesures de type ping ayant utilisé au maximum 20 sondes. Cette limitation concerne aussi le type traceroute, en effet, il est possible de visualiser seulement les mesures IPv6 built-in.

Afin d'éviter la surcharge des sondes et de l'infrastructure, RIPE Atlas a limité le nombre de mesures périodiques de 10 à la fois et de 10 mesures de type one-off vers n'importe quelle cible à un moment donné. De plus, il n'est pas possible d'utiliser plus de 500 sondes par mesure.

Pour les mesures one-off (non périodiques), une sonde peut effectuer au plus 10 mesures en parallèle. RIPE Atlas limite aussi la fréquence des mesures personnalisées. Un hébergeur d'une sonde peut effectuer :

- Ping chaque 60 secondes (par défaut 240 secondes).
- Traceroute chaque 60 secondes (par défaut 900 secondes).
- SSL chaque 60 secondes (par défaut 900 secondes).
- DNS chaque 60 secondes (par défaut 240 secondes).

Dans le cas d'une déconnexion, la sonde continue à effectuer les mesures. Pour la version 1 et 2, la sonde est capable de sauvegarder les 6 dernières heures de données. Tandis qu'avec la version 3, une sonde est capable de sauvegarder les résultats de plusieurs mois. Une fois la sonde est connectée, elle envoie les données à l'infrastructure centrale.

Concernant la consommation des crédits, RIPE Atlas limite cette consommation à $1,000,000$ crédits par jour.

I.3.16 Confiance aux données Atlas

De nombreux travaux ont exploité les données Atlas, cependant, peut-on faire confiance à la qualité des données ? les données sont-elles complètes ?

La question de la complétude des données est plus présente pour les mesures périodiques, celles qui se déroulent pendant une durée d et à un intervalle i. W. Shao et al. [25] ont traité les mesures manquantes. L'approche qu'ils ont adopté repose sur la corrélation entre l'absence de certaines mesures et les périodes durant lesquelles les sondes Atlas sont déconnectées. Pour précision, RIPE Atlas maintient les détails des connexions/déconnexions des sondes Atlas. Ils ont étudié les mesures en provenance des sondes v3, effectuées entre le $01/06/2016/$ et le

$01/07/2016/$ (UTC). Ils ont combiné les informations relatives à la connexion/-déconnexion des sondes et leurs mesures planifiées, leur approche se base sur l'attribut $timestamp$ qui est présent dans chaque résultat de mesure et dans les états de connexions.

Nous avons discuté des limitations du RIPE Atlas en terme de mesures autorisées par jour. Cela n'empêche qu'il est possible qu'un nombre important de mesures soit effectué. De plus, plus d'un utilisateur peut s'intéresser à la même sonde Atlas. C'est la question traitée dans le travail de T. Holterbach et al. dans [19], si les mesures lancées par les autres utilisateurs affectent les résultats obtenus par un autre utilisateur, si c'est le cas, comment s'y entreprendre. Les expériences réalisées ont montré la présence de l'interférence entre les mesures à destination des sondes et cela de deux manières. Premièrement, les mesures depuis et à destination des sondes Atlas augmentent le temps reporté par la sonde et ils ont conclu que l'amélioration du CPU a permis de limiter les interférences sur le temps mesuré par les sondes Atlas. Deuxièmement, ils ont conclu que les mesures perdent la synchronisation avec l'infrastructure d'Atlas, pendant plus d'une heure, à cause de la charge concurrentielle que subit le système d'Atlas. Dans ce cas, l'amélioration du matériel ne peut pas résoudre le problème.

I.4 Projets existants de mesures d'Internet

Dans les sections précédentes, on a développé le projet RIPE Atlas comme étant une plateforme pour la collecte des données des réseaux. Toutefois, il existe d'autres projets similaires à RIPE Atlas. Les sections suivantes reprennent une liste non exhaustive des projets similaires à RIPE Atlas.

I.4.1 Test Traffic Measurement Service

Avant l'arrivée du RIPE Atlas, Le RIPE NCC (Réseaux IP Européens Network Cordination Centre) a assuré la mesure de la connectivité entre les réseaux via d'autres plateformes, comme la plateforme Test Traffic Measurement Service (TTM). Il s'agit d'un projet qui permet de mesurer la connectivité entre un nœud source et un nœud destination sur Internet. C'était une des manières pour suivre la connectivité entre le réseau source et le réseau destination.

L'idée était la mise en place d'un dispositif, test-box, qui génère du trafic. Ce dernier n'affecte pas l'infrastructure réseau en matière de bande passante. De plus, il n'a pas l'accès aux données du réseau dans lequel il est mis en place.

Ce service a été assuré et géré, pendant une période de 6 ans, par une équipe au sein du RIPE NCC. Les fonctionnalités assurées par ce service étaient de tester l'accessibilité à une destination via le *ping*, ainsi, les mesures effectuées étaient

indépendantes des applications, elles dépendaient du réseau lui-même. RIPE NCC a arrêté la maintenance du TTM depuis le 1 juillet 2014 [23].

I.4.2 ProbeAPI

ProbeAPI [27] est une plateforme de mesure d'état du réseau, cette plateforme couvre 170 pays et des milliers d'ISPs. *ProbeAPI* est utilisée par les développeurs, les administrateurs des réseaux et les chercheurs, ils peuvent lancer des mesures d'un réseau depuis différents réseaux.

Le logiciel *ProbeAPI* s'exécute dans plusieurs systèmes : dans des ordinateurs (Win32/64), Android via une installation dans les mobiles et les tablettes et dans des routeurs au sein du DD-WRT.

> **DD-WRT** est un micrologiciel libre et gratuit, il est destiné aux routeurs sans fil et aux points d'accès. Il fonctionne avec un système d'exploitation Linux. Le rôle du DD-WRT est de remplacer le micrologiciel intégré aux routeurs par leurs fabricants. Ainsi, il est possible d'étendre des fonctionnalités du routeur en ajoutant d'autres fonctions supplémentaires.

ProbeAPI s'agit d'un logiciel qui tourne dans la machine de l'hébergeur. En conséquence, le suivi des réseaux dépend de la disponibilité de la machine qui le fait tourner. Cette dépendance affecte la disponibilité de la sonde logicielle, sa configuration et aussi les résultats de mesures.

Une étude comparative [28] entre les sondes Atlas et les sondes *ProbeAPI* est résumée dans le tableau I.3. En fin de cette étude, ils concluent qu'en comparant les résultats des mesures ICMP effectuées par les deux plateformes, des contrastes intéressantes ont été constatées. Les sondes Atlas ont montré un comportement stable lors de la réalisation des mesures, les résultats sont peu variables car les sondes sont indépendantes de l'utilisateur. Cependant, il était constaté qu'une forte variabilité au cours du temps pour les sondes logicielles (ProbeAPI), car elles dépendent fortement de l'hébergeur ; sa configuration réseau, sa disponibilité, etc.

Enfin, la force des sondes logicielles comme ProbeAPI réside dans sa capacité à effectuer des mesures depuis la couche application, la plus proche de l'utilisateur. L'exemple de l'évaluation du Time To First Byte et le taux de transfert dans deux pays.

« *Le **Time to First Byte** (TTFB) est le temps de chargement du premier octet, c'est la mesure qui nous permet d'évaluer la vitesse d'accès à un serveur. Plus la mesure est basse et plus le serveur commencera à servir les ressources rapidement.* »[a]

a. Source : `https://www.skyminds.net/calculer-le-time-to-first-byte-ttfb-dun-serveur/`, consultée le 10/08/2018.

RIPE ATLAS	PROBEAPI
Matériel homogène a un comportement prévisible	Matériel hétérogène a un comportement imprévisible
Connexions stables vu l'indépendance du software utilisateur	Connexions instables vu la dépendance du software utilisateur
Indépendance de l'OS et ses limitations ou vulnérabilités	Liaison à l'OS et ses limitations ou vulnérabilités, cependant utile pour les mesures au niveau application
La distribution des sondes est coûteuse, difficile de couvrir certaines régions	Mise en place du logiciel est rapide et moins chère, avec facilité de couvrir plusieurs régions
Les mesures HTTP se limitent aux ancres pour des raisons de sécurité	HttpGet, DNS et page-load sont disponibles via des librairies Mozilla et chromium, et ce pour toutes les destinations

TABLE I.3 – Comparaison entre sondes Atlas et ProbeAPI

Malgré le niveau de couverture assuré par ProbeAPI, cependant ces sondes se connectent et se déconnectent fréquemment, ce qui montre une forte volatilité. Cette volatilité est liée à la dépendance des sondes ProbeAPI de leur hébergeur ; tant qu'il est connecté, la sonde ProbeAPI est prête pour effectuer les mesures. Toutefois, si l'hébergeur est déconnecté, la sonde ProbeAPI ne peut pas effectuer des mesures, d'où le basculement fréquent entre les deux états : connectée et déconnectée.

I.4.3 Archipelago

Archipelago (Ark) [1] est l'infrastructure de mesures actives du CAIDA [2]. Elle est au service des chercheurs en réseau depuis 2007. L'objectif de ce projet est de couvrir un maximum de régions afin de collecter un maximum de mesures.

Ensuite, produire des visualisations qui améliorent la vue globale de l'Internet. Pour précision, c'est un Raspberry Pi 2nd gen.

I.4.4 DIMES

DIMES [26] est un logiciel qui devrait être installé dans une machine. Une fois installé, il fonctionne de sorte que la consommation d'énergie soit minimale et qu'il n'existe aucun impact sur les performances de la machine ou sur la connexion. L'objectif de *DIMES* est de collecter un maximum de données afin d'explorer la topologie d'Internet.

I.4.5 SamKnows

SamKnows [7] est une plateforme globale des performances d'Internet, elle regroupe les ISPs, ingénieurs, universitaires, codeurs et des organismes de régulation. Son objectif est d'évaluer les performances du haut débit des utilisateurs finaux et de trouver les problèmes avant que les clients ne commencent à se plaindre.

I.5 Quelques cas d'utilisation des données collectées par les sondes Atlas

Plusieurs travaux ont exploités les données collectées par les sondes Atlas. Ces travaux peuvent être classés de plusieurs manières, par exemple par thème, par type de mesures utilisé, etc. Nous distinguons les travaux ayant exploité les données collectées par les sondes Atlas à travers les mesures *built-in* ou bien ceux ayant utilisé les données des mesures personnalisées. Pour les premiers, ils permettent d'exploiter au mieux ces données sans surcharger le réseau des sondes Atlas, car ces données sont collectées quotidiennement. Cependant, les autres peuvent introduire une charge sur ces sondes. D'autre part, certains auteurs se sont intéressés aux données traceroute, d'autres aux données ping ou HTTP, etc. Nous allons présenter brièvement quelques travaux par thème.

I.5.1 Détection des coupures d'Internet

Les données collectées par les sondes Atlas ont permis de valider certaines coupures d'Internet, par exemple la coupure concernant le point d'échange AMS-IX (Amsterdam Internet Exchange). En 2015, Robert Kisteleki et al. [20] ont évalué l'état des pings en provenance des sondes Atlas à destination de trois ancres

Atlas qui se trouvent dans AMS-IX. En effet, peu de pings ont réussi d'atteindre leurs destinations, cependant, certains pings n'ont pas réussi à le faire. Ils ont conclu qu'il existe un problème du réseau, et le problème concerne les ancres plutôt que les sondes ayant lancé le ping. De même pour DNS, ils ont constaté l'absence des données DNS sensées être collectées par les ancres Atlas à destination du K-root.

I.5.2 Aide à la prise de décision

L'utilisation des sondes Atlas n'est pas limitée au domaine de recherche seulement, elle a permis aussi d'aider à la prise de décision pour certaines implantations et pour la mise en place des équipements comme les routeurs, les data-centers, les IXPs, etc.

Les ingénieurs de *Wikimedia Foundation* et du RIPE NCC ont collaboré dans un projet [9] pour étudier la latence vers les sites du Wikimedia. L'idée était d'exploiter la distribution des sondes Atlas dans le monde en vue de mesurer la latence vers les sites du Wikimedia. L'étude de la latence va permettre d'améliorer l'expérience des utilisateurs vers ces sites en réduisant la latence. Comme Wikimedia avait l'intention d'étendre son réseau de datacenters, ils ont profité des résultats de cette étude pour choisir les futurs emplacements de leurs data-centers.

Un groupe de chercheurs africains a évalué le routage inter-domaine afin d'étudier les emplacements adéquats pour la mise en place d'un IXP [24]. Après avoir analysé les données des mesures collectées par les sondes Atlas, ils ont constaté que le trafic de et à destination de l'Afrique quitte le continent vers les États-Unis ou bien l'Europe pour revenir en Afrique, d'où l'intérêt d'investir dans la mise en place des IXPs dans ce continent.

I.5.3 Le suivi des censures

En 2014, des chercheurs ont examiné les incidents de type content-blocking en Turquie et en Russie tout en prenant en considération le respect de l'aspect éthique des données. Ils ont aussi élaboré un aperçu comparatif des différents outils permettant de mesurer les réseaux [12]. C. Anderson et al. ont repris deux cas d'études où une censure a été appliquée : la Turquie et la Russie. L'idée de C. Anderson et al. est de créer des méthodes pour analyser ces censures en se basant sur les données collectées par les sondes Atlas.

Il existe plusieurs pratiques pour appliquer la censure. Ces pratiques dépendent des objectifs de cette censure ; bloquer un site web, rediriger le trafic, filtrer l'accès à travers des mots clés, etc.

En mars 2014, des utilisateurs turcs ont été interdits d'accéder au réseau social *Twitter*. Ce filtrage a été fait en utilisant *DNS Tampering* et *IP Blocking*.

Comme ces deux pratiques sont évaluables avec les sondes Atlas, ils ont planifié des mesures vers plusieurs destinations et depuis un nombre de sondes. Ces mesures sont reprises en détail dans le tableau I.4.

Cible	Type	Sondes	Fréquence (s)	Crédits
Twitter	SSL	10	3,600	2,400
YouTube	SSL	10	3,600	2,400
Tor	SSL	10	3,600	2,400
Twitter	DNS (U)	10	3,600	2,400
YouTube	DNS (U)	10	3,600	2,400
Twitter	Tracert	10	3,600	7,200

TABLE I.4 – Les détails des mesures effectuées dans le travail de C. Anderson [12]

L'analyse de données obtenues a permis de détecter six changements concernant les décisions du filtrage. Plus de détails se trouvent dans [12].

Quant à la Russie, les autorités ont décidé de mettre le blog d'*Alexei Navalny* sur *LiveJournal* dans la liste noire. En même temps, certains médias indépendants ont été aussi filtrés, l'exemple du site grani.ru. Pour le site *Grani*, les sondes Atlas ont reçu des réponses DNS aberrantes, d'où l'impossibilité de joindre *grani.ru*. Cependant, le filtrage du site *navalny.livejournal.com* a pris une autre forme, c'était une redirection d'adresse IP. La réponse d'une requête vers ce site donne 208.93.0.190 au lieu de 208.93.0.150. Ces deux adresses sont inclut dans le préfixe 208.93.0.0/22 géré par *LiveJournal Inc*. 208.93.0.190 correspond au contenu non-blacklisted, alors que 208.93.0.150 correspond au contenu correct.

I.5.4 Le suivi des performances d'un réseau

Les ancres Atlas

Les ancres Atlas ont des capacités avancées que les sondes Atlas. Les ancres servent comme cibles aux mesures des sondes. De plus, elles sont capables de fournir des détails sur l'état du réseau dans lequel elles sont déployées. S. Gasmi, un hébergeur d'une ancre Atlas, a développé un outil disponible au public [13]. A partir des données collectées par les ancres Atlas, cet outil permet d'analyser la qualité de la connectivité d'un réseau (ou d'un AS) et permet de suivre les changements relatifs à la topologie des réseaux

Par exemple, il a constaté que la vérification du BGP Prepending et des communautés BGP peut être faite en considérant les éléments suivants : adresse IP

13. Source : `http://ripeanchor.sdv.fr/`, consultée le 08/08/2018.

source, AS source, pays, le RTT du ping, les chemins du traceroute. En particulier, S. Gasmi a évalué deux corrélations. Dans un premier temps, il a visualisé la corrélation entre l'AS path et Round Trip Time (RTT). Il a regroupé des sondes par pays, ensuite, il a calculé, par ce pays, la moyenne du nombre de sauts et la moyenne du RTT des requêtes à destination de l'ancre depuis ces sondes Atlas. La figure I.8 reprend les résultats obtenus. Aucun renseignement sur la période des données. Pour les sondes en provenance de la France, le nombre de sauts et le RTT entre les sondes déployées en France sont faibles car l'ancre (la cible) se trouve aussi en France.

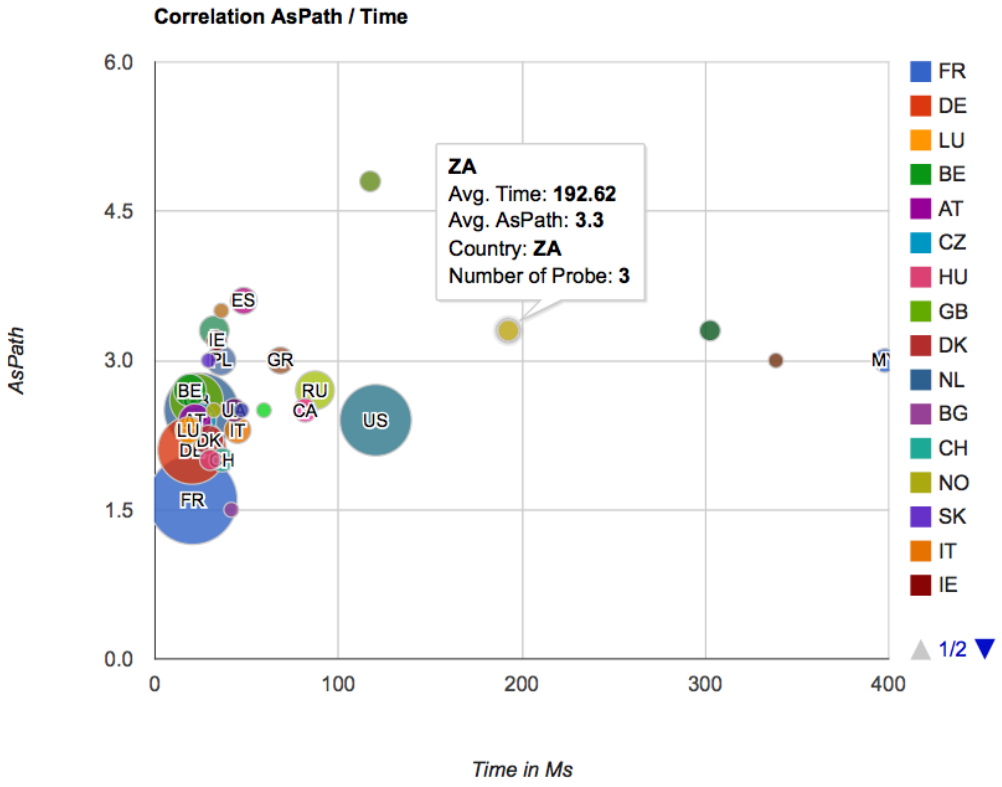


FIGURE I.8 – La corrélation entre la moyenne des AS paths et la moyenne des RTTs [15]

Ensuite, S. Gasmi a mesuré le RTT entre des sondes Atlas dans le monde et son ancre, il a aussi visualisé le nombre de sauts parcourus entre des sondes Atlas à travers le monde et son ancre. Ces deux visualisations permettent d'avoir une idée sur la latence entre certains pays et le pays de l'ancre en question. Plus de détails sur l'approche sont disponibles dans [15].

La vérification de la cohérence du Traceroute

Les chemins parcourus par traceroute pour aller d'une source s vers une destination d changent au cours du temps pour plusieurs raisons. Par exemple, suite à un changement BGP, à une répartition des charges, à des pannes des routeurs, à des pannes des liens physiques, etc.

Traceroute Consistency Check peut reprendre les chemins obtenus via traceroute au cours du temps . L'objectif est de suivre les nœuds apparaissant dans le chemin allant de s à d aux instants t, $t+1$, $t+2$, etc, et cela afin de voir les nœuds traversés plus fréquemment au cours du temps. Le chemin est mis à jour via Atlas streaming API.

L'outil proposé dessine les chemins traceroute comme étant un graphe dirigé, chaque nœud est coloré suivant sa cohérence. Le code source du projet est disponible sur GitHub [13]. La figure I.9 présente un exemple de la visualisation proposée. Ce résultat concerne la mesure 1663314 [14]. Ce sont des traceroutes à destination de l'adresse 213.171.160.1 entre 02/05/2014 13 : 00 et 03/05/2014 15 : 00.

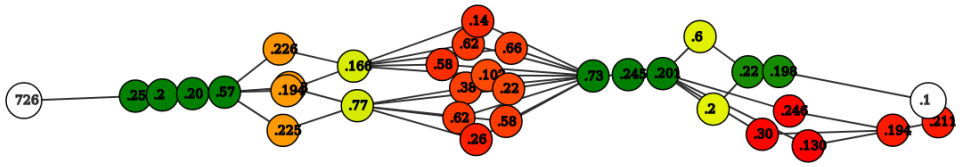


FIGURE I.9 – Visualisation des changements des chemins traceroute [13]

BGP+traceroute

C'est une combinaison des données BGP (RIPE RIS) et traceroute (RIPE Atlas). L'objectif de ce projet était de partir d'un AS path pour enfin géolocaliser les ASs. L'idée est de prendre un AS path des données RIPE RIS, puis, récupérer le préfixe (bloc d'adresses IP) annoncé via cet AS path, ensuite, lancer un traceroute vers une des adresses du bloc. Enfin, géolocaliser les ASs via les données du traceroute. Le code source et la présentation de ce projet sont disponibles GitHub [18, 17].

14. Source : `https://atlas.ripe.net/measurements/1663314/`, consultée le 05/08/2018.

BGP Atlas Monitor "BAM !"

Le projet *BAM* vise la visualisation , en temps réel, des informations utiles pour les opérateurs des réseaux. Par exemple, BAM montre la visibilité des préfixes obtenus par RIPE RIS. De plus, il est possible de voir le délai du *ping* obtenu via les sondes Atlas. Le code source est disponible sur GitHub [16]. En fournissant un ASN (identifiant d'un AS), *BAM* récupère les préfixes IPv4 et IPv6 et leur visibilité et il montre aussi les sondes dans cet AS. L'outil offre les fonctionnalités suivantes :

- Les préfixes annoncés par un ASN.
- La visibilité d'un ASN.
- La visibilité d'un préfixe.
- La liste des sondes par AS.
- Les objets route des préfixes.

Prédiction des routeurs provoquant la perte des paquets

Dans l'étude [14], Romain Fontugne et al. ont modélisé le comportement des routeurs, ils ont développé un modèle qui permet d'estimer l'endroit de la perte des paquets. A partir des traceroutes passant par un routeur r à une destination d, ils ont construit un modèle de forwarding pour ce routeur. Ce modèle reprend les prochains sauts (routeurs) et la fréquence de passage par ces derniers. Si le routeur r change le prochain saut qui a eu "l'habitude" de traverser pour atteindre d, alors, il est possible d'estimer l'origine de la perte de paquets.

I.5.5 Le suivi des détours dans un trafic local

Dans leur travail [11], E. Aben et al. avaient l'objectif de voir comment les mesures du RIPE Atlas peuvent fournir un aperçu sur le chemin du trafic local à un pays. Précisément si ce trafic traverse un autre pays en revenant au pays du départ. Ce qui pourrait aider à améliorer les performances et l'efficacité des IXPs. L'objectif est d'analyser les chemins identifiés dans le trafic d'Internet entre les sondes Atlas dans un pays donné et essayer d'identifier si le trafic traverse les IXPs.

France-IX est un point d'échange Internet (IXP) français créé en juin 2010. Afin d'apprendre la topologie de routage, un RIS route collector (RRC21) a été installé au sein du France-IX. Actuellement, la France compte 755 sondes Atlas et 9 ancres. Une ancre sur les 9 est installée au sein de France-IX.

Une des questions posées c'était si le trafic local de la France reste local, les sondes Atlas ne permettent pas de mesurer le trafic entre deux points, cependant, elles permettent de calculer le chemin entre deux points, adresses IP, ce qui permet d'inférer les sauts par lesquels le trafic passe le trafic. Le travail [10] s'intéresse au trafic depuis et vers une sonde en France en se basant sur l'étude dans [11].

Les résultats obtenus de l'analyse des détours peuvent être intéressants pour les opérateurs des réseaux afin d'améliorer leurs services, ainsi intéressants pour les IXPs tels qu'ils peuvent proposer des services de peering dans les endroits où il le faut.

I.5.6 Visualisation : indicateurs et dashboard

L'objectif de certains travaux était d'exploiter les données collectées par les sondes Atlas pour concevoir des tableaux des indicateurs. Par exemple, à partir des données de connexion/déconnexion des sondes Atlas, visualiser les sondes connectées, déconnectées, abandonnées. Un autre projet avait comme objectif la reconstruction d'un graphe reprenant les routeurs (nœuds) impliqués dans certains traceroutes, ainsi, identifier les nœuds les plus traversés. D'autres travaux ont repris les détails de la latence, essentiellement, sont les valeurs des RTT dans les pings et les traceroutes qui permettent de visualiser ce type d'information.

La liste des travaux basés sur le projet RIPE Atlas est très longue. Nous avons essayé d'énumérer quelques projets, les classer par thèmes, toutefois, ce n'est pas un classement unique, tel qu'on peut retrouver un travail dans plus d'une catégorie, ou bien les classer par un autre classement.

I.6 Conclusion

Dans ce chapitre, nous avons présenté les sondes Atlas et leur fonctionnement, ainsi que quelques travaux qui ont impliqué les données collectées par ces sondes dans plusieurs domaines tels que la prise de décision, le suivi des censures, la conception des tableaux de visualisation, etc. Ces données sont cruciales pour mener à toute analyse. Cependant, ces données sont massives, elles sont dans l'ordre d'une dizaine de Go pour une heure de mesures du type traceroute par exemple, y incluent toutes les destinations, d'où la nécessité d'impliquer des outils du Big Data pour une meilleure extraction d'informations utiles. En effet, le chapitre 2 aborde le sujet du Big Data dans ses différentes dimensions.

Chapitre II

Algorithme de détection des anomalies

II.1 Introduction

Dans leur travail [14], R. Fontugne et al. ont exploité la distribution répandue des sondes Atlas dans le monde afin d'étudier un des problèmes relatifs aux performances des réseaux informatiques.

Il est difficile d'avoir une idée globale sur la topologie de l'Internet. Toutefois, les opérateurs des réseaux informatiques disposent d'un aperçu de l'état des entités qui forment leurs réseaux, les relations entre ces entités ainsi que les éventuels problèmes. Avec la distribution abondante des sondes Atlas dans le monde en terme de type d'adressage : sondes Atlas supportant seulement l'adressage IPv4, d'autres qui supportent en plus l'adressage IPv6, en terme de la diversité géographique, la diversité en terme d'ASs hébergeant les sondes Atlas, etc, il était possible d'aborder les délais dans les réseaux informatiques à travers de nouvelles approches, reposées sur des fondements statistiques. Parmi les points forts de l'analyse menée par R. Fontugne et al., c'était la possibilité de valider les méthodes proposées avec des événements marquants sur Internet.

Le travail de R. Fontugne et al. reprend trois méthodes basées sur les données collectées par les sondes Atlas, chaque méthode reflète l'approche utilisée pour étudier les performances des réseaux informatiques. Ces méthodes sont les suivantes :

1. la détection des changements des délais que subissent les liens intermédiaires dans les traceroutes ;

2. la conception d'un modèle de forwording pour un routeur donné. Ce modèle prédit l'acheminement du trafic afin d'identifier les routeurs et les liens en

panne dans le cas d'un problème de perte de paquets ;

3. la création d'un score par Système Autonome afin d'évaluer l'état de ce dernier.

Dans la suite de ce travail, nous allons reprendre seulement la première méthode. Il s'agit d'étudier le délai d'un lien topologique, c'est le délai entre deux routeurs adjacents sur Internet.

II.2 L'étude des délais des liens

II.2.1 Les données utilisées dans l'analyse des délais

La méthode conçue pour la détection des changements des délais se base sur des fondements statistiques. Ces derniers sont capables de montrer leurs performances si la taille des échantillons [1] considérés est grande. Afin de surveiller un grand nombre de liens sur Internet, il faut avoir un grand nombre de sondes Atlas avec une certaine diversité et qui sont capables de collecter une quantité importante de données relatives aux performances des réseaux informatiques, c'est ce qu'assure le projet RIPE Atlas. Le travail de référence implique principalement les mesures de traceroutes, ainsi deux catégories de mesures sont utilisées :

– *builtin* : ce sont les traceroutes effectués par toutes les sondes Atlas vers les instances des 13 serveurs DNS racines. Les traceroutes sont effectués chaque 30 minutes. En pratique, certains serveurs racines DNS déploient l'anycast. Au moment de la réalisation du travail de référence, c'étaient des traceroutes vers 500 instances des serveurs DNS racines ;

> **DNS Anycast** est une solution utilisée pour accélérer le fonctionnement des serveurs DNS. Les serveurs DNS adoptant cette approche fournissent des temps de réponse plus courts, et ce partout dans le monde. Les requêtes en provenance de l'utilisateur sont redirigées vers un nœud adéquat suivant un algorithme prédéfini.

– *anchoring* : ce sont les traceroutes effectués par environ 400 sondes Atlas à destination de 189 serveurs [2] et ce chaque 15 minutes.

1. L'échantillon de la métrique qui caractérise un lien : RTT différentiel.
2. Sondes Atlas ayant des fonctionnalités avancées.

En ce qui concerne les traceroutes analysés, le tableau II.1 reprend plus de détails.

	Nombre de traceroutes	**Nombre de sondes**
IPv4	2.8 billion	11,538
IPv6	1.2 billion	4,30

TABLE II.1 – Récapitulatif des traceroutes utilisés dans le travail de référence

L'étude des délais ne concerne pas les adresses privées, ainsi, le suivi des délais ne concerne pas les réseaux privés. De plus, ce suivi se base sur les requêtes de type traceroute, et traceroute reprend une partie de la topologie de l'Internet. En effet, les liens considérés sont ceux topologiques et ne sont pas les liens physiques.

II.3 La description de la détection des délais anormaux des liens

II.3.1 RTT différentiel

Il est indispensable de présenter la définition du RTT (Round Trip Time) différentiel d'un lien avant de procéder à la description de l'algorithme de la détection des anomalies.

Le **temps RTT** est obtenu en calculant la différence entre le timestamp associé à l'envoi du paquet sondé et le timestamp associé à la réception de la réponse ICMP [a]. C'est une métrique pour évaluer les performances d'un réseau en matière de temps de réponse. Les mesures du RTT sont fournies avec l'utilitaire traceroute et ping. En ce qui concerne traceroute, ce dernier fournit les sauts impliqués dans le chemin de forwarding, c'est le chemin parcouru par le trafic entre la source et la destination. Le temps RTT inclut le temps pour atteindre un saut dans le sens du forwarding, à ce temps, il s'ajoute le temps de propagation des réponses. De plus, il y a le temps de traitement des requêtes au niveau des routeurs.

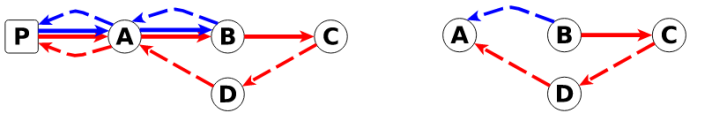
a. Internet Control Message Protocol est un protocole utilisé pour véhiculer des messages de contrôle sur Internet.

La figure II.1 (a) illustre le RTT entre la sonde P et les deux routeurs B et C. Le RTT différentiel entre deux routeurs B et C adjacents, noté Δ_{PBC}, est la

différence entre le RTT entre la sonde P et B (bleu) d'une part, et le RTT entre la sonde P et C (rouge) dans la figure II.1 (b).

$$\begin{aligned}\Delta_{PBC} &= RTT_{PC} - RTT_{PB} \\ &= \delta_{BC} + \delta_{CD} + \delta_{DA} - \delta_{BA} \\ &= \delta_{BC} + \varepsilon_{PBC}\end{aligned}$$

où δ_{BC} est le délai du lien BC et ε_{PBC} est la différence entre les deux chemins de retour (B vers P et C vers P).



(a) Round-trip to router B (blue) and C (red).

(b) Difference of the two round-trips (Δ_{PBC}).

FIGURE II.1

II.3.2 Le principe de la détection des changements des délais

Le suivi du délai d'un lien est déduit du suivi de l'évolution de son RTT différentiel. Reprenons la formule du RTT différentiel du lien BC : δ_{BC} + ε_{PBC}. Supposons qu'on dispose d'un nombre n de sondes Atlas P_i, $i \in [1, n]$, telles que toutes les sondes ont un chemin de retour différent depuis B et depuis C. En effet, les RTTs différentiel pour chacune des sondes Atlas Δ_{P_iBC} partagent la même composante δ_{BC}, toutefois, ces RTTs ont des valeurs des ε_{P_iBC} indépendantes. L'indépendance de ces valeurs implique que la distribution Δ_{P_iBC} est estimé d'être stable au cours du temps si δ_{BC} est constant. Cependant, un changement significatif de la valeur du δ_{BC} influence les valeurs des RTTs différentiel, dans ce cas, la distribution des RTTs différentiel changes si δ_{BC} change. Enfin, les changements des délais sont déduits des changements des RTTs différentiel qu'on peut les quantifier.

La détection des anomalies en délais repose sur un théorème très important en statistiques, c'est le théorème central limite (TCL). Ce théorème annonce que si on a une suite de variables aléatoires indépendantes ayant la même espérance et la même variance, la moyenne de ces variables aléatoires est une variable aléatoire qui suit une loi normale. De manière générale, le théorème central limite explique la distribution des moyennes des échantillons. Ce théorème peut être appliquer

aux différents lois. Par exemple la loi normale [3], binomiale, etc.

II.3.3 Les résultats de l'analyse des délais des liens

Nous distinguons deux sortes de résultat à l'issu de l'analyse des changements des délais. Premièrement, ce sont les changements identifiés tout au long de la durée de l'analyse, chaque changement est caractérisé par un ensemble de détails. Pour le deuxième résultat, ce sont des graphiques reprenant les changements ainsi que les changements jugés anormaux, ce qu'on va appeler par la suite par *anomalies*, précisément ces graphiques présentent l'évolution du RTT différentiel d'un lien au cours du temps.

II.3.4 Présentation de l'algorithme de la détection des changements anormaux : Résultat I

Description de l'algorithme de détection

Les étapes principales de la détection des changements des délais sont résumées dans l'algorithme 1. Nous allons détailler chaque étape : ses objectifs, ses entrées et ses sorties. En ce qui concerne l'entrée du programme principal, ce sont les paramètres présentés dans la section II.3.4.

Algorithm 1 Les étapes de la procédure detectRttChangesMongo()

```
1 : procedure DETECTRTTCHANGESMONGO(expId)
2 :     for all currDate ∈ dates do
3 :         computeRtt()
4 :         mergeRttResults()
5 :         outlierDetection()
6 :     end for
7 : end procedure
```

Description des paramètres de l'analyse des délais

La détection des changements des délais nécessite l'ajustement d'un nombre de paramètres. La valeur de chaque paramètre est relative au fondement utilisé théorique ou bien empirique, qui a été justifié par les auteurs du travail de référence. Ci-dessous les paramètres à ajuster avant de lancer une analyse. Chaque paramètre sera défini dans son contexte.

3. Un exemple illustratif dans A.

start : c'est la date de début de l'analyse. Ce sont les traceroutes capturés par les sondes Atlas à partir de cette date qui seront analysés.

end : c'est la date marquant la fin de l'analyse. Comme le paramètre *start*, c'est la date des derniers traceroutes capturés par les sondes Atlas à considérer dans la présente analyse.

timeWindow : ce paramètre est exprimé en seconde. La durée de l'analyse, qui est le temps écoulé entre *start* et *end*, est divisée sur des périodes de même taille : *timeWindow*. Pour chacune de ces périodes, on caractérise les liens identifiés sur les traceroutes capturés en cette période. La figure II.2 reprend le contexte des trois paramètres *start*, *end* et *timeWindow* avec les étapes principales.

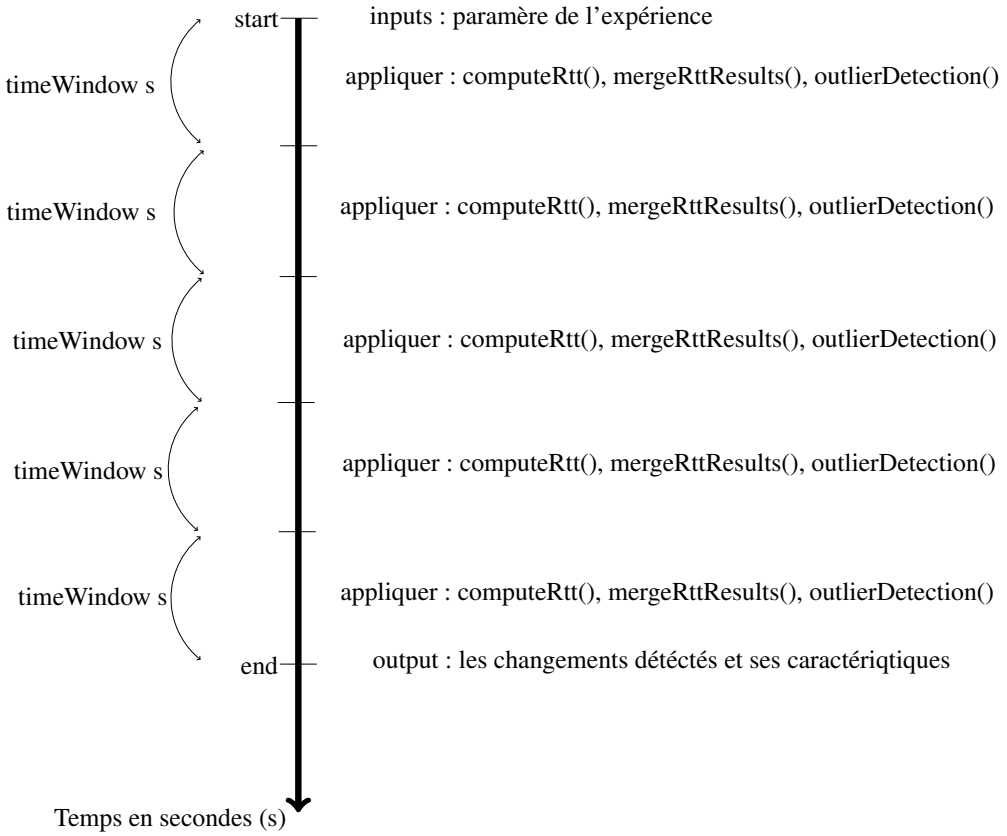


FIGURE II.2 – Illustration du paramètre timeWindow

alpha : c'est le paramètre du lissage exponentiel.

« *Les méthodes de lissage exponentiel sont un ensemble de techniques empiriques de prévision qui accordent plus ou moins d'importance aux valeurs du passé d'une série temporelle.* [4] »

Pour calculer la prochaine valeur de la médiane des RTTs différentiel de référence $\overline{m}_t$ du timeWindow courant t , soit :

m_t = $\Delta^{(m)}$ la médiane des RTTs différentiel observée pour un lien durant le timeWindow t.

$\overline{m}_{t-1}$ = $\overline{\Delta}^{(m)}$ est la médiane des RTTs différentiel de référence durant le timeWindow $t-1$, la prochaine valeur de la médiane de référence $\overline{m}_t$ est obtenue par :

$$\overline{m}_t = \alpha\ m_t + (1\text{- }\alpha)\ \overline{m}_{t-1}$$

Le paramètre alpha α, tel que $\alpha \in (0, 1)$, est le seul paramètre à définir dans le calcul de $\overline{m}_t$. Ce paramètre contrôle l'importance des mesures précédentes par rapport aux mesures récentes.

«*Plus α est proche de 1 plus les observations récentes influent sur la prévision, à l'inverse un α proche de 0 conduit à une prévision très stable prenant en compte un passé lointain.* [5]». Dans la présente étude, le paramètre α est préféré d'être petit.

minASN : les paramètres *minASN*, *minASNEntropy* ainsi que *minSeen* dont les deux derniers seront détaillés dans la suite sont associés à l'aspect diversité des sondes Atlas pour assurer une exactitude des résultats obtenus.

L'analyse des RTTs différentiel est appliquée seulement sous certaines conditions. Ainsi, la détection des anomalies dans les délais d'un lien est valide si les éléments suivants sont vrais. (1) Le lien est surveillé par plusieurs sondes et que le chemin de retours vers ces sondes soit différent à chaque fois. (2) Les paquets ayant passés par le lien XY, doivent aussi passer par le lien XY en leur retour, mais dans le sens opposé.

Les valeurs des RTTs ambiguës sont filtrées en éliminant les liens surveillés seulement par les sondes appartenant au même Système Autonome, car généralement le chemin de retour est similaire pour ces sondes suite à leur présence au sein du même Système Autonome (généralement même politique de routage). Seuls les liens surveillés par au moins 3 Systèmes Autonomes qui sont conservés, la valeur de 3 pour le paramètre *minASN* est choisie de manière empirique. Sachant qu'une valeur plus petite que 3 peut affecter l'exactitude des résultats.

4. Source : https://perso.math.univ-toulouse.fr/lagnoux/files/2013/12/Chap6.pdf, consultée le 30/09/2018.

5. Source : https://www.math.u-psud.fr/~goude/Materials/time_series/cours3_lissage_expo.pdf, consultée le 30/09/2018.

minASNEntropy : Il important qu'un lien soit identifié par au moins *minASN* Systèmes Autonomes, toutefois, le nombre de sondes Atlas ayant identifié ce lien doit être équilibré entre ces Systèmes Autonomes. En ce qui concerne l'équilibre du nombre de sondes ayant surveillé un lien par AS, il est mesuré par une entropie normalisée. Soit $A = \{a_i \mid i \in [1, n]\}$, a_i est le nombre de sondes pour chaque AS parmi les n ASs surveillant un lien donné. L'entropie $H(A)$ est défini avec :

$$H(A) = -\frac{1}{\ln} \sum_{i=1}^{n} P(a_i) \ln P(a_i)$$

Le nombre de sondes par Système Autonome sera équilibré jusqu'à l'atteinte de l'entropie minimale donnée par *minASNEntropy*. L'idée est d'éliminer des sondes Atlas, de manière aléatoire, de l'AS qui a un grand nombre de sondes Atlas jusqu'à avoir l'équilibre mesuré par l'entropie. Dans la présente analyse, les liens ayant une entropie > 0.5 sont conservés, cependant, si l'entropie d'un lien est < 0.5, chercher une sonde, de manière aléatoire, qui se trouve dans l'AS i tel que $a_i = max(A)$, ensuite recalculer l'entropie avec la sonde. L'opération de l'élimination est répétée jusqu'à avoir une entropie > 0.5.

L'entropie

L'entropie *est une grandeur d'état extensive qui caractérise l'état de désordre du système.*[a] De faible valeurs d'entropie,$H(A) \backsimeq 0$, indiquent que la majorité des sondes sont concentrées dans un seul AS, et les grandes valeurs d'entropie, $H(A) \backsimeq 1$, indiquent que les sondes sont réparties équitablement sur les ASs.

a. Source : `http://ressources.univ-lemans.fr/AccesLibre/UM/Pedago/chimie/01/03-Reaction_chimique/co/module_03-Reaction_chimique_26.html`, consultée le 30/09/2018.

minSeen : comme l'analyse est faite sur plusieurs périodes : timeWindow, le paramètre *minSeen* indique le nombre de fois où un lien a été identifié. Par exemple, un lien peut être identifié dans 3 timeWindow, ou bien être identifié en une seule fois durant toute la période de l'analyse.

af : ce paramètre indique l'étendue de l'analyse des délais en matière de type d'adressage. Telle qu'une analyse peut concentrer sur les liens en IPv4 ou bien en IPv6. Pour précision, il est pris en compte le type d'adressage IP lors du stockage

des traceroutes dans MongoDB [6]. Ce que permet de choisir les traceroutes suivant ce paramètre [7].

comment : un commentaire est utile pour donner plus d'informations sur une analyse en particulier. Les commentaires sont à titre informatifs et n'affectent pas l'analyse.

prefixes : les liens analysés sont finalement les liens entre deux routeurs adjacents dans la topologie. Avec l'expression régulière fournie à travers le paramètre *prefixes*, il est possible de limiter l'analyse sur les liens où les routeurs appartiennent aux blocs d'adresses définis par *prefixes*.

experimentDate : le paramètre *experimentDate* indique la date de lancement de l'expérience. Les auteurs enregistrent les expériences dans une collection dans la base de données MongoDB. Ce que permet de faciliter la comparaison entre les différentes expériences.

confInterval :

Afin de calculer l'incertitude associée à un ensemble de résultats, il faut répéter les mesures. Chaque mesure sur un échantillon peut donner des résultats différents. Ainsi, en se basant sur la déviation sur les résultats, il est possible de calculer l'incertitude de la "moyenne" calculée de ces résultats. Cette incertitude permet de donner une indication sur les données. Par exemple, est-ce que la moyenne calculée N représente la valeur réelle avec une incertitude de plus ou moins m ?

6. C'est une base de données NoSQL utilisée dans le stockage des données dans le travail de référence.

7. Dans MongoDB, la collection traceroute_2017_05_15 reprend les traceroutes de la date 15/05/2017 en IPv4, et la collection traceroute6_2017_05_15 pour la même date mais en IPv6.

En statistiques, le *binomial proportion confidence interval* est l'intervalle de confiance pour la probabilité de succès calculée à partir des séries d'expériences de succès-échec. C'est un intervalle qui estime la probabilité de succès p si seulement le nombre d'expériences n et celles réussites n_s sont connus.
Il existe plusieurs formules pour calculer l'intervalle de confiance binomial. Toutefois, elles se basent toutes sur une distribution binomiale. Une distribution binomial s'applique si une expérience est répétée un nombre fixe de fois, chaque tentative a deux possibilités : succès ou échec. La probabilité est la même à chaque tentative et les tentatives sont statistiquement indépendantes. La distribution binomiale est une distribution de probabilité discrète, il est difficile de calculer pour un grand nombre de tentatives, il existe une variété d'approximations pour le calcul de l'intervalle de confiance.

les intervalles de confiance sont formulés par un calcul binomial avec distribution - free . Ce calcul est approché par le score de Wilson. C'est une méthode pour calculer l'intervalle de confiance. Le score de Wilson fournit deux valeurs dans l'intervalle $[0, 1]$.

Intervalle de confiance d'une moyenne

L'intervalle de confiance (IC) est défini comme représentant les valeurs probables que peut prendre une moyenne, si l'on accepte une marge d'erreur définie à l'avance (e.g 5%). Il existe plusieurs méthodes pour calculer l'intervalle de confiance d'une proportion. Parmi les critères impliqués sur le choix de la méthode de calcul, il y a N, le nombre total d'essais (nombre des expériences).

Dans le travail de référence, les auteurs ont utilisé la méthode du score de wilson pour calculer l'intervalle de confiance de la médiane [8]. Le score de wilson a montré ses performances même dans le cas où le nombre total d'essais est petit. Par exemple, il se peut qu'un lien soit caractérisé par seulement 3 RTTs différentiel durant un timeWindow. [9]

Chaque valeur de médiane a son intervalle de confiance. On compare le chevauchement entre l'intervalle de confiance de la médiane de référence avec l'intervalle de confiance de la valeur de médiane calculée en cours, d'un lien donné. Afin d'évaluer si la différence entre ces deux intervalles est significative statistiquement. En particulier une différence de $1\ ms$ est non significative. On distingue trois cas comme illustré par la figure II.3 et la formule II.1.

8. Le choix d'utilisation de la médiane à la place de la moyenne a été justifié dans le travail de référence.

9. Source : `http://npsycog.over-blog.com/article-3274585.html`, consultée le 12/10/2018.

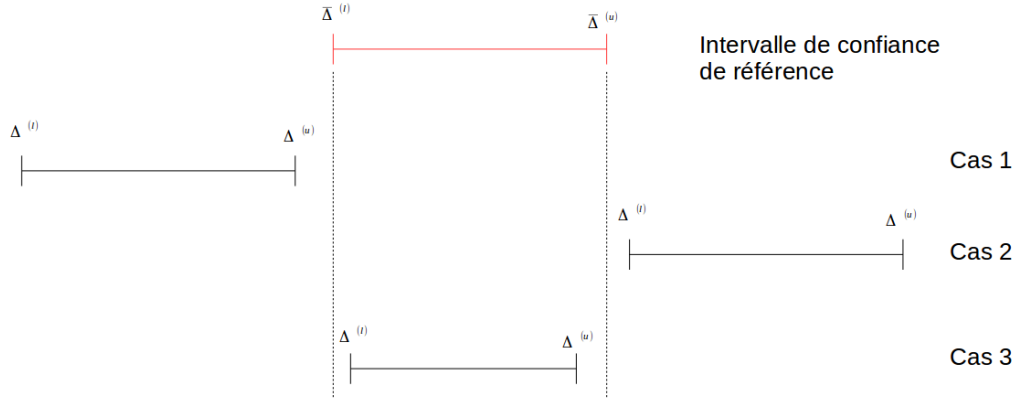


FIGURE II.3 – La comparaison entre les intervalles de confiance de la médiane de référence et de la médiane calculée.

Le cas 1 et 2 dans la figure II.3 illustrent un changement anormal dans le délai du lien en question. Toutefois, le cas 3 où l'intervalle de confiance courant est inclus dans l'intervalle de référence, est le cas normal, autrement dit, le délai de ce lien est normal. La différence entre les deux intervalles intervalles de confiance est quantifiée par la déviation d définie par la formule II.1.

$$d = \begin{cases} \frac{\Delta^{(l)} - \overline{\Delta}^{(u)}}{\overline{\Delta}^{(u)} - \overline{\Delta}^{(m)}}, & \text{if } \overline{\Delta}^{(u)} < \Delta^{(l)}. \\ \frac{\overline{\Delta}^{(l)} - \Delta^{(u)}}{\overline{\Delta}^{(u)} - \overline{\Delta}^{(m)}}, & \text{if } \overline{\Delta}^{(m)} < \Delta^{(l)}. \\ 0, & \text{otherwise.} \end{cases} \tag{II.1}$$

L'intervalle de confiance est calculé, dans la présente implémentation, avec la fonction *sm.stats.proportion_confint* [10].

```
statsmodels.stats.proportion.proportion_confint(count, nobs, alpha=0.05, method='normal')
```

proportion_confint est une implémentation, en python, pour le calcul du score de Wilson. Elle prend les paramètres suivants :

- count : c'est le nombre de fois où une expérience réussit.
- nobs : c'est le nombre total d'expériences.
- alpha : c'est le paramètre de risque. Généralement il prend les valeurs suivantes 5%, 1% ou $0,1\%$. C'est un des paramètres de l'expérience.
- method : on distingue plusieurs méthodes, celle utilisée est *wilson*.

10. Source : https://www.statsmodels.org/dev/generated/statsmodels.stats.proportion.proportion_confint.html, consultée le 07/10/2018.

En retour, la fonction *proportion_confint* fournie *ci_low* et *ci_upp*. Ces deux grandeurs sont utilisés pour construire l'intervalle de confiance.

Exemple des paramètres de l'analyse des délais

```
expParam = {
        "timeWindow": 60*60, # in seconds
        "start": datetime(2018, 1, 1, 0, 0, tzinfo=timezone("UTC")),
        "end":   datetime(2018, 1, 1, 23, 0, tzinfo=timezone("UTC")),
        "alpha": 0.01,
        "confInterval": 0.05,
        "minASN": 3,
        "minASNEntropy": 0.5,
        "minSeen": 3,
        "experimentDate": datetime.now(),
        "af": "",
        "comment": "some comment",
        "prefixes": None
}
```

Les étapes de l'analyse des délais

Les sections suivantes présentent chaque étape brièvement des étapes présentées dans l'algorithme 1, c'est pour l'illustration, car des détails peuvent manquer comme les paramètres des fonctions.

computeRtt :
Objectif : identification de tout lien dans les traceroutes analysés.
Entrées : af, start, end, prefixes.
Sorties : la caractérisation des liens en calculant leur RTT différentiel.
Pseudo-code :

Algorithm 2 caractérisation des liens

```
function COMPUTERTT(af, start, end, prefixes)
    diffRtt ← {}
    nbRow ← 0
    traceroutes ← findTraceroutes(af, start, end, prefixes)
    for all trace ∈ traceroutes do
        readOneTraceroute( trace )
    end for
     return diffRtt, nbRow
end function
```

Description :

1. Rechercher les traceroutes, dans la base de données, capturés entre la date *start* et *end*, parmi les traceroutes ayant *af* comme adressage. Si *prefixes* est fixé, seuls les traceroutes ayant impliqué des routeurs appartenant au bloc d'adresses défini par *prefixes* qui seront conservés.

2. Analyser chaque traceroute, *trace*, à travers les étapes suivantes :

   (a) élimination du traceroute échoué en vérifiant la présence des attributs "error" ou "err" dans les résultats de ce traceroute ;

   (b) si le traceroute n'a pas été éliminé durant l'étape précédente, on passe à l'évaluation de chaque saut. Un saut est éliminé si l'adresse IP est absente pour ce saut, s'il s'agit d'une adresse privée, si l'information sur le RTT est absente et enfin si le RTT a une valeur négative ;

   (c) caractérisation des liens en notons pour chaque couple d'adresses IP la distribution des RTTs, les sondes Atlas et les identifiants des mesures.

Soit un exemple illustratif des opérations décrites ci-dessus, la figure II.4 reprend les détails. Tel que :

- Lien : lien à suivre, défini par deux adresses IP.
- RTT différentiel : le RTT différentiel calculé du lien en question.
- Probe : l'ensemble de sondes ayant surveillé le lien.
- msmId : l'ensemble de mesures ayant surveillé le lien. Comme un identifiant de mesure définit la requête à lancer par toutes les sondes Atlas, ainsi, un lien peut être surveillé dans le cadre d'une mesure et avec une ou plusieurs sondes Atlas.

Pour le lien $('160.242.100.88', '196.216.48.144')$, le calcul du RTT différentiel est décrit ci-dessous :

$$
\begin{aligned}
RTT(160.242.100.88) &= 4.263; 6.082; 11.834 \\
mediane(4.263; 6.082; 11.834) &= 6.082 \\
RTT(160.242.100.88) &= 3.678; 15.568; 3.655 \\
mediane(3.678; 15.568; 3.655) &= 3.678 \\
RTTDiff('160.242.100.88', '196.216.48.144') &= 6.082 - 3.678 = 2,404
\end{aligned}
$$

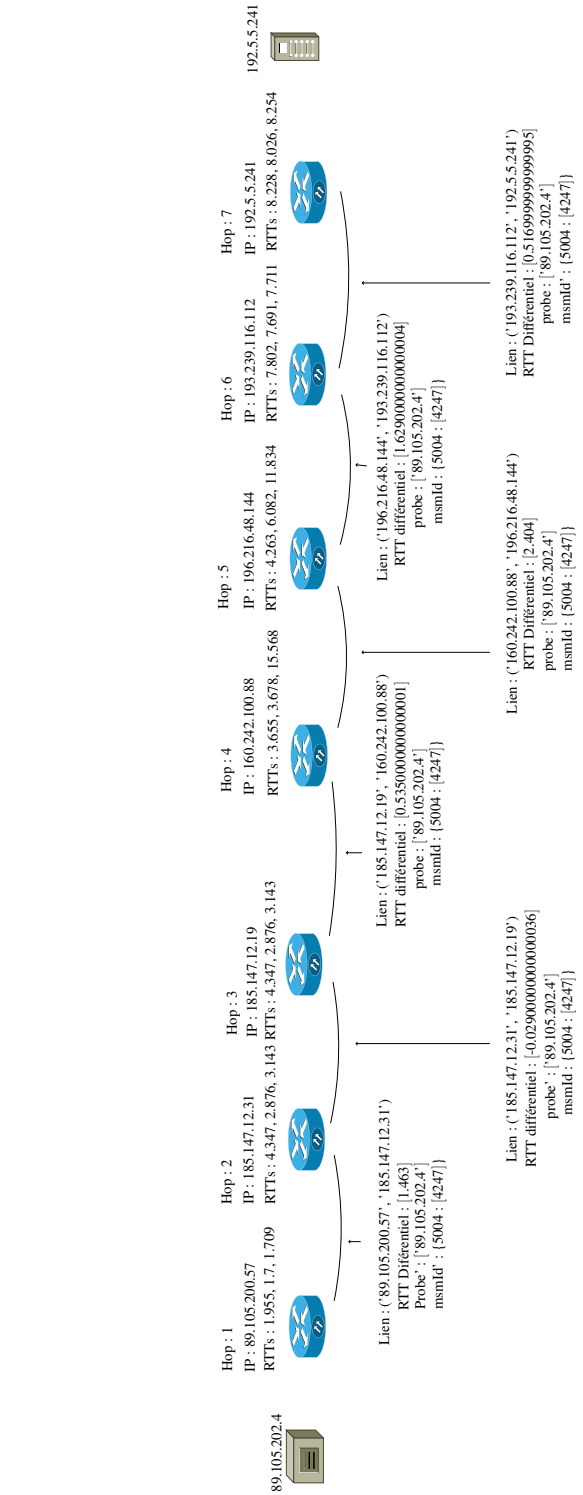


FIGURE II.4 – Caractérisation des liens dans un traceroute

mergeRttResults

Durant l'analyse des traceroutes identifiés dans un timeWindow, il se peut qu'un lien soit identifié dans plusieurs traceroutes. La méthode *mergeRttResults* permet de fusionner les détails d'un lien donné, ce sont des détails obtenus avec plus d'un traceroute. Prenons le lien $(\prime 160.242.100.88\prime, \prime 196.216.48.144\prime)$, ce dernier a été identifié dans trois traceroutes comme suit :

(IP1,IP2)	RTT	Probes	MSM
('160.242.100.88', '196.216.48.144')	[63.474000000000004]	[u'196.216.164.50']	5004 : set([14465])
('196.49.6.10', '192.5.5.241')	[3.4729999999999848]	['196.216.164.50']	5004 : set([14465])
('196.216.164.1', '196.12.10.246')	[-1.379]	['196.216.164.50']	5004 : set([14465])
('196.12.10.246', '160.242.100.88')	[0.21799999999999997]	['196.216.164.50']	5004 : set([14465])

TABLE II.2 – Exemple de traceroute : 1

(IP1,IP2)	RTT	Probes	MSM
('185.147.12.31', '185.147.12.19')	[-0.02900000000000036]	[u'89.105.202.4']	5004 : set([4247])
('185.147.12.19', '160.242.100.88')	[0.5350000000000001]	['89.105.202.4']	5004 : set([4247])
('89.105.200.57', '185.147.12.31')	[1.463]	['89.105.202.4']	5004 : set([4247])
('196.216.48.144', '193.239.116.112')	[1.6290000000000004]	['89.105.202.4']	5004 : set([4247])
('193.239.116.112', '192.5.5.241')	[0.5169999999999995]	['89.105.202.4']	5004 : set([4247])
('160.242.100.88', '196.216.48.144')	[2.404]	['89.105.202.4']	5004 : set([4247])

TABLE II.3 – Exemple de traceroute : 2

(IP1,IP2)	RTT	Probes	MSM
('199.189.117.17', '199.189.116.229')	[-0.02400000000000091]	['134.197.113.7']	5004 : set([6201])
('160.242.100.88', '207.197.17.101')	[19.261999999999997]	['134.197.113.7']	5004 : set([6201])
('199.189.116.229', '63.158.61.169')	[2.34]	[u'134.197.113.7']	5004 : set([6201])
('205.171.1.18', '192.5.5.241')	[-0.18400000000000105]	['134.197.113.7']	5004 : set([6201])
('63.158.61.169', '205.171.234.102')	[-2.0219999999999985]	['134.197.113.7']	5004 : set([6201])
('134.197.113.14', '196.216.48.144')	[-0.5369999999999999]	['134.197.113.7']	5004 : set([6201])
('207.197.17.101', '199.189.117.17')	[2.1290000000000013]	['134.197.113.7']	5004 : set([6201])
('205.171.234.102', '205.171.1.18')	[-0.17099999999999937]	['134.197.113.7']	5004 : set([6201])
('196.216.48.144', '160.242.100.88')	[0.065]	[u'134.197.113.7']	5004 : set([6201])

TABLE II.4 – Exemple de traceroute : 3

Afin d'illustrer l'opération de fusion, on reprend seulement les détails du lien qui a subi la fusion en se référant aux résultats intermédiaires présentés dans les tableaux II.2, II.3 et II.4. En effet, on obtient :

(IP1,IP2)	RTT	Probes	MSM
('160.242.100.88', '196.216.48.144') :	[0.065, 2.404, 63.474000000000004]	['134.197.113.7', '89.105.202.4', '196.216.164.50']	5004 : set([6201, 4247, 14465])

Ce qu'on peut apprendre à travers cette fusion, le délai d'un lien ne dépend pas de l'ordre des routeurs.

outlierDetection

Dans le présent contexte, la détection des *outliers* dénote la détection des changements anormaux. Cette opération implique plusieurs notions et principes. Les étapes de la détection des outliers sont illustrées dans l'algorithme 3.

Objectif : trouver les changements anormaux des délais après la caractérisation des liens.

Entrées : diffRTT, le résultat de l'étape *mergeRttResults*, la liste des éléments par lien comme :

```
{
   (u'160.242.100.88', u'196.216.48.144'): {'rtt': [0.065, 2.404, 63.474000000000004],
   'probe': [u'134.197.113.7', u'89.105.202.4', u'196.216.164.50'],
   'msmId': defaultdict(<type 'set'>, {5004: set([6201, 4247, 14465])})}
}
```

Sorties : la liste des alarmes.

Pseudo-code :

Algorithm 3 La détection des changements anormaux des liens

```
1 : function OUTLIERDETECTION(diffRTT, smoothMean)
2 :     smoothMean ← {}                          ▷ La médiane de référence d'un lien
3 :     alarms ← []                              ▷ La liste des anomalies détectées
4 :     alpha ← float(param["alpha"])
5 :     minAsn ← param["minASN"]
6 :     minASNEntropy ← param["minASNEntropy"]
7 :     confInterval ← param["confInterval"]
8 :     minSeen ← param["minSeen"]
9 :     for all ipPair ∈ sampleDistributions do
10 :        dist ← la distribution des RTTs différentiel précédemment calculée.
11 :        probes ← les sondes ayant surveillé le lien ipPair
12 :        asn ← probe2asn(probes)
13 :        asnEntropy ← computeAsnEntropy()
14 :        while asnEntropy < minASNEntropy AND len(asn) > minAsn do
15 :            trimDistribution()
16 :        end while
17 :        findAlarms(smoothMean)
18 :    end for
        return diffRtt, nbRow
19 : end function
```

Description :

– probe2asn(probes) : cette étape permet de faire l'association entre l'adresse IP de la sonde Atlas et son ASN. Cette association permet d'avoir une idée

sur le nombre de sondes Atlas ayant surveillé le lien par Système Autonome. Ce que explique la section II.3.4.

- trimDistribution() : l'objectif de cette étape est d'équilibrer le nombre de sondes Atlas par AS ayant surveillé un lien en vue d'éviter les résultats biaisées.

- computeAsnEntropy() : calcul de l'entropie d'une distribution pour des valeurs de probabilité données. *stats.entropy* est l'implémentation, en python, du calcul de l'entropie. $stats.entropy(asn.values())/np.log(len(asn))$.

- findAlarms() : dans cette étape, on discute la procédure d'identification des anomalies, autrement dit, les alarmes. Pour une raison de clarté et de lisibilité, cette étape est détaillée avec l'algorithme 4.

Algorithm 4 caractérisation des liens

```
1 : nbProbes ← len(probes)
2 : n ← len(dist)
3 : med ← np.median(dist)
4 : wilsonCi ← sm.stats.proportion_confint(len(dist)/2, len(dist), confInterval,
    "wilson")
5 : currLow ← dist[int(wilsonCi[0])]
6 : currHi ← dist[int(wilsonCi[1])]
7 : reported ← False
8 : if ipPair in smoothMean then ▷ mise à jour de la référence d'un lien identifié
9 :     if ref["nbSeen"] >= minSeen then
10 :        if ref["high"] < currLow or ref["low"] > currHi then
11 :            if med < ref["mean"] then
12 :                update diff, diffMed, deviation, devBound
13 :            else
14 :                update diff, diffMed, deviation, devBound
15 :            end if
16 :            alarm ← describeAlarm()
17 :            reported ← True
18 :        end if
19 :    end if
20 :    updateReference()
21 : else                                   ▷ nouveau lien donc nouvelle référence
22 :    if minSeen > 1 then
23 :        smoothMean[ipPair] ← updateReference()
24 :    else
25 :        smoothMean[ipPair] ← updateReference()
26 :    end if
27 : end if
```

Les deux valeurs de l'intervalle de confiance calculé avec le score de Wilson sont illustrées dans les deux lignes 6 et 5 de l'algorithme 4.

Le principe de la détection des anomalies est repris dans la section II.3.5, c'est le même principe de la détection, toutefois les étapes sont adaptées pour générer les résultats.

II.3.5 Présentation de l'algorithme de la détection des changements anormaux : Résultat II

Introduction

Cette analyse permet le suivi des RTTs différentiel d'un lien donné, entre la date *start* et la date *end*. Ce suivi se base sur les dernières valeurs du RTT différentiel. Rappelons que chaque lien est caractérisé par son RTT différentiel qui représente la médiane de tous les RTTs différentiel enregistrés pendant une durée d. Cette médiane a un intervalle de confiance. Afin de pouvoir répondre à la question suivante : à 95 %, la médiane de référence calculée, *smoothAvg*, ayant comme intervalle de confiance *smoothHi* et *smoothLow* représente l'estimation réelle du RTT différentiel du lien en question.

L'idée de base de l'outil de détection est d'analyser les données, qui sont des traceroutes, d'une période D en la décomposant en de petites périodes :

$d_1, d_2, ..., d_i, ..., d_j, ..., d_n$, tel que *timeWindow* = d_{j+1} - d_j, *timeWindow* est exprimé en secondes, d_1 représente *start* et d_n représente *end*.

Caractéristiques d'un lien

A l'issue d'une préparation préalable des traceroutes, chaque lien est caractérisé par les dates pendant lesquelles il était identifié ainsi que les RTTs différentiel relatives à ces dates. C'est ce que illustre *rttDiff*.

rttDiff = $([d_1, d_2, ..., d_i, d_j, ..., d_n], [r_1, r_2, ..., r_n])$, $i, j \in [0, n]$, n est entier.

où d_i est la date marquant le début d'un timeWindow et r_i est le RTT différentiel enregistré durant d_i, n est le nombre de fois où e a été identifié. $\exists i, j$ tel que $d_i = d_j$, c'est le cas où un lien a été identifié plusieurs fois durant un timeWindow, et comme on présente l'évolution par timeWindow, les RTTs différentiel seront combinés par la suite en calculant la médiane par ce timeWindow.

```
rttDiff[e] = (dates, rttDiffs)
dates : [datetime.datetime(2018, 1, 2, 1, 5), datetime.datetime(2018, 1, 2, 1,5), datetime.datetime(2018,
    1, 2, 2, 5)]
rttDiffs : [1.463, 4.394, 358.394]
```

On caractérise un lien e durant la période d_j avec les éléments suivants :

- *dest[]* : c'est la distribution des RTTs différentiel du lien e durant d_i. Une distribution de moins de 3 RTTs différentiel pour d_i est à ne pas considérer.
- *smoothAvg[]* : c'est l'estimation de la médiane de référence.
- *smoothHi[]* : c'est l'estimation de la médiane minimale de référence. C'est la borne supérieure de l'intervalle de confiance de référence.
- *smoothLow[]* : c'est l'estimation de la médiane maximale de référence. C'est la borne inférieure de l'intervalle de confiance de référence
- *alarmsDates[]* : les dates pendant lesquelles une anomalie a été détectée.
- *alarmsValues[]* : les médianes des RTTs différentiel considérées comme anormales.

- *median[]* : les médianes des RTTs différentiel calculées caractérisant le lien tout au long de la période de l'analyse, une médiane par d_j.
- *mean[]* : la moyenne des RTTs différentiel. L'utilisation de la moyenne a pour but la comparaison des performances de cette dernière avec celles de la médiane. Alors que l'outil de détection repose principalement sur la médiane.
- *ciLow[]* : la borne inférieure de l'intervalle de confiance de la médiane calculée.
- *ciHigh[]* : la borne supérieure de l'intervalle de confiance de la médiane calculée.
- *dates[]* : les dates à présenter dans le graphique. Certaines dates ne sont pas présentables si le nombre des RTTs différentiel est petit durant ces dates.

La procédure de création de l'évolution des RTTs différentiel d'un lien

Objectif : Analyser l'évolution du RTT différentiel d'un lien donné, noté e.

Entrées : les détails du lien, c'est une instance du rttDiff.

Sorties : l'évolution des RTTs différentiel du lien avec la présentation des anomalies si elles étaient identifiées.

Etapes : d'abord on distingue deux étapes principales, la première a pour objectif la préparation de données. Ainsi, pour chaque lien trouvé, on le caractérise avec leurs RTTs différentiels ainsi que la date pendant laquelle un RTT différentiel a été enregistré. Deuxièmement, en partant des résultats de la première étape, on génère l'évolution du RTT d'un lien. La deuxième étape est décrite avec l'algorithme 5 dont on peut noter les étapes de cette dernière :

1. Préparation des données du timeWindow, entre 2 et 14.
   - calcul du score de wilson
   - calcul de la médiane et de son intervalle de confiance.
2. Mise à jour des données du lien en calculant la médiane de référence et l'intervalle de confiance de la médiane de référence :

- Tant que le nombre des médianes est inférieure strictement à 24, on met à jour les distributions smoothAvg, smoothHi et smoothLow.
- Si le nombre de médianes est égale à 24, on construit une sorte de médiane de référence (25 ème), c'est la médiane des médianes précédemment notées pour ce lien, ensuite les 24 premières médianes seront mises à jour avec cette médiane de référence.
- A partir d'un nombre de médianes plus grand que 24, la nouvelle médiane est prédite à travers la formule du lissage exponentiel.

3. Détection des anomalies en comparant les deux intervalles de confiance : le courant avec celui de référence.
4. Génération de l'évolution : après avoir calculé pour chaque timeWindow de la période l'analyse :
   - la médiane des RTTs différentiel de référence ;
   - l'intervalle de confiance de la médiane de référence ;
   - la médiane des RTTs différentiel estimée ;
   - l'intervalle de confiance de la médiane estimée.

   La comparaison des deux intervalles de confiance permet d'affirmer si la médiane estimée est une anomalie.

Précisions relatives au pseudo-code :

- a.append(b) : ajouter b à la fin de la série de valeurs a.
- median(a) : calculer la médiane de la série de valeurs a.
- a.sort() : ordonner les valeurs de la série de valeurs a, par défaut, par ordre croissant.
- size(a) : donner la taille de la série de valeurs a.
- proportion_confint() : calcul de la borne inférieure et supérieur du score de Wilson.
- *dateranges* : la succession des dates marquant la durée de l'analyse.

Pseudo Code :

Algorithm 5

```
1 : for all d in dateranges do
2 :     indices ⇐ rttDiff[1]==d        ▷ Trouver les indices des RTTs différentiel identifiés durant d
3 :     dist ⇐ rttDiffs[indices]    ▷ Récupérer les RTTs Différentiel aux indices
4 :     if size(dist) < 3 then
5 :         passer à la date suivante dans dateranges
6 :     end if
7 :     dates.append(d)
8 :     median.append(median(dist))                ▷ mettre à jour la médiane
9 :     mean.append(mean(dist))
10 :    dist.sort()
11 :    wilsonCiLow, wilsonCiHi Leftarrow proportion_confint()  ▷ Calcul de score de Wilson
12 :    wilsonCi ⇐ np.array(wilsonCi)*len(dist)
13 :    currLow.append(median[-1] - dist[int(wilsonCi[0])])    ▷ mettre à jour la médiane minimale
14 :    ciHigh.append( dist[int(wilsonCi[1])] - median[-1] )    ▷ mettre à jour la médiane maximale
15 :    if len(smoothAvg) < 24 then
16 :        smoothAvg.append(median[-1])
17 :        smoothHi.append(dist[int(wilsonCi[1])])
18 :        smoothLow.append(dist[int(wilsonCi[0])])
19 :    else if len(smoothAvg) == 24 then
20 :        smoothAvg.append(np.median(smoothAvg))
21 :        smoothHi.append(np.median(smoothHi))
22 :        smoothLow.append(np.median(smoothLow))
23 :        for i in [0,24] do
24 :            smoothAvg[i] = smoothAvg[-1]
25 :            smoothHi[i] = smoothHi[-1]
26 :            smoothLow[i] = smoothLow[-1]
27 :        end for
28 :    else        ▷ Utilisation du lissage exponentiel pour prédire smoothAvg, smoothHi et smoothLow
29 :        smoothAvg.append(0.99*smoothAvg[-1]+0.01*median[-1])
30 :        smoothHi.append(0.99*smoothHi[-1]+0.01*dist[int(wilsonCi[1])])
31 :        smoothLow.append(0.99*smoothLow[-
    1]+0.01*dist[int(wilsonCi[0])])
            ▷ La détection des anomalies se déclenche dés avoir un échantillon de taille 25
32 :        if (median[-1]-ciLow[-1] > smoothHi[-1] OR median[-1]+ciHigh[-1]
    < smoothLow[-1]) AND np.abs(median[-1]-smoothAvg[-1])>1 then
33 :            alarmsDates.append(d)
34 :            alarmsValues.append(median[-1])
35 :        end if
36 :    end if
37 :
38 : end for
```

II.3.6 Quelques chiffres sur la médiane des RTTs différentiel

L'adaptation du théorème centrale limite pour l'utilisation de la médiane au lien de la moyenne peut engendrer des contraintes en matière de performances si la distribution pour laquelle on souhaite calculer la médiane est grande.

La figure II.5 présente les trente premières distributions des médianes. L'axe des abscisses représente les liens et l'axe des ordonnées représente la taille de la distribution des médianes des RTT différentiel d'un lien, ce sont les RTTs différentiel caractérisant le lien pendant lors un timeWindow parmi les timeWindows entre *start* et *end*. Pour une raison de clarté, le reste des distributions est présenté dans le document disponible sur GitHub [11].

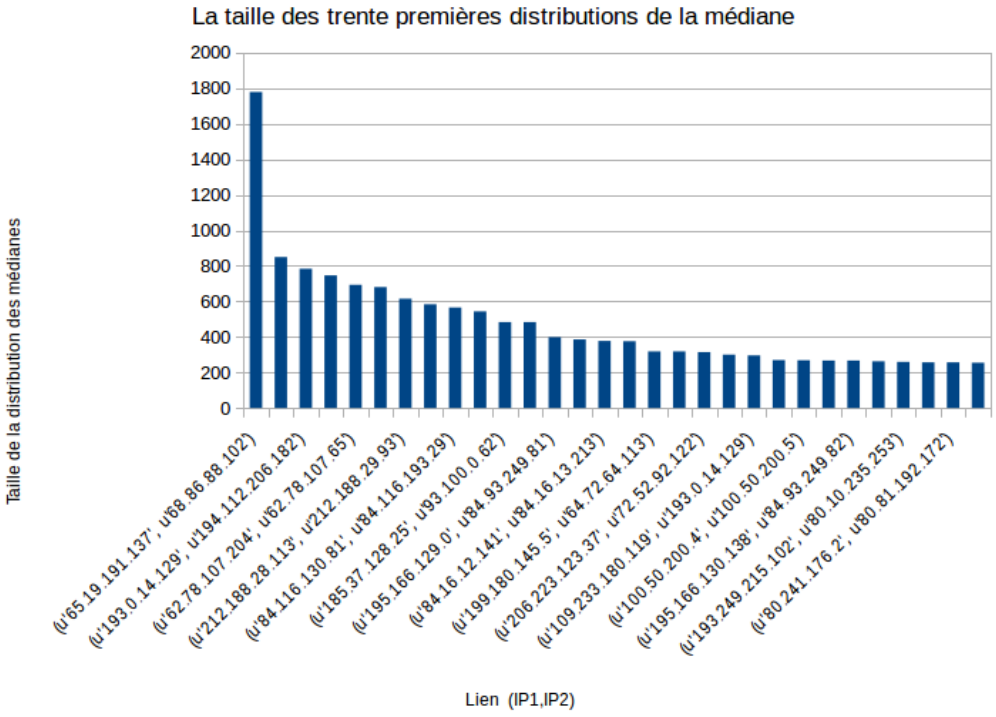


FIGURE II.5 – La taille de la distribution des médianes des RTT différentiel

II.3.7 Notes sur les traceroutes

L'analyse menée sur les traceroutes collectées par les sondes Atlas n'utilise pas totalement les traceroutes sélectionnés suivant quand ils ont été capturés.

11. `RipeAtlasTraceroutesAnalysis/tableaux/taille_distributions_medianes.pdf`, consulté le 14/10/2018.

Traceroute réussi ou échoué totalement. Dans le travail de référence, les traceroutes sont stockés tels qu'ils sont, sans prétraitement à l'avance. Or un traceroute peut être éliminé à l'avance si ce dernier a échoué pour atteindre la destination prédéfinie.

Traceroute réussi ou échoué partiellement. Dans certains cas, la sonde Atlas ne parvient pas à atteindre un routeur intermédiaire durant son chemin vers la destination finale. Rappelons que pour un saut, l'implémentation du traceroute utilisée par les sondes Atlas envoie 3 signaux par saut, ainsi, on distingue deux cas :

- la sonde ne reçoit aucune information sur les trois signaux ;
- la sonde reçoit les informations de 1 ou 2 signaux.

Les adresses IP privées dans traceroute. Dans la présente analyse, les adresses IP privées n'ont aucune valeur ajoutée. Les liens à surveiller sont ceux visibles sur Internet, alors que les adresses IP privées reflètent la configuration des réseaux non publique.

L'utilité des attributs d'un enregistrement traceroute. Une requête traceroute est caractérisée par plusieurs attributs, plus de 40, décrits dans la figure II.6. Les nœuds en gris sont de type liste, un élément de la liste est un objet formé les successeurs du nœud en question. Les nœuds en vert sont les attributs utilisés par l'outil de détection des changements anormaux dans les délais d'un lien.

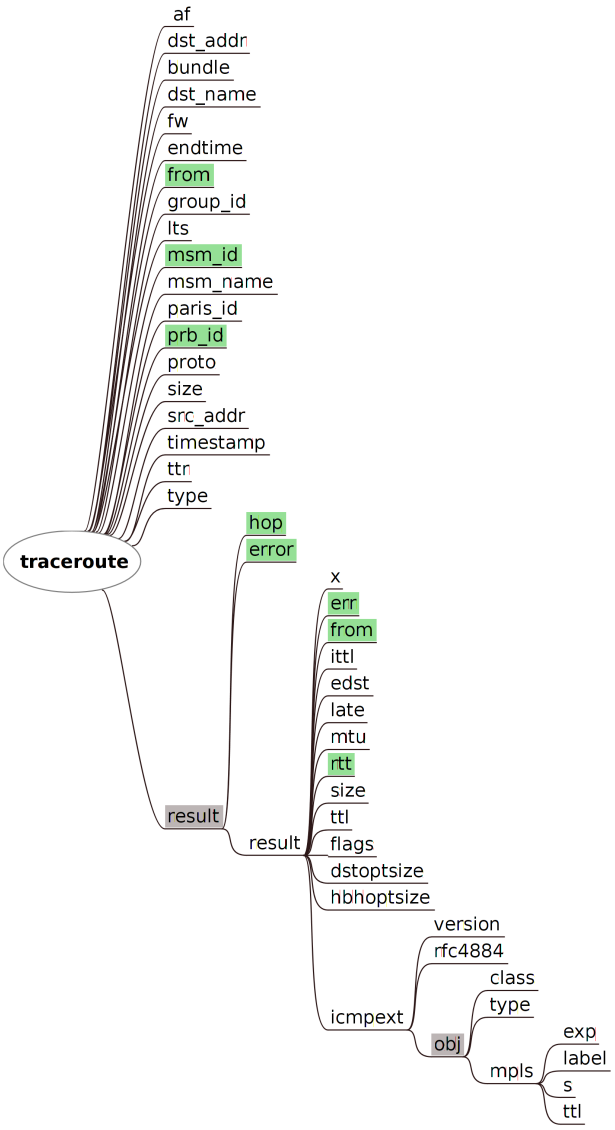


FIGURE II.6 – Les attributs possibles dans le résultat d'une requête traceroute

II.4 En pratique ...

Le travail de référence

Le processus de l'analyse se fait en récupérant d'abord les enregistrements des traceroutes depuis la base de données MongoDB. Ensuite, le traitement de chaque traceroute se poursuit dans la machine locale. Les résultats de type I, qui sont les changements des délais de tous les liens identifiés, sont stockés localement dans la machine locale. De plus, les détails de l'expérience sont aussi stockées dans MongoDB. Pour conclure toutes les opérations se déroulent dans la machine locale comme c'est illustré dans la figure II.7.

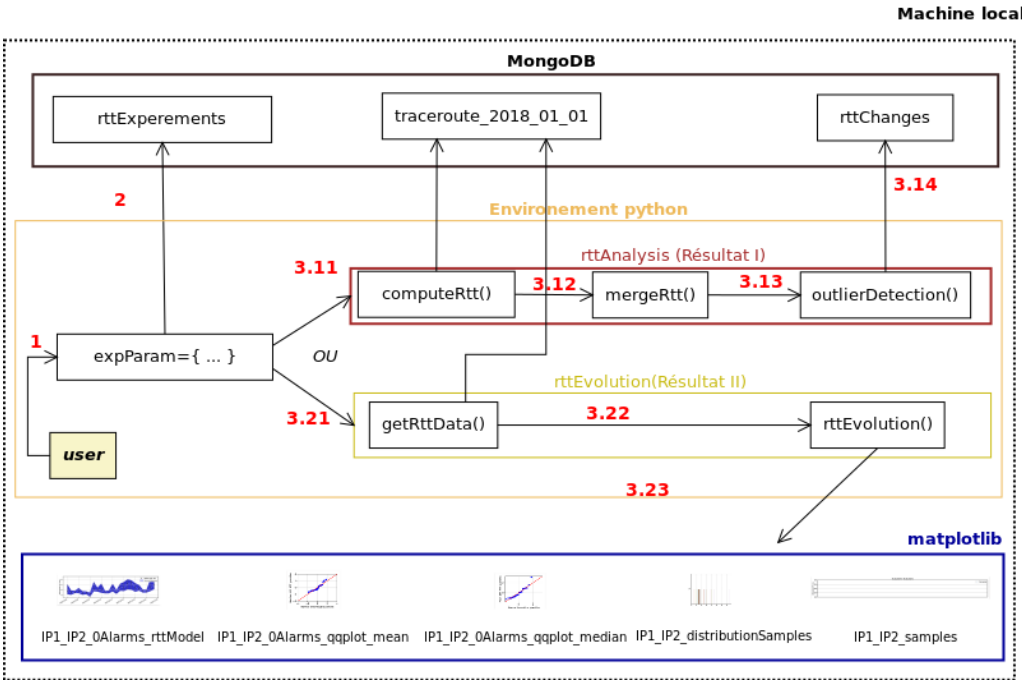


FIGURE II.7 – Le processus d'analyse des traceroutes dans le travail de référence (MongoD)

Intégration des AWS Athena + S3 dans le travail de référence

La première adaptation du travail de référence en vue d'intégrer les services web d'Amazon est présentée dans la figure II.8. La différence par rapport à l'implémentation proposée dans le travail de référence est au niveau du stockage des données. En effet, au lieu de récupérer les traceroutes depuis les collections présentes dans MongoDB (traceroute_2018_01_01), les traceroutes sont récupérés depuis le service de stockage Amazon S3. Ces données ont été adaptées pour être utilisées par l'outil de la détection.

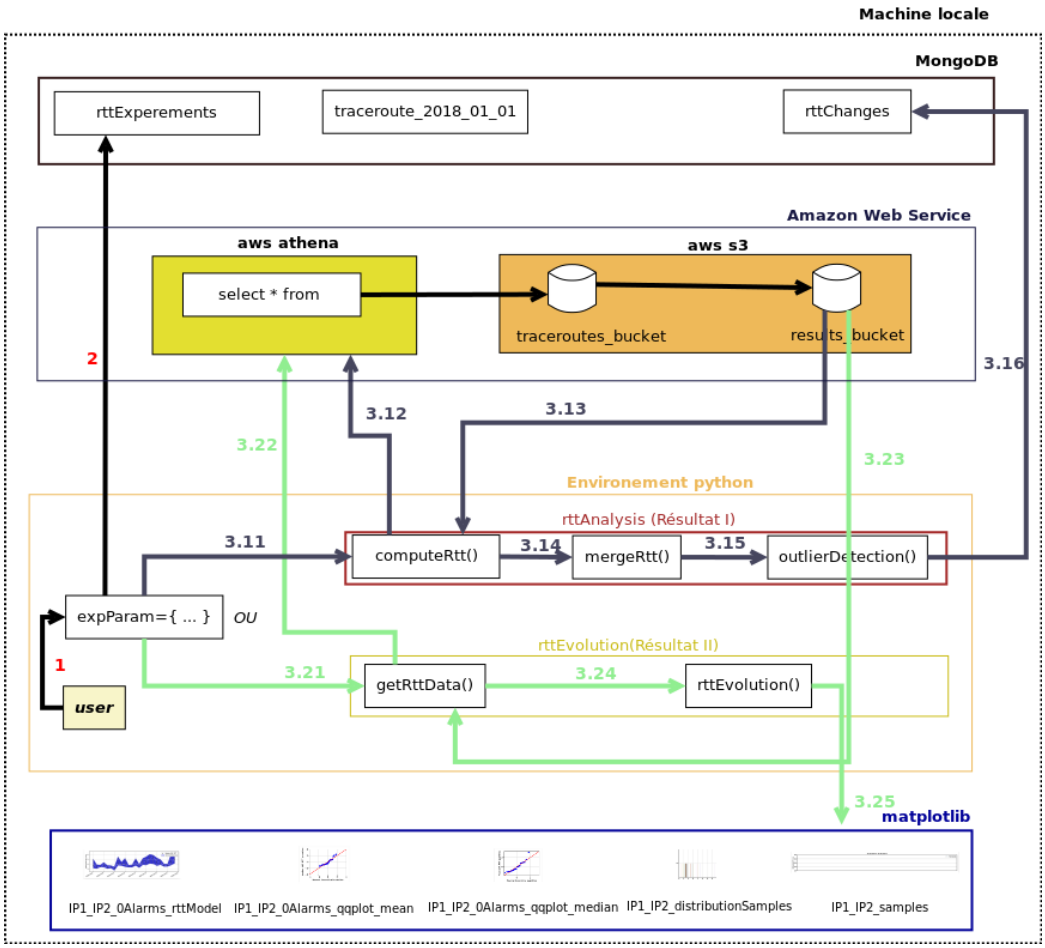


FIGURE II.8 – Intégration des AWS Athena + S3 dans le travail de référence

Aller au delà du stockage sur Amazon S3

Les données manipulées par l'outil de la détection sont volumineuses, en plus du besoin du stockage, il faut aussi adopter les outils adéquats pour le traitement de ces données, ce qui n'était pas pris en considération dans l'adaptation décrite dans la section II.4.

Et s'il existe une implémentation prenant en considération la manipulation des données massives en plus du stockage de ces dernières ? On rappelle qu'une détection se déroule en récupérant d'abords les traceroutes de la période souhaitée, ensuite, chaque traceroute est analysé pour identifier les liens avec leur RTT différentiel (computeRTT()).Ces résultats sont fusionnés avec mergeRTT(). Enfin, la détection des anomalies se déclenche (outlierDetection()). Afin d'évaluer cette possibilité, on présente dans la figure II.9 l'organigramme de la première étape dans la détection les changements des délais : computeRtt().

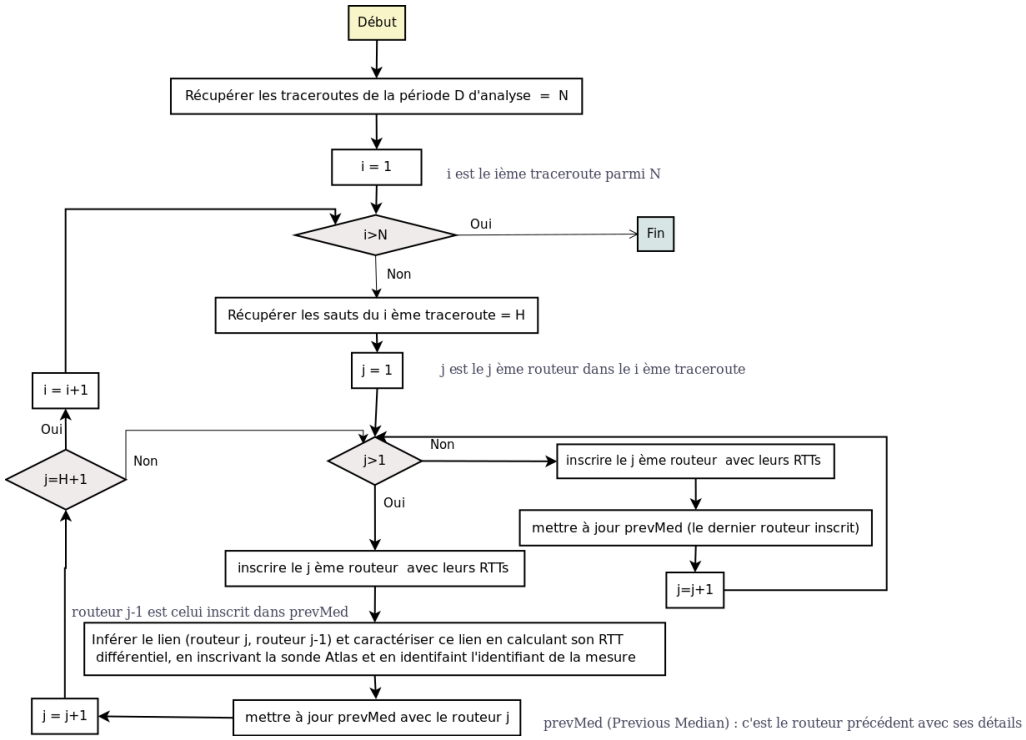


FIGURE II.9 – L'organigramme de l'étape computeRTT()

Soient les deux cas suivants :

cas 1 Supposant qu'il n'existe pas une requête SQL valide sur AWS Athena pour traiter tous les traceroutes en une seule fois, dans ce cas, on doit créer une requête par traceroute. De plus, on doit créer un fichier résultat qui stocke les résultats à passer à l'étape mergeRTT(). En pratique, une heure comme période d'analyse a repris environ 18400 traceroutes. En effet, pour une heure, il faut lancer 18400 requêtes sur AWS Athena et 18400 opérations de lecture/écriture sur un fichier résultat si on souhaite le stocker dans AWS S3. Alors pour une semaine d'analyse ?

cas 2 Jusqu'à maintenant, avec la requête suivante [12] :

12. A faire évoluer si les services web d'Amazon qui seront utilisés dans la suite du mémoire.

```
with dataset AS
    (SELECT de,
         msm_id,
         prb_id,
         mydata.hop,
         hopDetails."from",
         hopDetails.rtt
    FROM
        (SELECT "from" AS de,
         msm_id,
         prb_id ,
        mydata
        FROM traceroutes
        CROSS JOIN unnest(result) AS t(mydata)) AS tab
        CROSS JOIN unnest(mydata.result) AS tt(hopDetails))
    SELECT cast(approx_percentile(rtt,0.5) AS double) AS med ,
         "from",
         cast(hop AS integer) AS hop
FROM dataset
GROUP BY  "from", hop
```

FIGURE II.10 – Requête AWS Athena intermédiaire concernant le cas 2

Nous avons comme résultat :

Résultats

	med	from	hop
1			5
2	24.05699920654297	63.158.61.169	7
3	0.5440000295639038	196.12.10.246	2
4	182.28799438476562	196.49.6.10	6
5	3.1429998874664307	185.147.12.19	3
6	1.9229999780654907	196.216.164.1	1
7	21.86400032043457	205.171.1.18	9
8	3.171999931335449	185.147.12.31	2
9	8.253999710083008	192.5.5.241	7
10	6.081999778747559	196.216.48.144	5
11	0.6980000138282776	160.242.100.88	3
12	7.710999965667725	193.239.116.112	6
13	64.23600006103516	196.216.48.144	4
14	3.677999973297119	160.242.100.88	4
15	22.03499984741211	205.171.234.102	8

FIGURE II.11 – Résultats de la requête AWS Athena intermédiaire concernant le cas 2

Un lien est formé par l'adresse IP ayant avec *hop* = i+1 et l'adresse IP ayant *hop* = i. Le RTT différentiel est calculé en faisant la différence entre *med* avec hop

= i+1 et med avec hop = i, avec i entier et doit concerner le même traceroute. Les résultats présentés dans II.11 ne différencient pas les sauts du même traceroute.

Supposons qu'il existe une seule requête SQL valide sur AWS Athena et capable d'assurer toutes les opérations décrites dans l'organigramme II.9 (y inclus l'inférence des liens), dans ce cas, il faut stocker les résultats intermédiaires pour qu'ils soient l'entrée de l'étape mergeRTT(). Et supposons aussi qu'il existe une requête SQL sur Athena capable de lire les résultats en question et d'appliquer la fusion entre les liens. Evaluons maintenant l'étape *outlierDetection()* [13].

On peut présenter l'étape *outlierDetection()* brièvement via l'organigramme II.12 :

13. Le principe de la détection est le même pour outlierDetection (résultat I) et rttEvolution (Résultats II)

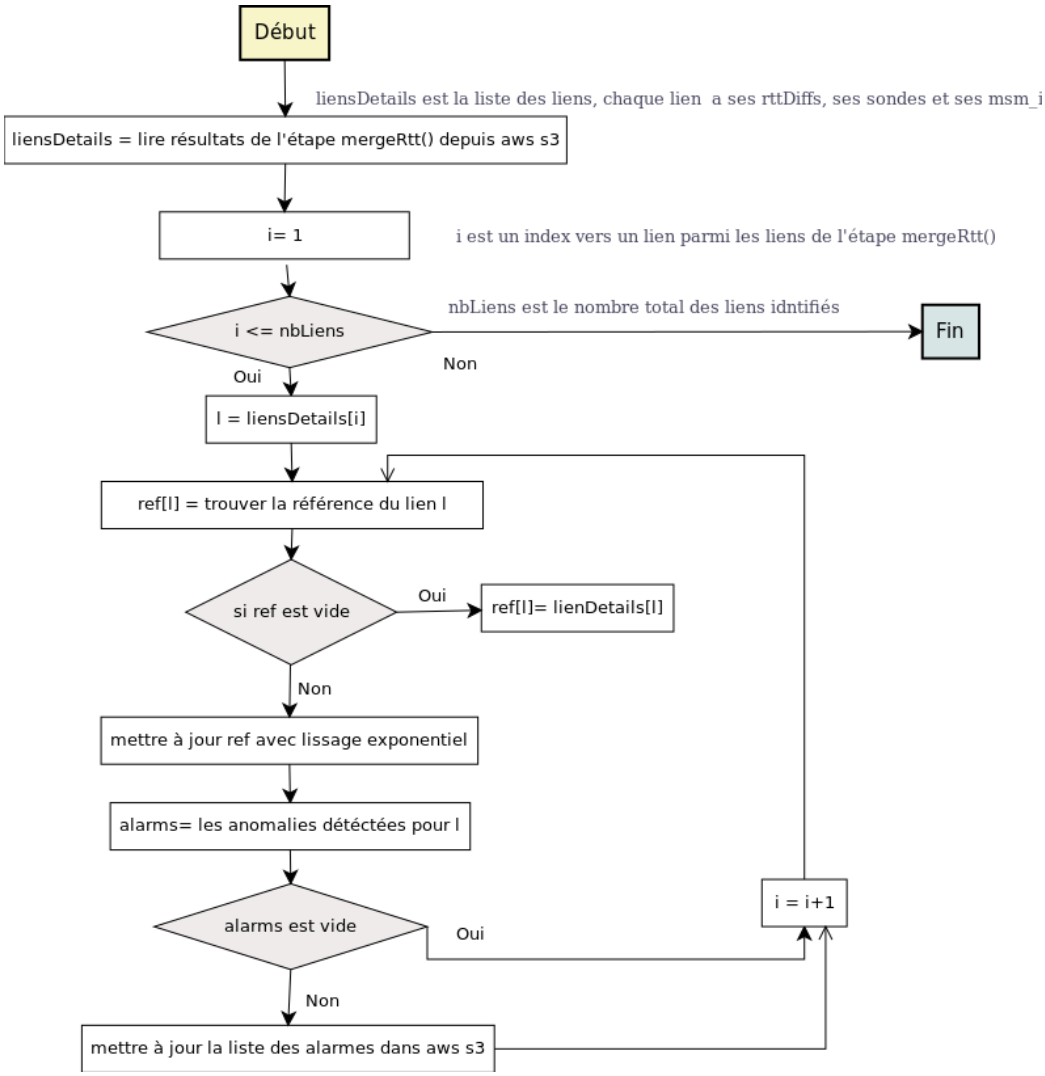


FIGURE II.12 – L'organigramme de l'étape outlierDetection()

Ce qu'on peut noter à travers cet organigramme, c'est que l'évaluation de chaque lien l nécessite l'accès en lecture aux résultats de l'étape mergeRtt() d'une part et d'accéder à la référence de chaque lien en lecture et écriture d'autre part. Si on suppose que les résultats intermédiaires sont stockés dans des fichiers disponibles sur AWS S3, est-il efficace de continuer l'analyse des traceroutes avec les deux service AWS Athena et S3 ?

En pratique si on considère l'analyse ayant les caractéristiques suivantes :

Période	entre 2015-10-20 00 :00 :00 et 2015-10-20 01 :00 :00
Id de la mesure	5004
Nb traceroutes	17816
Adressage IP	4
Nb de liens	26295

TABLE II.5 – Récapitulatif d'une expérience de détection des anomalies

Les données analysées sont disponibles sur GitHub [14]. Le tableau **??** reprend 10 (10/26295) liens issus de l'analyse décrite dans II.5. Ce sont les liens les plus rencontrés durant l'heure de l'analyse, autrement dit, ceux ayant la distribution la plus grande des RTTs différentiel.

Lien	Nb RTT différentiel	Nb de sondes
('192.5.5.241', '80.249.208.111')	3737	3737
('192.5.5.241', '80.81.194.57')	1084	1084
('192.5.5.241', '216.200.0.10')	827	827
('216.200.0.10', '64.125.31.46')	802	802
('192.5.5.241', '80.249.208.140')	629	629
('195.219.194.46', '80.249.208.111')	443	443
('64.125.25.53', '64.125.31.46')	422	422
('192.5.5.241', '193.232.244.140')	420	420
('192.5.5.241', '193.239.116.112')	396	396
('64.125.20.250', '64.125.25.53')	387	387
('192.5.5.241', '195.35.65.250')	371	371
('192.5.5.241', '193.232.246.140')	369	369
('192.5.5.241', '62.115.42.86')	360	360
('192.5.5.241', '206.223.119.2')	353	353
('192.5.5.241', '5.57.80.224')	353	353

Conclusion Le service web d'Amazon Athena permet le mode lecture seulement des données, ainsi les tables créées ne peuvent pas être mises à jour, ce sont des tables utiles pour lire les données présentes sur AWS S3. L'opération de la détection des anomalies des délais dans les liens passe par plusieurs étapes intermédiaires, de ce fait, il faut passer plusieurs opérations de lecture/écriture des résultats intermédiaires. En matière d'efficacité, on peut trouver mieux que les deux services d'Amazon Athena et S3 pour ce projet en particulier. En ce qui concerne

14. Source : https://github.com/hayatbellafkih/RipeAtlasTraceroutesAnalysis/blob/master/data/2015-10-20%2000:00:00_msmId5004.json.gz., consultée le 19/10/2018.

la médiane, on a pas considéré l'efficacité de la fonction *approx_percentile(col, 0.5)* [15] en terme de précision, vu que le choix de ces deux services d'AWS n'est pas efficace pour l'opération de la détection.

15. Plus de détails sur `https://prestodb.io/docs/current/functions/aggregate.html`, consultée le 19/10/2018.

Annexe A

Illustration du théorème central limite

La figure A.1 illustre par l'exemple le principe du TCL avec une distribution qui suit la loi normale.

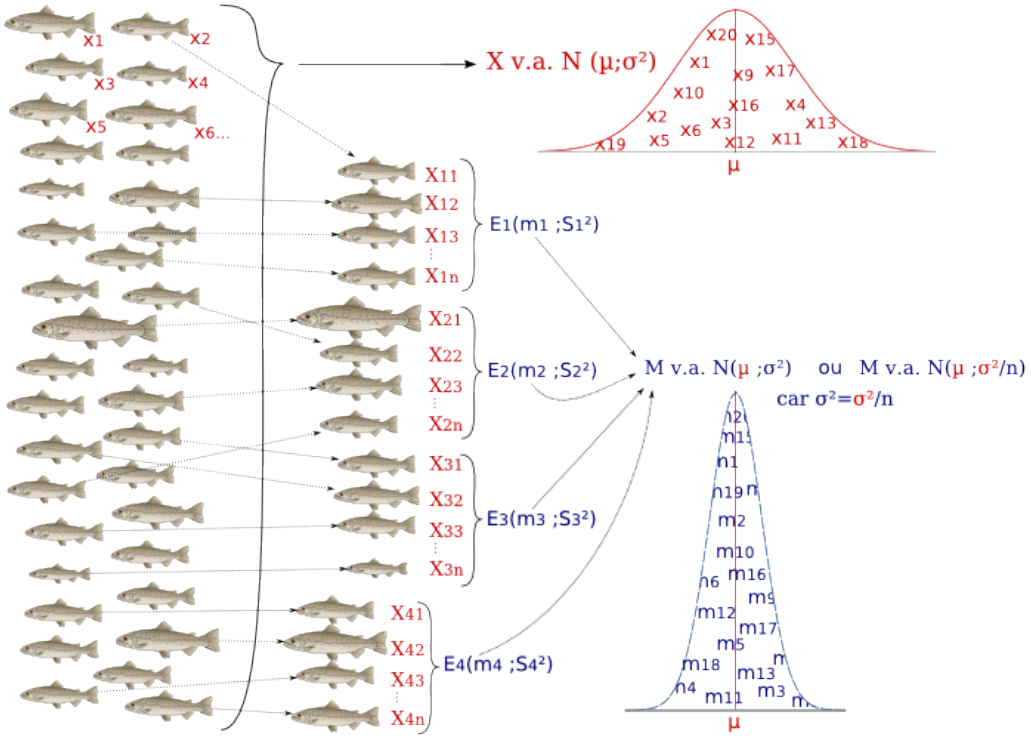


FIGURE A.1

Source : http://webapps.fundp.ac.be/biostats/biostat/modules/module70/page4.html, consultée le 28/09/2018.

Conclusion

Quelques lignes pour finir...

Bibliographie

[1] Archipelago (Ark) Measurement Infrastructure. URL : http://www.caida.org/projects/ark/. (consulté le 19/01/2018).

[2] Center for Applied Internet Data Analysis (CAIDA). URL : http://www.caida.org/. (consulté le 06/04/2018).

[3] Create a New Measurement - RIPE Atlas. URL : https://atlas.ripe.net/measurements/form/. consulté le 05/08/2018.

[4] Le dépôt des données RIPE Atlas. URL : https://data-store.ripe.net/datasets/atlas-daily-dumps/. (consulté le 26/07/2018).

[5] Les archives des détails des sondes atlas. URL : https://ftp.ripe.net/ripe/atlas/probes/archive/. (consulté le 28/01/2018).

[6] RIPE Atlas - Known Bugs and Limitations. URL : https://atlas.ripe.net/docs/bugs/. (consulté le 05/04/2018).

[7] Samknows. URL : https://www.samknows.com/global-platform. (consulté le 23/01/2018).

[8] Xport pro. URL : https://www.lantronix.com/products/xport-pro/. (consulté le 08/08/2018/).

[9] ABEN, E. How RIPE Atlas Helped Wikipedia Users. URL : https://labs.ripe.net/Members/emileaben/how-ripe-atlas-helped-wikipedia-users, 2014. (consulté le 18/08/2018).

[10] ABEN, E. Looking at France-IX with RIPE Atlas and RIS. URL : https://labs.ripe.net/Members/emileaben/looking-at-france-ix-with-ripe-atlas-and-ris, 2015. (consulté le 08/08/2018).

[11] ABEN, E. Measuring Countries and IXPs with RIPE Atlas. URL : https://labs.ripe.net/Members/emileaben/measuring-ixps-with-ripe-atlas, 2015. (consulté le 08/08/2018).

[12] ANDERSON, C., WINTER, P., AND ROYA. Global network interference detection over the RIPE atlas network. In *4th USENIX Workshop on Free and Open Communications on the Internet (FOCI 14)* (San Diego, CA, 2014), USENIX Association.

[13] DONATO, V. D. Traceroute Consistency Check. URL : https://github.com/vdidonato/Traceroute-consistency-check, 2015. (Consulté le 08/08/2018).

[14] FONTUGNE, R., ABEN, E., PELSSER, C., AND BUSH, R. Pinpointing delay and forwarding anomalies using large-scale traceroute measurements. *CoRR abs/1605.04784* (2016).

[15] GASMI, S. Visualising RIPE Atlas Anchor Measurements. URL : https://labs.ripe.net/Members/salim_gasmi/visualising-ripe-atlas-anchor-measurements, 2015. (consulté le 08/08/2018).

[16] GUILLAUME VALADON, FRANCOIS CONTAT, M. H., AND HOLTERBACH, T. BGP Atlas Monito (BAM). URL : https://github.com/guedou/bam, 2015. (consulté le 08/08/2018).

[17] HEROLD, J. Bgp + traceroute presentation. URL : https://labs.ripe.net/Members/becha/ripe-atlas-hackathon-presentations/bgp-traceroute, 2015. (consulté le 08/08/2018).

[18] HEROLD, J. BGP + Traceroute using RIPE NCC Atlas. URL : https://github.com/wires/bgp-traceroutes, 2015. consulté le 08/08/2018.

[19] HOLTERBACH, T., PELSSER, C., BUSH, R., AND VANBEVER, L. Quantifying interference between measurements on the ripe atlas platform. In *Proceedings of the 2015 Internet Measurement Conference* (New York, NY, USA, 2015), IMC '15, ACM, pp. 437–443.

[20] KISTELEKI, R. The AMS-IX Outage as Seen with RIPE Atlas. URL : https://labs.ripe.net/Members/kistel/the-ams-ix-outage-as-seen-with-ripe-atlas, 2015. (consulté le 23/01/2018).

[21] KISTELEKI, R. RIPE Atlas Architecture - how we manage our probes. URL : https://labs.ripe.net/Members/kistel/ripe-atlas-architecture-how-we-manage-our-probes, 2017. consulté le (08/08/2018).

[22] KISTELEKI, R. RIPE Atlas probes as IoT devices. URL : https://labs.ripe.net/Members/kistel/ripe-atlas-probes-as-iot-devices, 2017. (consulté le 21/12/2017).

[23] RIPE NCC. Test Traffic Measurement Service (TTM). URL : https://www.ripe.net/analyse/archived-projects/ttm. (consulté le 14/01/2018).

[24] RODERICK, F. On the Diversity of Interdomain Routing in Africa. URL : https://labs.ripe.net/Members/fanou_roderick/on-the-diversity-of-interdomain-routing-in-africa, 2015. (consulté le 11/01/2018).

[25] SHAO, W., ROUGIER, J., DEVIENNE, F., AND VISTE, M. Missing measurements on RIPE atlas. *CoRR abs/1701.00938* (2017).

[26] SHAVITT, Y., AND SHIR, E. Dimes : Let the internet measure itself. *SIGCOMM Comput. Commun. Rev. 35*, 5 (Oct. 2005), 71–74.

[27] SPEED CHECKER. Global Internet testing - ProbeAPI. URL : http://probeapi.speedchecker.xyz/. (consulté le 06/08/2018).

[28] VARAS, C. A Practical Comparison Between RIPE Atlas and ProbeAPI. URL : https://labs.ripe.net/Members/cristian_varas/a-practical-comparison-between-ripe-atlas-and-probeapi, 2016. (consulté le 19/01/2018).