Télécom
ParisTech

Télécom
SudParis

Mémoire de stage

Sécurité d'un contrôleur SDN : ONOS

Julien Schoumacher

Diplôme préparé : Ingénieur
Stage effectué du 20 juillet 2016 au 20 janvier 2017 à
Télécom SudParis sous la direction de Grégory Blanc

Remerciements

Avant d'entamer la lecture de ce rapport, je tiens avant tout à remercier toute l'équipe du département RST (Réseaux et Services des Télécommunications) de Télécom SudParis qui m'a si bien accueilli durant ce stage. Je remercie également mon encadrant côté Télécom ParisTech Rida Khatoun, m'ayant mis en relation avec le maître de conférences Gregory Blanc qui m'a encadré avec bienveillance pendant toute la durée du stage. Enfin, toutes les autres personnes que j'ai pu cottoyer plus ou moins longtemps à l'occasion d'évènements ponctuels comme la conférence RAID qui s'est tenue en septembre.

Table des matières

1 Réseau SDN (Software Defined Network)

1.1 Motivation

Bien qu'il soit préférable d'éviter les approches rasoirs lorsqu'on souhaite introduire un sujet quelconque, on ne peut pas dans le cas de cette étude portant en partie sur les réseaux SDN, oublier de mentionner quelques éléments difficilement contestables sur les réseaux actuels[1] :

- ※ La demande ne cesse de croître : on observe un accroissement considérable des enjeux liés au traitement de masse importante de données, de l'utilisation de services cloud, du trafic mobile et peut être bientôt de l'utilisation d'objets connectés. Or tous ces éléments présentent le point commun de communiquer avec de nombreuses entités situées sur des réseaux potentiellement éloignés. Cela mobilise donc un trafic réseau intense.
- ※ Les technologies actuelles pour soutenir cette demande énorme sont capables de fournir un débit titanesque : que l'on considère des technologies sans fil ou non, au coeur des réseaux tant au niveau des terminaux des utilisateurs, on atteint aujourd'hui des débits théorique de l'ordre du Gigabit par seconde pour l'utilisateur. Tout cela sans que l'on ait vraiment conscience des conditions que cela requiert.
- ※ Les méthodes d'accès sont aujourd'hui bien différentes. Précédemment le modèle client/serveur était largement employé, avec dans le cas d'une entreprise, un réseau interne constitué de plusieurs LAN séparés, et connecté à internet de manière quasiment unique. Cela entrainant une configuration possiblement statique et donc aisée, le trafic se déroulant principalement sur un mode requête/réponse. Or la tendance, notamment à cause des deux premiers points, est à l'émergence de nouveaux modes d'accès plus horizontaux. Ce type de communication tient entre autres de la distribution plus éparse des données à travers le réseau : grossissement de la taille des bases de données, duplication de celles-ci (mise en cache sur différents serveur à travers le monde pour permettre un accès plus rapide),

1. Aussi résumé dans `https://www.opennetworking.org/sdn-resources/sdn-definition`

augmentation du trafic volumineux (vidéo, voix) et de nouveaux trafics (IoT (Bring Your Own Device), ...) même au sein de l'entreprise. Enfin, l'utilisation de plus en plus répandue de services cloud, avec ses implications en terme de virtualisation (que ce soit des applications, ou bien des bases de données), susceptible de changer en permanence la localisation des serveurs pour garantir une certaine flexibilité.

Or, le réseau principal global tel que nous le connaissons (la partie reposant sur TCP/IP en tout cas) a été conçu d'abord dans un but de résilience : chaque paquet doit être reçu, et peu importe la route empruntée. L'architecture distribuée actuelle n'est donc pas batie pour assurer spécifiquement extensibilité, ni qualité de service définie. Le routeur (et le réseau d'ailleurs) des années 1980 a donc été progressivement amélioré sur la base de ce paradigme initial, avec le plan de données et le plan de contrôle attachés aux mêmes équipements, configurés en partie manuellement. Tout s'est complexifié également : nouveaux protocoles, ajouts d'équipements capables de répartir la charge réseau, filtrer les paquets, prévenir de certaines tentatives d'attaque, etc ...
Certains éléments de réflexion peuvent éventuellement nous mettre sur la voie d'une complexité qui, à défaut d'être exponentielle, l'est d'avantage que simplement linéaire (c'est du moins une conviction personnelle non vérifiée) :

- Plus il y a d'éléments statiques dans un réseau, et plus la modification en profondeur de celui-ci est coûteuse (puisqu'il faut penser à chaque impact sur les parties statiques).
- Si un problème survient, dans le même ordre d'idée, il est difficile d'avoir une vue globale de ce qui se passe puisqu'aucune vue globale du réseau n'est accessible : il faut vérifier (potentiellement) que chaque élément se comporte correctement et est bien configuré.
- Les interfaces entre switchs, routeurs et autres éléments peuvent varier selon le constructeur, et le logiciel sur les équipements est souvent propriétaire et complexe.

De nombreux problèmes se résolvent avec un niveau d'abstraction supplémentaire[2]. Si le développement de systèmes de plus en plus complexes s'est fait de manière très rapide sur PC, c'est d'abord grâce à la première couche d'abstraction qu'ont concstituées les instructions assembleur, puis à la seconde qu'a été le système d'exploitation. Certaines personnes ont eu l'idée, au lieu de considérer le réseau comme un élément périphérique, de le voir comme un processeur capable d'exécuter des instructions basiques, fournissant de fait un service plus facilement adaptable. C'est sur ce principe que repose le Software Defined Network (SDN). Avec une couche d'abstraction supplémentaire que constitue le protocole choisi pour véhiculer le flot d'instructions (Openflow dans notre cas, mais il y en a d'autres que nous évoquerons succintement plus tard), et un système d'exploitation spécifique (Network Operating System, NOS), l'idée est de découpler les différents chemins qu'empruntent les données et le plan de contrôle, à la manière d'un système d'exploitation qui sépare le code d'un programme et les données qu'il utilise.

1.2 Concepts

On peut poursuivre la dernière analogie (avec un ordinateur) de la manière suivante. Sur un PC classique, on crée et utilise des applications qui reposent sur un système d'exploitation responsable des éléments matériels. C'est la même chose dans un modèle SDN : on souhaite créer des applications "réseau" sans se soucier de la manière dont les éléments de ce dernier vont s'organiser pour répondre à la problématique. On sépare ainsi la couche applicative, le système d'exploitation réseau et la communication entre les entités du réseau.

Pour réaliser cela, il est nécessaire, puisqu'un réseau est constitué d'entités physiquement séparées, de disposer d'un protocole de communication standard entre celles-ci. Mais ça n'est pas suffisant : si tous les éléments communiquent entre eux, encore faut-il qu'ils aient des

2. https://en.wikipedia.org/wiki/Fundamental_theorem_of_software_engineering

choses à se dire pour faire circuler, outre les données à faire transiter, les instructions qui vont leur indiquer quoi faire avec ces données. Cela n'est possible que si il existe un cerveau central qui coordonne les opérations (il n'existe pas vraiment d'intelligence collective à ce jour). C'est le rôle du contrôleur SDN. On déporte ainsi l'intelligence humaine déployée dans la configuration de tous les éléments du réseau vers un seul (même si il peut être dupliqué).

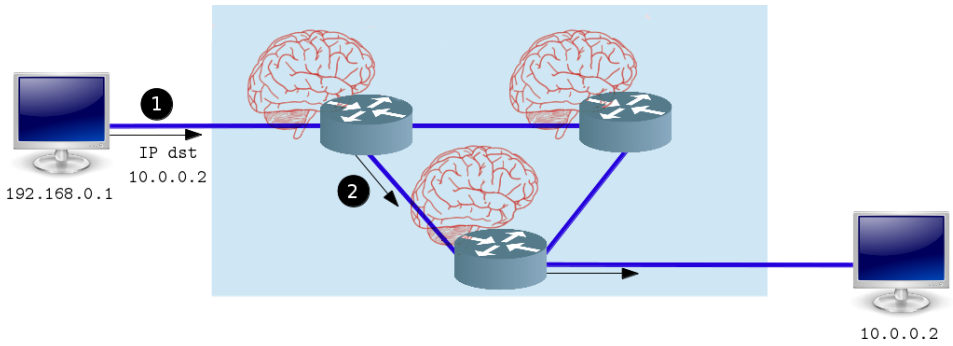

FIGURE 1 – Réseau classique : chaque routeur contient une partie de la logique de contrôle

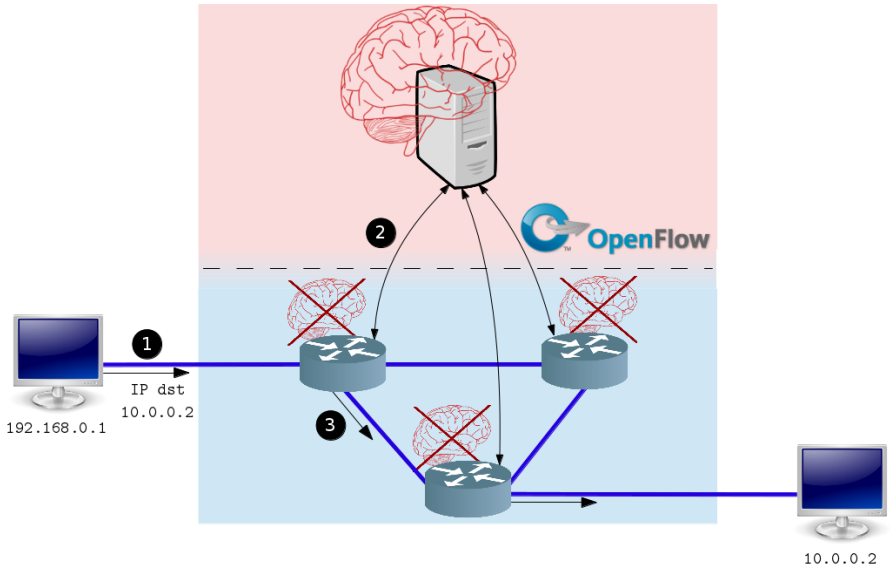

FIGURE 2 – Réseau SDN : "l'intelligence" est déportée vers le contôleur [3]

Les avantages de cette architecture sont multiples :

※ D'abord cela réduit grandement la complexité de configuration manuelle et le risque d'erreur (si le système d'exploitation réseau est

3. Schéma extrait du mémoire "Étude d'OpenFlow dans le contexte de la sécurité" de Maxence Tury

fiable).

- Cela facilite donc énormément le développement d'applications réseau complexes, puisque la tâche peut être quasiment séparée de sa réalisation physique.
- Les routes optimales sont plus facilement calculables qu'au sein d'un réseau classique : un seul élément gère les différentes distances et métriques qui peuvent changer selon le trafic, et être modifiées à la volée par des applications spécifiques.
- La gestion du réseau devient plus simple, les évènements importants (perte d'un lien, disfonctionnement, ralentissement ...) peuvent être remarqués rapidement, la réaction pouvant être automatique et quasiment instantanée.
- Les coûts matériels sont diminués puisque seuls subsistent les switchs "de base" et le contrôleur qui n'est pas réellement un équipement spécifique. Le reste peut être pris en charge logiciellement au niveau du contrôleur.

Bref, pour résumer, beaucoup plus de flexibilité est permise par cette approche, économisant temps et matériel. Evidemment, l'idée n'est pas nouvelle, mais n'a pas que des avantages. Notamment : la sécurité du contrôleur devient un point brûlant, puisque toute la gestion du réseau provient de lui.

1.3 Historique

La ressource "A Survey of Software-De ned Networking : Past, Present, and Future of Programmable Networks" [4] présente un état de l'art à la date du 19 janvier 2014 (donc assez récemment pour avoir une vue globale de l'évolution de ce type de technologie). En voici un rapide résumé complété :

Assez tôt lors de l'essor d'internet tel que nous le connaissons, des idées pour fournir une sorte d'API réseau ont émergées (milieu des années 1990 environ) : le groupe Open Signaling propose un protocole

4. https://hal.inria.fr/hal-00825087/file/hal_final.pdf

d'accès universel au matériel (switchs) permettant de distribuer facilement des nouveaux services, pendant qu'Active Networking propose un mécanisme de propagation de code que l'équipement réseau exécute lorsqu'il reçoit les paquets encapsulant le code (ce qui pose au passage un énorme problème de sécurité).

Dans le même temps, le groupe DCAN (Devolved Control of ATM Networks) propose une approche qui se rapproche très fortement du paradigme SDN : convaincus que les fonctions de contrôle et de gestion des différents éléments du réseau doivent être séparées du routage des données et déléguées à des équipements spécialisés, ils développent un protocole minimaliste entre contrôleur et autres équipements, à la manière du protocole Openflow aujourd'hui majoritaire dans les réseaux SDN.

Le projet 4D [5] initié en 2004 (et semblant s'être fini un peu avant 2010), ajoute quant à lui des abstractions diverses (découverte des voisins proches et remontée des informations, dissémination d'informations sur l'état général du réseau, puis prise de décision sur la base des informations récoltées). C'est ce genre de projet qui a inspiré l'idée de système d'exploitation réseau, qu'implémentent les contrôleurs SDN.

On peut encore citer NETCONF et Ethane (2006), le premier pouvant être vu comme une extension de SNMP, le second comme un ancêtre immédiat d'openflow (un contrôleur qui décide si et où un paquet devrait être redirigé, et des switchs qui constitués d'une table de flux et d'un canal sécurisé vers le contrôleur).

Openflow a quant à lui précédé l'apparition du terme SDN lors d'expérimentations à Stanford vers 2010 (la première spécification d'Openflow pour la production (1.0.0), a été publiée début 2010).

5. http://www.cs.cmu.edu/~4D/

1.4 Exemples d'applications

En 2011, l'Open Networking Foundation (ONF) est créée. Regroupant des gros acteurs comme Google, Yahoo, Facebook, Verizon, Microsoft ou encore Deutsche Telekom, c'est l'organisme principal qui encourage l'adoption de la technologie SDN, en publiant régulièrement de nouvelles spécifications Openflow.

Google, en 2012, présente, pour la première fois, une architecture SDN pour ses datacenters, utilisant Openflow sur des switchs conçus par eux-mêmes (étant pionniers, leur position étant qu'ils auraient utilisé des switchs existants si ceux-ci implémentaient toutes les fonctionnalités Openflow leur étant nécessaires). Grâce à ce nouveau paradigme, Google affirme (en 2012) obtenir des performances dix fois supérieures en terme de débit, et surtout utiliser 100% [6] de leurs lignes (contrairement aux 30 à 40% en vigueur dans l'industrie, notamment pour garantir un service même en cas de nombreuses pannes, ce qui n'est plus nécessaire avec un réseau SDN qui adapte automatiquement le routage pour pallier aux problèmes).

Microsoft semble également s'être intéressé au SDN pour son service "Azure" depuis quelques années maintenant [7], et de manière générale toutes les figures de proue de l'industrie numérique (celles qui disposent de nombreux serveurs et gèrent des flux énormes et grandissants de données comme Amazon, AT&T, Facebook, ...) se sont plus ou moins annoncées investies dans le processus, possiblement avec leur propre protocole SDN. Si des grosses entreprises ont annoncé l'utilisation de ce type de réseau, les données précises concernant les performances obtenues sont difficilement accessibles, ce qui rend compliquée l'évaluation des avantages réels de SDN.

Par ailleurs, les switchs compatibles Openflow, ou les switchs Openflow seuls, que fournissent (depuis 2011 environ) certaines entreprises majeures du domaine comme Brocade, HP, IBM, Juniper ou encore

6. http://www.networkworld.com/article/2189197/lan-wan/google-s-software-defined-openflow-backbone-drives-wan-links-
html
7. http://www.networkworld.com/article/2937396/cloud-computing/microsoft-needs-sdn-for-azure-cloud.
html

NEC, Pronto ou Pica8, manquent encore parfois de maturité (RFC parfois interprétée différemment, vulnérabilités spécifiques, ...). L'évolution des versions Openflow est également parfois difficile à suivre (la spécification de la version 1.0.0 (fin 2009) fait 42 pages [8], celle de la dernière version stable (1.5.0, fin 2014) en fait 277 [9]). Or l'industrie dans ce domaine présente une certaine inertie (peu sont les compagnie qui proposent des switchs compatibles avec la dernière version d'Openflow, certaines vendant encore des switchs Openflow 1.0).

Les applications semblent donc d'un premier abord limitées aux datacenters (beaucoup de données à traiter, grande variabilité de la topologie (les machines virtuelles changent souvent d'emplacement/adresse)). En réalité, SDN peut être utilisé de manière effective dans plusieurs cas :

- Réseaux d'entreprises/universités : dans le cas de nombreux équipements/protocoles différents utilisés, SDN permet théoriquement de faciliter le déploiement de politiques réseau complexes
- Réseaux optiques : faciliter la transition, la gestion et l'incorporation des réseaux optiques au sein du réseau actuel.
- Infrastructure based WAN (réservé aux entreprises disposant de nombreux serveurs à travers le monde) : à la manière de Google, un moyen de connecter de grosses entités en évitant les goulots d'étranglement. Egalement un moyen envisageable pour un utilisateur de se connecter depuis n'importe quel endroit en disposant des services auxquels il souscrit.
- Infrastructure personnelle, petites entreprises : surveiller le trafic et alerter l'administrateur local ou le fournisseur internet en cas de détection d'activité réseau suspecte.

Au final, une technologie pouvant sembler prometteuse, mais demeurant assez peu utilisée pour plusieurs raisons. Nous allons en étudier deux d'entre elles par la suite : la sécurité générale du réseau, et l'im-

8. http://archive.openflow.org/documents/openflow-spec-v1.0.0.pdf
9. https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-switch-v1.5.0.noipr.pdf

portance capitale du contrôleur qui devient quasiment l'unique clé de voûte du système.