SUJET : Software-defined networking : principes et solutions

Sommaire

[Introduction 4](#_Toc509825725)

[I. Le software-defined networking : Définition et enjeux 4](#_Toc509825726)

[A. Comprendre le SDN et son utilité. 4](#_Toc509825727)

[1.Networking sans software. 4](#_Toc509825728)

[1.1 Définition d’un réseau 4](#_Toc509825729)

[1.2 Caractéristique d’un réseau 5](#_Toc509825730)

[2. Le software-defined networking, une technologie nouvelle et en évolution. 6](#_Toc509825731)

[2.1 Définition du SDN 6](#_Toc509825732)

[2.2 Une technologie nouvelle sur le marché 6](#_Toc509825733)

[2.3 L’évolution du SDN sur le marché 7](#_Toc509825734)

[3.Les avantages du SDN. 7](#_Toc509825735)

[3.1 Configuration simplifiée 7](#_Toc509825736)

[3.2. L’amélioration des performances. 8](#_Toc509825737)

[3.3 Indépendance faces aux fournisseurs. 8](#_Toc509825738)

[B.Le software-defined networking une nouvelle approche 9](#_Toc509825739)

[1.Comprendre un réseau défini par logiciel. 9](#_Toc509825740)

[2.Comprendre le plan de donnée de réseaux 9](#_Toc509825741)

[C. L’architecture du SDN 9](#_Toc509825742)

[1.Couche infrastructure 10](#_Toc509825743)

[2.Couche contrôle 11](#_Toc509825744)

[3.Couche applications 12](#_Toc509825745)

[II. Moyen utilisé par le software-defined networking 14](#_Toc509825746)

[A. Les API les plus répandus 14](#_Toc509825747)

[1.ForCES 14](#_Toc509825748)

[2.OpenFlow 14](#_Toc509825749)

[3.NETCONF/YANG 16](#_Toc509825750)

[III. Mise en oeuvre d’un SDN avec openFlow 18](#_Toc509825751)

[Création de différente typologie de réseau à l’aide de mininet 23](#_Toc509825752)

[Configuration d’un réseau SDN avec un contrôleur OpenFlow et un seul switch. 24](#_Toc509825753)

[Configuration d’un réseau SDN avec deux switch. 25](#_Toc509825754)

[Création d’un load-balancer 25](#_Toc509825755)

[Conclusion 27](#_Toc509825756)

[Annexes 28](#_Toc509825757)

[Bibliographie 29](#_Toc509825758)

[A. Ouvrages 29](#_Toc509825759)

[B. Sites Internet 29](#_Toc509825760)

[C. Documents audiovisuels 29](#_Toc509825761)

[Glossaire 30](#_Toc509825762)

[Table des illustrations 31](#_Toc509825763)

## 

## 

## Introduction

## Le software-defined networking : Définition et enjeux

### A. Comprendre le SDN et son utilité.

#### 1.Networking sans software.

#### 1.1 Définition d’un réseau

Un réseau est construit avec un grand nombre de périphérique tels que les switches, les routeurs, des pares-feux, les commutateurs réseaux, les commutateurs téléphoniques, les antennes GSM, d’intrusion detection System (IDS) pour la détection d’intrusion. Chaque périphérique a un protocole qu’il faut configurer. Généralement il s’agit du protocole IP permettant d’acheminer la donnée au niveau de la couche réseau du Modèle OSI. Il est nécessaire également d’avoir des logiciels dédiés sur chaque périphérique afin de gérer ces protocoles.

On retrouve deux autres protocoles qui sont UDP et TCP qui se situe au niveau de la couche Transport du modèle OSI. L’UDP est un protocole qui permet le transport de donnée d’une manière non sécurisé. TCP permet le transport de données d’une façon fiable.

Les différents services du réseau doivent alors être configuré un a un au risque de faire des erreurs de configuration dans le routage ou même la sécurité du réseau lors de la configuration d’un pare-feu ou un proxy.[[1]](#footnote-1)

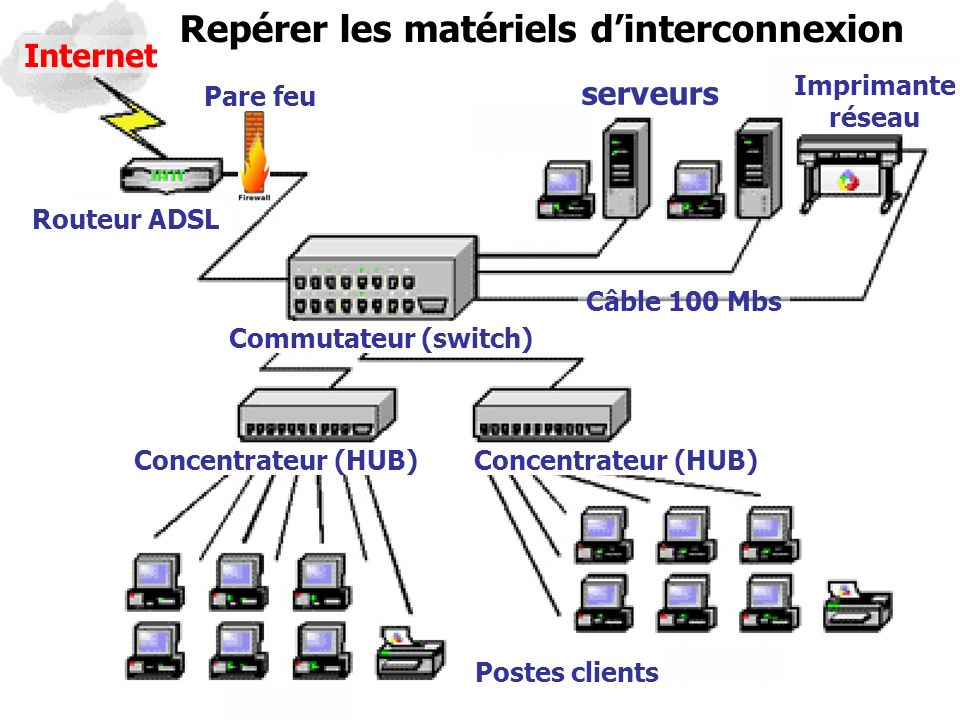


Figure Réseau informatique sans Software Defined Networking[[2]](#footnote-2)

Voici la représentation d’un réseau qui pourrait être celui d’une PME contenant des ordinateurs, une imprimante, des serveurs. Ces périphériques sont tous connectés à l’aide de câbles qui garantissent la communication à l’aide de switch ayant le rôle de gare routière.

Nous avons également un routeur ADSL permettant le routage de la donnée d’un réseau à un autre et également d’un pare-feu placé entre le réseau et le routeur, celui-ci est utile pour la sécurité du réseau.

#### 1.2 Caractéristique d’un réseau

On définit un réseau selon six caractéristiques principales.

Tout d’abord un réseau est défini par sa topologie. La plus utilisé est la topologie en étoile. Contenu de la complexité des réseaux actuellement il est plus correct d’admettre qu’il y a utilisation hybride des différentes topologies.

La vitesse de transmission des données est un élément important. Elles sera déterminés en fonction des câbles utilisés, de la performance des câbles et s’il s’agit d’un réseau ADSL, la fibre, l’ATM, l’UMTS etc... source : <https://www.youtube.com/watch?v=ds4VP0Ibc_M&t=310s>

Le coût de l’infrastructure d’un réseau, de sa maintenance est également un point critique chez les architectes réseaux qui ont des contraintes de budget imposé par les entreprises.

La sécurité du réseau pour les entreprises mais aussi les particuliers, est un aspect primordial pour garantir l’intégrité, la non altérité des données ainsi que la confidentialité des données.

Les utilisateurs du réseau souhaitent que l’accès à la donnée ou un matériel soit possible quel que soit le trafic sur le réseau. Le réseau doit être disponible.

Enfin le réseau doit pouvoir être fiable en informant de la transmission de la donnée ou de la récupération de la donnée par un autre périphérique.

Ayant défini ce qu’était un réseau, nous sommes maintenant en mesure de définir en quoi le SDN est une nouvelle technologie et ce qu’il apporte de bénéfique.

#### 2. Le software-defined networking, une technologie nouvelle et en évolution.

#### 2.1 Définition du SDN

Voici la définition du SDN selon le site internet [opennetworking](https://www.opennetworking.org/sdn-definition/) :

*Software-Defined Networking (SDN) is an emerging architecture that is dynamic, manageable, cost-effective, and adaptable, making it ideal for the high-bandwidth, dynamic nature of today’s applications. This architecture decouples the network control and forwarding functions enabling the network control to become directly programmable and the underlying infrastructure to be abstracted for applications and network services. The OpenFlow protocol is a foundational element for building SDN solutions.*

Au-delà de cette définition il y a trois composantes importantes qui définissent ce qu’est une architecture SDN:

* La décorrélation du plan de contrôle et du plan de donnée.
* L’abstraction du réseau physique.
* La programmabilité du réseau.

source : <https://www.randco.fr/actualites/2014/sdn-et-openflow/>

#### 2.2 Une technologie nouvelle sur le marché

Le SDN permet de programmer le réseau à partir d’un contrôleur ce qui constitue une nouveauté.

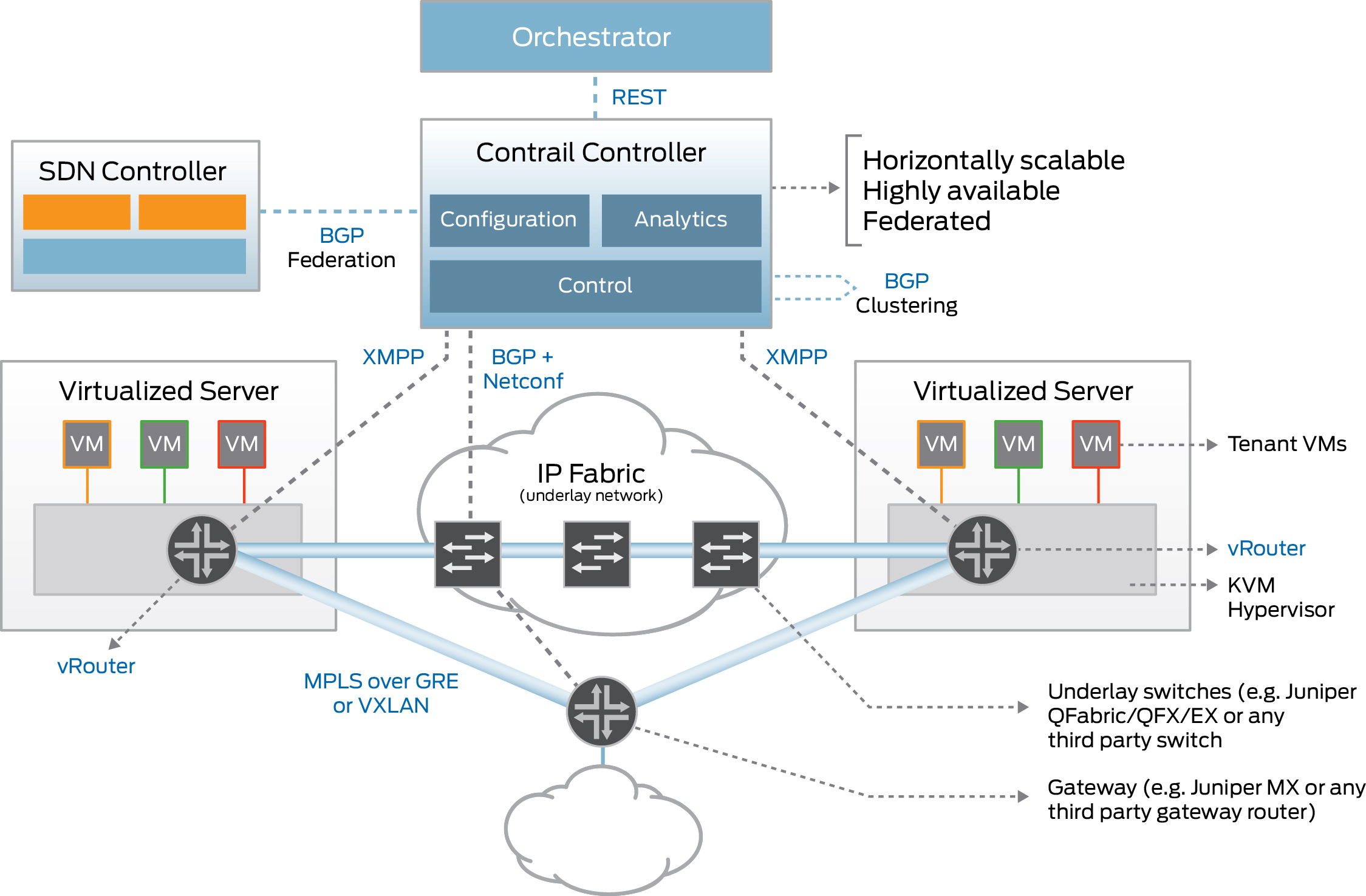


Figure Contrôleur SDN

La figure ci-dessus montre que le contrôleur du SDN à un rôle de maître d’orchestre. Il est capable d’identifier tout changement sur le réseau. Par exemple la panne d’un routeur, l'ajout d’un ordinateur ou d’un switch. Les logiciels mis en place dans les différentes couches voir cf. chapitre B1).

#### 2.3 L’évolution du SDN sur le marché

En 2009 il n‘y avait pas encore d’entreprises sur le marché qui développait des solutions SDN. (Westcon\_5\_minute\_expert\_SDN\_FR). Aujourd’hui le SDN représente un marché de près de 40 millions. Cette technologie à une approche logiciel.

#### 3.Les avantages du SDN.

#### 3.1 Configuration simplifiée

La configuration d’un réseau est une des étapes les plus importantes pour le bon fonctionnement de celui-ci. C’est aussi une étape qui est sujet à beaucoup d’erreur de configuration. L’ajout d’un nouvel équipement sur le réseau nécessite une bonne configuration de réseau afin que celui-ci soit cohérent. Il nécessite généralement une configuration manuelle du fait de l’hétérogénéité des fabricants de périphérique réseau et des interfaces de configuration. De plus s’il s’avère qu’il y a une erreur de configurations ou une erreur de réseaux quelconque. Sans SDN, il est nécessaire de se déplacer sur site.

Avec une configuration SDN, l’unification du plan de contrôle sur différent types de dispositif de réseau incluant les commutateurs, les routeurs, les traducteurs d’adresses réseau (NAT), les pare-feu et load balancers, permet de configurer les périphériques réseau à partir d’un seul point, automatiquement via le logiciel de contrôle.[[3]](#footnote-3)source :(a survey on software defined networking chapitreB.1)page29)

#### 3.2. L’amélioration des performances.

L’une des problématiques sur un réseau est de conserver de bonnes performances.

Une tendance est d’améliorer l’expérience utilisateur uniquement sur certain service réseau ou d’optimiser les performances sous un sous-ensemble du réseau.

Le SDN grâce à une vue globale permet d’avoir une vue globale du réseau et un contrôle de rétroaction avec des informations échangées entre les couches de l’architecture du SDN.

Les algorithmes au sein du SDN, permet de gérer l’optimisation du réseau en fonction de l’utilisation des différents sous-réseaux.

Ainsi le SDN règle des problèmes tel que l’ordonnancement de trafic, le contrôle de congestion de bout en bout, de routage équilibré par charge, l’économie d’énergie.

L’amélioration de ces différents services contribuent à l’amélioration de la performance générale du réseau.source :(a survey on software defined networking chapitreB.2)page29)

#### 3.3 Indépendance faces aux fournisseurs.

Le SDN permet le déploiement plus rapide et des applications et des services.

Il limite les erreurs lors du déploiement des applications. Il permet un approvisionnement plus rapide des nouveaux utilisateurs ou clients. Le SDN permet également de réduire les coûts de fonctionnement, de maintenance et de structure.

Enfin il garantit une indépendance des faces aux fournisseurs. (Westcon\_5\_minute\_expert\_SDN\_FR) et ( a survey on software defined networking page29). (a survey on software defined networking chapitreB.1page29)

### B.Le software-defined networking une nouvelle approche

#### 1.Comprendre un réseau défini par logiciel.

Un réseau défini par logiciel est une nouvelle approche d’architecture comprenant trois couches. On va y retrouver les couches suivantes :

* La couches infrastructure où sont les équipements de transmission réseau.
* La couche contrôle : fournit configuration et ordre de transmission. Elle est en charge de configurer la couche dite application. La couche contrôle distribue les requêtes de services sur la couche infrastructure de la manière la plus optimal possible par configuration dynamique.
* La couche applications sert aux applications du cloud de gestion et d’entreprise pour placer leurs requêtes sur le réseau par la couche contrôle.

#### 2.Comprendre le plan de donnée de réseaux

### C. L’architecture du SDN

Voici une représentation d’un modèle de référence pour SDN selon L’Open Networking Foundation qui se compose de trois couches qui sont la couche infrastructure, la couche de contrôle et la couche d’application. (cf survey on sdn.pdf page 30 section D)

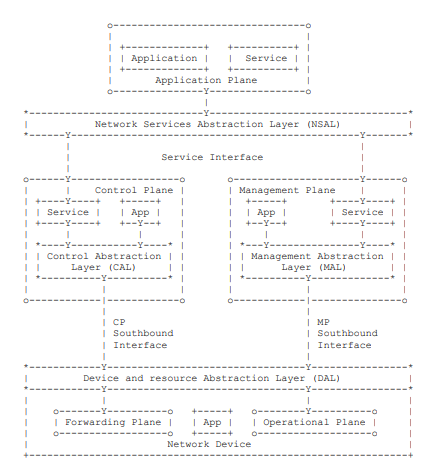


Figure SDN couches et architecture : source : (Layers&Architecture-technologie-SDN.pdf p8

#### 1.Couche infrastructure

La couches réseau est constitué de commutateurs, routeurs, serveurs. Ils sont d’abord responsables pour collecter le statut du réseau, les stocker temporairement dans les périphériques locaux puis les envoyer aux contrôleurs. Le dispositif de commutation envoie aux contrôleurs des informations comme la topologie du réseau, le trafic réseau ainsi que l’utilisation du réseau.

De plus, en fonction des directives de la couche contrôleur, la couche infrastructure est responsable de l’acheminement des paquets. (cf survey on sdn.pdf page 30 section d)

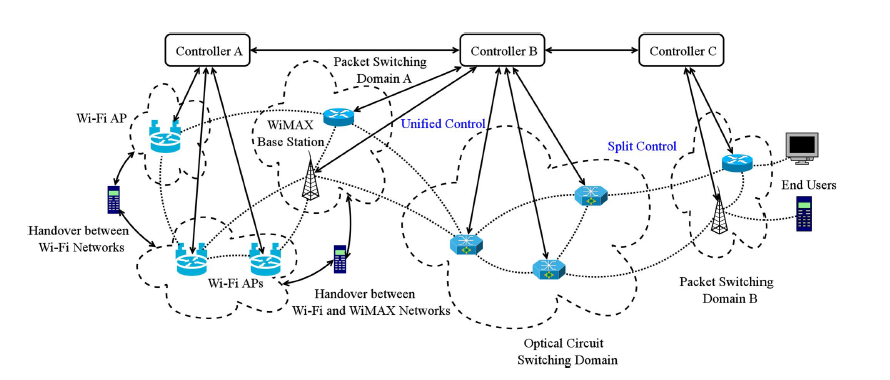


Figure : SDN Infrastructure Architecture : switching devices are connected to formulate a mesh topology via various transmission media, including copper wires, wireless radio and optical fibre[[4]](#footnote-4) (source : survey on sdn.pdf fig2)( Xia, Wenfeng, et al. "A survey on software-defined networking." *IEEE Communications Surveys & Tutorials* 17.1 (2015):)

#### 2.Couche contrôle

La couche dit de contrôle est le cerveau et la tour de contrôle du SDN.

Il a en mémoire la topologie du réseau. C’est lui, qui en fonction du trafic, décide de diriger le flux sur le réseau.

Cette couche à la capacité de comprendre différent protocoles réseau ce qui lui permet de configurer si un nouveau périphérique est connecté au réseau. Il a également la capacité de communiquer avec d’autres contrôleurs. (cf sdn-décodage.pdf page3)source(https://www.juniper.net/fr/fr/local/pdf/whitepapers/sdn-whitepaper.pdf) Cette communication permanente avec les autres contrôleurs du réseau affecte les performances, et la latence du réseau et dans le même la possibilité d’évolution de cette couche.[[5]](#footnote-5) (cf survey on sdn p37 section C)(Xia, Wenfeng, et al. "A survey on software-defined networking." *IEEE Communications Surveys & Tutorials* 17.1 (2015): 37).

2.1 La performance peut être améliorer en utilisant des techniques comme le parallélisme ainsi que le dosage du réseau lors du traitement des données[[6]](#footnote-6) (cf survey on sdn p37 section C1))(Xia, Wenfeng, et al. "A survey on software-defined networking." *IEEE Communications Surveys & Tutorials* 17.1 (2015): 37). Cette technique est utilisé dans McNettle. (Scaling\_Software-defined\_network\_Controlers on multicore servers.pdf page 3)

McNettle est un système de contrôle SDN extensible et évolutif basé sur des commutateurs openFlow. Il est répond à deux principes à savoir l’utilisation de serveurs multi coeurs et un environnement de calcul polyvalents.Les expériences montrent que McNettle permet de servir jusqu’à 5000 commutateurs en utilisant un seul contrôleur. Il est capable d’une telle performance car il alloue de la mémoire, il planifie les gestionnaires d’évènements, optimise le message, analyse et sérialise, réduit le nombre de système d’exploitation et le temps des systèmes d’exécution (cf source A. Voellmy, B. Ford, P. Hudak, and Y. R. Yang, “Scaling softwaredefined

network controllers on multicore servers,” Comput. Sci., Yale

Univ., New Haven, CT, USA, Yale CS TR 1468, 2012.)

)

Pour l’optimisation on utilise également Maestro dont la fonctionnalité principale est la prise de décision de plan de contrôle. Cela signifie qu’en fonction du paquet en entrée, Mastro va déterminer le nombre de cycles processeur nécessaire et le nombre de messages de configuration à envoyer. (cf CaiZ.pdf page 26 section 2.2.2). Il démontre que cette conception permet des performances accrues et une évolutivité de performance casi-linéaire sur des processeurs multi-coeurs.

#### 3.Couche applications

Cette couche contient les applications du SDN. Les applications communiquent avec la couche contrôle afin d’atteindre une fonction du réseau en fonction du besoin des opérateurs du réseau. Il existe différent type d’application et ceux en fonction de leur fonctionnalité. On y trouve les applications gérant la sécurité, la qualité du service (Qos), l’ingénierie du trafic (TE), les listes de contrôle d’accès universel Gestion (U-ACL) et l’équilibrage de gestion (LB). Les applications SDN sont développées afin de répondre à un besoin spécifique. On peut prendre l’exemple des applications gérant cloud computing, la virtualisation. [[7]](#footnote-7)(cf source Survey&LayeredTaxonomyOfSDN.pdf page 1970 section C1).

SDMN pour Software-Defined mobile networking est une approche de la conception de réseaux mobile où les fonctionnalités spécifiques au protocole sont implémentées dans un logiciel. Il est proposé comme une extension au SDN. source <https://en.wikipedia.org/wiki/Software-defined_networking#Applications>

L’application de sécurité Anomalie detection systems (ADS) est une application qui va permettre de détecter les attaques DDoS (Distributed Denial of Service) avec l’utilisation de contrôleurs NOX. Dans cette approche le contrôleur récupère les informations demandées par l’application ce qui va permettre de détecter rapidement d’éventuel attaques.[[8]](#footnote-8)

Sans une approche logicielle, il aurait fallu mettre un système de détection sur chaque commutateur afin d’arriver au même résultat ce qui engendre un coût financier et de temps considérablement plus élevé.

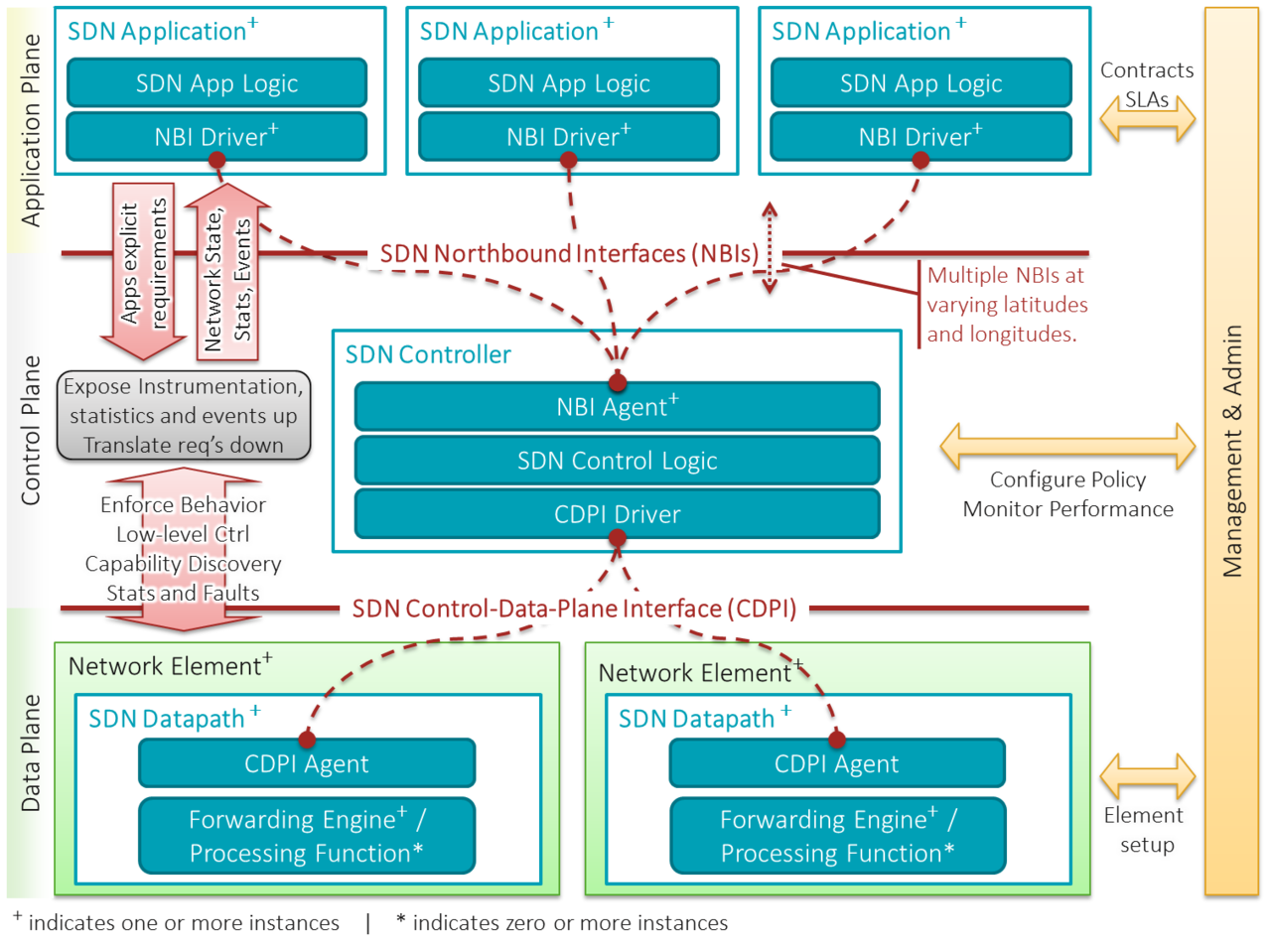


Figure : Fonctionnement entre la couche application et la couche contrôleur. [[9]](#footnote-9)

Sur le schéma ci-dessous on constate que la couche application envoie ses exigences à la couche contrôleur. Entre les deux couches un driver (API) est représenté le NBI (Northbound Interfaces) qui permet de faire la jonction entre les deux couches[[10]](#footnote-10) . D’une manière plus précise son rôle est de fournir une API de haut niveau entre le contrôleur et les applications pour faciliter le calcule et certaines opérations de réseau en fonction des événements pour le plan de données envoyé par le contrôleur et les applications SDN (Nunes, Bruno Astuto A., et al. "A survey of software-defined networking: Past, present, and future of programmable networks." *IEEE Communications Surveys & Tutorials* 16.3 (2014): 1617-1634.)

## II. Moyen utilisé par le software-defined networking

### A. Les API les plus répandus

### 1.ForCES

### 2.OpenFlow

Le protocole OpenFlow définit une interface pour le SDN. Il fait la jonction entre les commutateurs et le contrôleur. Les commutateurs sont connectés en TCP en permanence avec le contrôleur du SDN dans un réseau de contrôle VLAN désigné. Les contrôleurs envoient des commandes sur ces cessions afin de modifier les tables de flux de commutateurs. Cela consiste en une” suite de règles pour acheminer selon des conditions les paquets.

Les commutateurs Openflow communiquent aux contrôleurs les évènements les plus importants. Un contrôleur peut émettre des modifications de table à plusieurs commutateurs en réponse à un évènement d’absence de paquet afin de fournir un chemin à travers le réseau pour le flux. [[11]](#footnote-11)cd( Scaling\_Software-defined\_network\_Controlers on multicore servers.pdf partie 2)

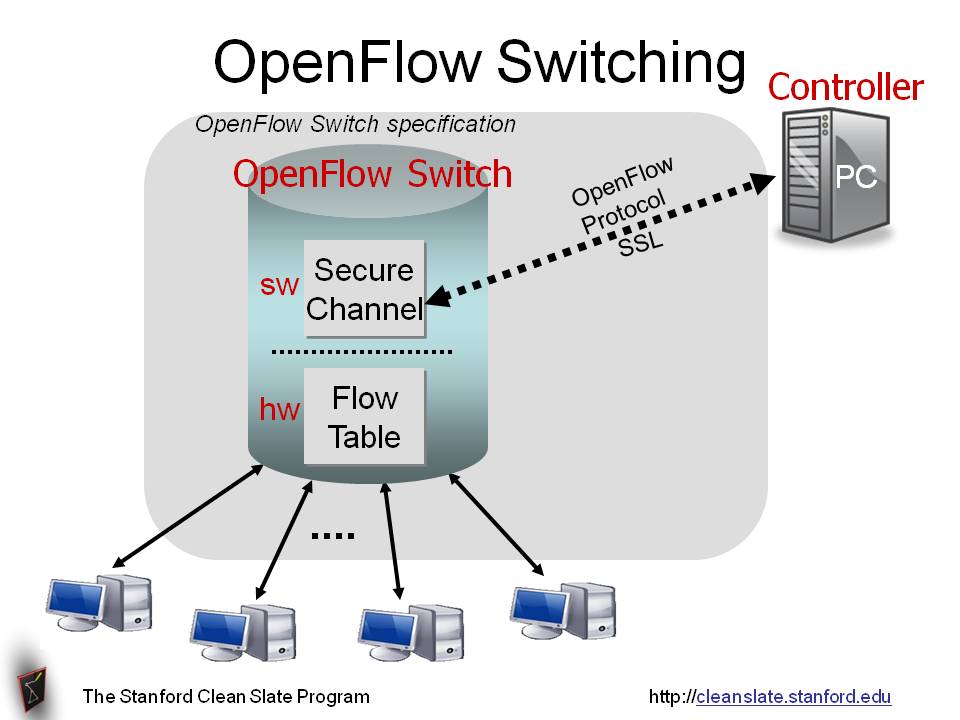


Figure OpenFow Switch Communication dans un réseau SDN

Ce schéma met en lumière la position centrale qu’à le protocole OpenFlow dans la transmission des flux entre les commutateurs et le contrôleur du SDN.

#### 

Les outils du contrôleur OpenFlow [[12]](#footnote-12)

**NOX:[[13]](#footnote-13) [[14]](#footnote-14)** NOX est un système d'exploitation réseau qui offre un contrôle et une visibilité sur un réseau de commutateurs OpenFlow. Il prend en charge les applications simultanées écrites en Python et C ++, et il inclut un certain nombre d'applications de contrôleur d'échantillons.

**Beacon [[15]](#footnote-15):** Beacon est un contrôleur OpenFlow Java extensible. Il a été construit sur une structure open service gateway initiative (OSGI)[[16]](#footnote-16), permettant de créer / arrêter / actualiser / installer les applications OpenFlow sur la plate-forme au moment de l'exécution, sans déconnecter les commutateurs.

**Helios [[17]](#footnote-17):** Helios est un contrôleur OpenFlow extensible basé sur C et construit par NEC, ciblant les chercheurs. Il fournit également un shell programmatique pour effectuer des expériences intégrées.

**NEC ProgrammableFlow[[18]](#footnote-18) :** le flux programmable de NEC automatise et simplifie l'administration du réseau pour une meilleure agilité métier et fournit une interface programmable à l'échelle du réseau pour unifier le déploiement et la gestion des services réseau avec le reste de l'infrastructure informatique. Programmable Flow prend en charge OpenFlow 1.3 et 1.0, et a été le premier à être certifié par Open Networking Foundation.

**Contrôleur SDN Brocade:** Le contrôleur SDN Brocade est basé sur la sortie Helium d'OpenDaylight, annoncée en septembre 2014. Il intègre l'orchestration OpenStack.

**BigSwitch [[19]](#footnote-19):** BigSwitch a lancé un contrôleur à source fermée basé sur Beacon qui cible les réseaux d'entreprise de production. Il dispose d'une interface CLI conviviale pour la gestion centralisée de votre réseau.

**SNAC:** SNAC est un contrôleur ciblant les réseaux d'entreprise de production. Il est basé sur NOX 0.4 et propose un langage de définition de politique flexible et une interface conviviale pour configurer les périphériques et surveiller les événements.

**Maestro:** Maestro est un contrôleur OpenFlow Java extensible publié par Rice University. Il supporte le multi-threading et cible les chercheurs.

#### 3.NETCONF/YANG

Le protocole de configuration réseau (NETCONF) est basé sur le protocole XML. Il a un rôle d’organisateur de la configuration des équipements du réseau. Il fournit les mécanismes pour installer, manipuler et supprimer les configurations des périphériques réseau.[[20]](#footnote-20) [[21]](#footnote-21)

YANG (Yang Another Next Generation) est un langage de modélisation de données utilisé pour la définition des données d'état manipulées par le protocole de configuration réseau (NETCONF), les appels de procédure distante NETCONF et les notifications NETCONF.[[22]](#footnote-22)[[23]](#footnote-23)

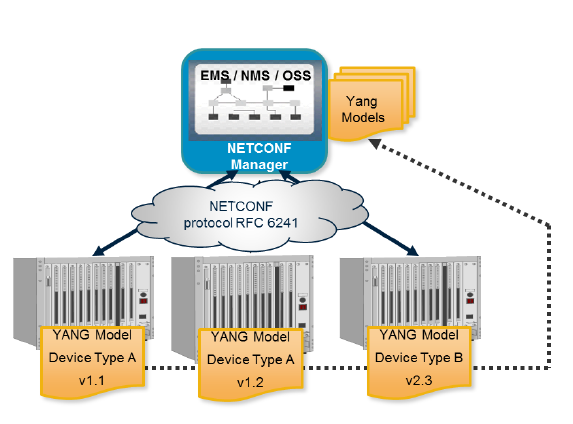


Figure Représentation du protocole NETCONF et du modèle YANG. [[24]](#footnote-24)

Le diagramme ci-dessus représente le manager NETCONF qui envoie ces instructions en utilisant le protocole NETCONF aux appareils. SNMP est beaucoup utilisé.

De plus on constate Le Modèle Yang décrit toute la configuration, le moniteur, les actions des administrateurs, les notifications pour chaque dispositif.

#### 

7.NETCONF

couche application et couche contrôleur

## III. Mise en oeuvre d’un SDN avec openFlow

Configuration du pc



Figure Configuration PC

Installation en ligne de commande

Installation de mininet

Mininet est un émulateur de réseau.

Mininet permettra aux utilisateurs de créer différents type de typologie de réseau avec différents hôtes and switches. Le type de switch peut également être choisi.

Etape 1 : Ouverture du terminal

Etape 2 : sudo apt-get install mininet



Figure installation mininet

Description paquets :

* **cgroup-bin** : il permet de faciliter la mise à niveau à partir des anciens paquets cgroup-bin vers le nouveau paquet cgroup-tools source : <https://packages.debian.org/fr/sid/cgroup-bin>
* **cgroup-lite** : package léger pour configurer les groupes de contrôle au démarrage. source : https://packages.ubuntu.com/fr/xenial/cgroup-lite
* **cgroup-tools** : Les groupes de contrôle (cgroups) fournissent un mécanisme d'agrégation/ensembles de partitionnement des tâches et tous leurs futurs fils en groupes hiérarchiques avec des comportements spécialisés. La bibliothèque libcgroup permet de manipuler, contrôler, administrer et surveiller les groupes de contrôle et les contrôleurs associés. Ce paquet contient les outils en ligne de commande.

source : <https://packages.debian.org/fr/jessie/cgroup-tools>

* **iperf :** logiciel informatique permettant la mesure de différents variables d’une connexion réseau IP (exemple : bande passante) entre deux éléments réseau, on utilise iperf en client sur une machine et en serveur sur une autre. Le client iperf va uploader les données Le serveur iperf va downloader les données. source : https://doc.ubuntu-fr.org/iperf
* **libcgroup1** : Les groupes de contrôle (cgroups) fournissent un mécanisme pour agréger / partitionner des ensembles de tâches, et tous leurs futurs enfants, en groupes hiérarchiques avec un comportement spécialisé.  
  libcgroup permet de manipuler, contrôler, administrer et surveiller les groupes de contrôle et les contrôleurs associés.  
  Ce paquet contient la bibliothèque partagée. source : <https://packages.debian.org/fr/sid/libcgroup1>
* **net-tools** : Ce paquet contient des outils importants pour contrôler le sous-système réseau du noyau Linux. Cela inclut arp, ifconfig, netstat, rarp, nameif et route. De plus, ce paquet contient des utilitaires qui touchent à la configuration d'équipements réseaux spécifiques (plipconfig, slattach, mii-tool) ainsi que des outils de configuration IP avancés (iptunnel, ipmaddr). Dans le paquet d'origine, les utilitaires hostname et autres sont inclus. Ceux-ci ne sont pas installés par ce paquet, puisqu'ils font partie d'un paquet spécifique « hostname\*.deb ». source : <https://packages.debian.org/fr/jessie/net-tools>
* **openvswitch-common** : Open vSwitch est un commutateur virtuel Ethernet multicouche basé sur logiciel, de qualité production. Il est conçu pour permettre une automatisation massive du réseau grâce à l'extension programmatique, tout en prenant en charge les interfaces de gestion standard et les protocoles (par exemple NetFlow, IPFIX, sFlow, SPAN, RSPAN, CLI, LACP, 802.1ag). En outre, il est conçu pour prendre en charge la distribution sur plusieurs serveurs physiques similaires au vswitch distribué de VMware ou au Nexus 1000V de Cisco.  
  openvswitch-switch fournit les composants et les utilitaires de l'espace utilisateur pour le commutateur basé sur le noyau Open vSwitch. source : <https://packages.debian.org/fr/jessie/openvswitch-switch>
* **socat :** Socat (pour SOcket CAT) établit deux flux d'octets bidirectionnels et transfère les données entre eux. Les canaux de données peuvent être des fichiers, des tuyaux, des appareils (terminal ou modem, etc.) ou des sockets (Unix, IPv4, IPv6, raw, UDP, TCP, SSL). Il fournit le forking, la journalisation et le traçage, différents modes pour la communication interprocessus et beaucoup plus d'options. Il peut être utilisé, par exemple, en tant que relais TCP (one-shot ou démon), en tant que socksifier externe, comme interface shell pour les sockets Unix, en tant que relais IPv6, en tant que netcat et rinetd de remplacement, pour rediriger vers TCP programmes à une ligne série, ou pour établir un environnement relativement sécurisé (su et chroot) pour l'exécution de scripts shell client ou serveur à l'intérieur de connexions réseau. source : <https://packages.debian.org/fr/jessie/socat>

Etape 3 : sudo mn -c

Permet d’invoquer le contrôleur. Mininet permet d’invoquer différents controleurs dont le controleur de référence d’openflow, l’open vswitch’s et le controleur NOX.

Il est donc possible avec les commandes suivantes de faire une invocation spécifique :

* sudo mn --controller ref
* sudo mn --controller ovsc
* sudo mn --controller NOX,pyswitch

Chacun des exemples utilisent un controleur qui transforme vos commutateurs OVS en ponts Ethernet. OVS ne supporte que 16 commutateurs. source <http://mininet.org/blog/2013/06/03/automating-controller-startup/>

<http://mininet.org/walkthrough/>

Il est également possible d’utiliser un autre contrôleur comme POX.

Dans un autre terminal il est possible de rentrer la commande suivante :

* sudo mn --controller remote,ip=127.0.0.1
* ou sudo mn --controller remote si le contrôleur est en local.

Etape 4 : installation de git

git est un logiciel de versioning permettant de sauvegarder sur un serveur des données venant de son pc local. source : <https://git-scm.com/>

* sudo apt-get install git

Etape 5 : revenir au repertoire mininet

cd mininet

Etape 6 : taper la commande :

git tag # list available versions

Permet de lister l’ensemble des versions de tag disponibles.

Etape 7 : git checkout -b cs244-spring-2012-final

permet de créer une nouvelle branche : source : <https://git-scm.com/docs/git-checkout>

Etape 8 : revenir à la racine

taper : mininet/util/install.sh [option]

Cette commande permet d’obtenir la version du système d’exploitation dont on aura besoin pour la suite.



Figure obtenir la version du système d'exploitation

Etape 9 :

taper : mininet/util/install.sh Ubuntu 17.10 artful amd64.

J’ai ajouté à la commande la version de l’OS du pc.

Cette commande à pour effet d’installer Mininet-compatible ainsi que les dépendances de mininet. Si certaine dépendance sont déjà installé. Le système d’installation s’assure que le l’os est bien la dernière version disponible.

A la fin de l’installation des dépendances.



Figure Processus de dépendances pour mininet

L’installation de Openflow débute avec le téléchargement ou la mise à jour des dépendances qu’Openflow utilise.

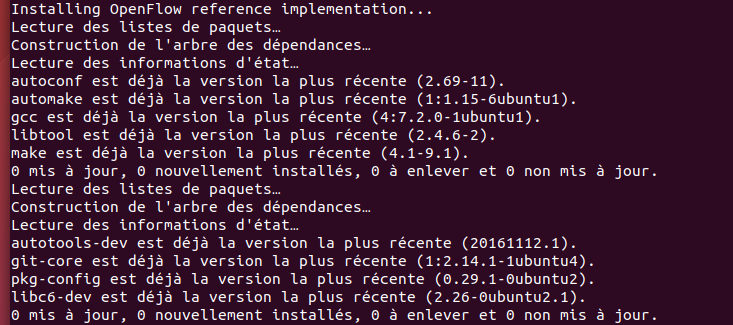


Figure Installation d'OpenFlow

Etape 10 :

Ouverture de wireshark

sudo wireshark &

Etape 11 : Création d’un réseau

sudo mn

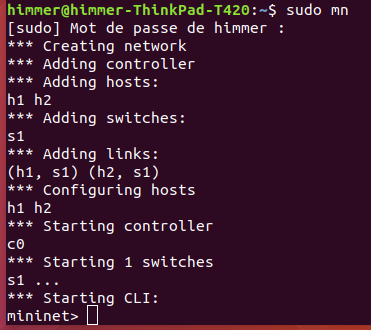


Figure Création d'un réseau

Etape 12 : créer un ping entre deux hosts

taper h1 ping h2

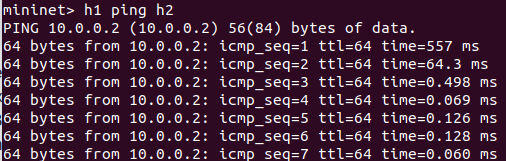


Figure H1 ping h2

On peut alors sur à l’aide de wireshark sniffer le réseau afin de voir les paquets entre h1 et h2.

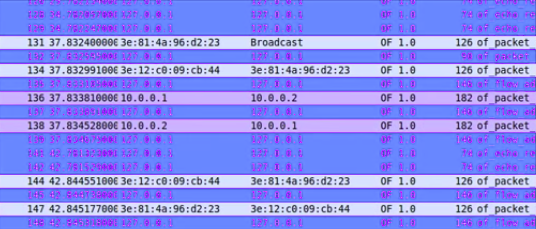


Figure wireshark lecture des paquets entre hosts

#### Création de différente typologie de réseau à l’aide de mininet

source :

<http://sdnhub.org/resources/useful-mininet-setups/>

<http://mininet.org/walkthrough/>

#### Configuration d’un réseau SDN avec un contrôleur OpenFlow et un seul switch.

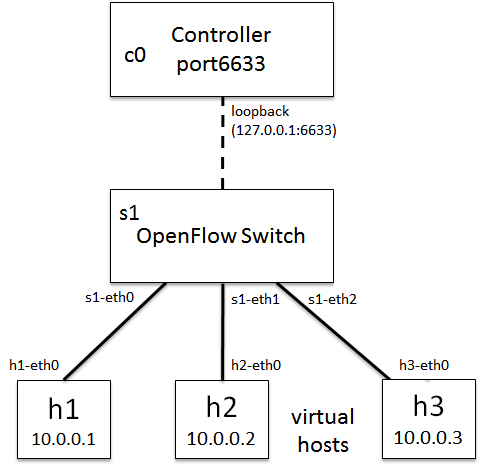


Figure Représentation d’une topologie avec un switch openFlow et trois hôtes virtuel.

La commande suivante génère un seul commutateur avec 3 hôtes attachés à celui-ci. Les hôtes recevront des adresses IP statiques et des adresses MAC.

sudo mn --arp --topo single,3 --mac --switch ovsk --controller remote

* -mac : définition automatique des adresses MAC
* -arp : rempli les entrées ARP statiques de chaque hôte les uns dans les autres.
* -switch : ovsk fait référence au noyau OVS
* -controller : la télécommande peut prendre l’adresse IP et le numéro de port comme options.

#### Configuration d’un réseau SDN avec deux switch.

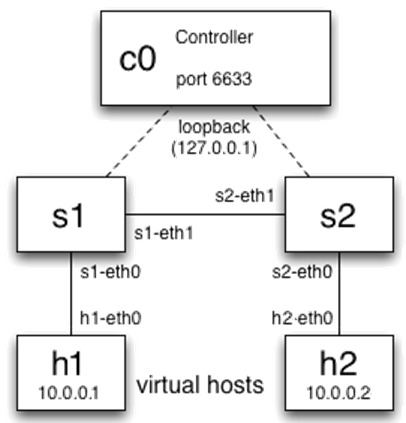


Figure Représentation d’un réseau SDN avec deux commutateurs liés et deux hôtes virtuels.

La commande suivante génère deux commutateurs connectés l'un à l'autre avec un lien et possède un hôte sur chaque commutateur. Toutes les autres options sont similaires à la dernière configuration.

sudo mn --topo linear --switch ovsk --controller remote

#### Création d’un load-balancer

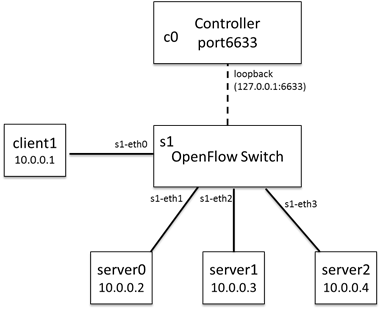


Figure Représentation d’un load-balancer avec mininet dans un réseau SDN utilisant OpenFlow

sudo mn --arp --topo single,4 --mac --switch ovsk --controller remote

**IP / MAC virtuel :** Il est nécessaire de choisir une adresse IP virtuelle (VIP) et MAC pour l'équilibreur de charge. C'est l'adresse IP à laquelle les clients vont faire une requête HTTP. Le contrôleur va pousser les règles pour réécrire le VIP avec le serveur HTTP sélectionné. Pour que cela fonctionne, il est nécessaire de définir une entrée ARP pour le VIP dans le client. Si 'h1' est le client et 10.0.0.5 est le VIP, la commande suivante ajoutera l'entrée ARP statique :

mininet> h1 arp -s 10.0.0.5 00:00:00:00:00:05

**Configuration du serveur :** Le mot-clé -arp est très important pour remplir les adresses MAC de chaque hôte. En outre, nous devons exécuter les commandes suivantes dans mininet:

mininet> h2 python -m CGIHTTPServer &

mininet> h3 python -m CGIHTTPServer &

mininet> h4 python -m CGIHTTPServer &

**Apprentissage du contrôleur :** Nous allons déployer les hôtes. Nous allons faire en sorte que le contrôleur connaisse l’emplacement de chacun des hôtes.

La commande suivante sert à cela.

mininet> pingall

**Demande de client :** Dans la VM personnalisée, il y a un script CGI configuré pour rendre compte du serveur qui gère une requête client particulière. Ainsi, lorsqu'un client exécute la commande suivante, vous recevrez l'adresse IP du serveur de gestion.

mininet> h1 curl http://10.0.0.5:8000/cgi-bin/serverip.cgi

## 

## Conclusion

Le SDN est pour l’instant réservé aux grands opérateurs télécom ou de cloud computing, mais son adoption est relativement rapide. Tous les éditeurs et constructeurs réseau proposent tous une solution SDN.

Le risque pour les constructeurs réseaux est de voir arriver massivement d’un côté des “bare-metal switches”, des switches génériques à bas coût vendus sans système d’exploitation ou bien un OS basic avec OpenFlow et de l’autre des vendeurs de solutions purement logicielles pour piloter ces équipements. Cela pourrait avoir un impact sur le business model de constructeurs qui vendent logiciel et matériels de façon liée (avec une marge autour de 60% aujourd’hui), et ce ne sera pas sans rappeler l’histoire de Sun qui vendait Solaris sur des machines SPARC avant la démocratisation des processeur x86. source : <https://www.randco.fr/actualites/2014/sdn-et-openflow/>

## Annexes

## 

## Bibliographie

### A. Ouvrages

### B. Sites Internet

<https://www.opennetworking.org/sdn-definition/>

<https://fr.wikipedia.org/wiki/R%C3%A9seau_informatique#Protocoles_et_services>

<https://www.youtube.com/watch?v=ds4VP0Ibc_M&t=310s>

<https://en.wikipedia.org/wiki/Software-defined_networking#Applications>

<http://sdnhub.org/resources/useful-mininet-setups/>

<http://mininet.org/walkthrough/>

### C. Documents audiovisuels

<https://www.youtube.com/watch?v=UwYYvT7DBvg>

<https://www.youtube.com/watch?feature=player_detailpage&v=eXsCQdshMr4&t=168>

<https://www.youtube.com/watch?v=a8_v4GZ-roY>

### 

### Glossaire

### 

### SDN : Software Defined Networking

### Table des illustrations

[Figure 1 Réseau informatique sans Software Defined Networking 5](#_Toc509773624)

[Figure 2 Contrôleur SDN 7](#_Toc509773625)

[Figure 3 SDN couches et architecture : source : (Layers&Architecture-technologie-SDN.pdf p8 10](#_Toc509773626)

[Figure 4 : SDN Infrastructure Architecture : switching devices are connected to formulate a mesh topology via various transmission media, including copper wires, wireless radio and optical fibre (source : survey on sdn.pdf fig2)( Xia, Wenfeng, et al. "A survey on software-defined networking." IEEE Communications Surveys & Tutorials 17.1 (2015):) 11](#_Toc509773627)

[Figure 5 : Fonctionnement entre la couche application et la couche contrôleur. 13](#_Toc509773628)

[Figure 6 OpenFow Switch Communication dans un réseau SDN 15](#_Toc509773629)

[Figure 7 Représentation du protocole NETCONF et du modèle YANG. 17](#_Toc509773630)

[Figure 8 Configuration PC 18](#_Toc509773631)

[Figure 9 installation mininet 19](#_Toc509773632)

[Figure 10 obtenir la version du système d'exploitation 21](#_Toc509773633)

[Figure 11 Processus de dépendances pour mininet 21](#_Toc509773634)

[Figure 12 Installation d'OpenFlow 22](#_Toc509773635)

[Figure 13 Création d'un réseau 22](#_Toc509773636)

[Figure 14 H1 ping h2 23](#_Toc509773637)

[Figure 15 wireshark lecture des paquets entre hosts 23](#_Toc509773638)

[Figure 16 Représentation d’une topologie avec un switch openFlow et trois hôtes virtuel. 24](#_Toc509773639)

[Figure 17 Représentation d’un réseau SDN avec deux commutateurs liés et deux hôtes virtuels. 25](#_Toc509773640)

[Figure 18 Représentation d’un load-balancer avec mininet dans un réseau SDN utilisant OpenFlow 26](#_Toc509773641)

References

Réseau informatique. (2018, February 24). Retrieved from https://fr.wikipedia.org/wiki/Réseau\_informatique

1. (n.d.). Réseau informatique — Wikipédia. Consulté le février 25, 2018 à l'adresse <https://fr.wikipedia.org/wiki/R%C3%A9seau_informatique> [↑](#footnote-ref-1)
2. "1re STG LES RESEAUX INFORMATIQUES - ppt video ... - SlidePlayer." <http://slideplayer.fr/slide/448498/>. Date de consultation : 16 mars. 2018. [↑](#footnote-ref-2)
3. "A Survey on Software-Defined Networking - IEEE Journals & Magazine." <http://ieeexplore.ieee.org/document/6834762/>. Date de consultation : 19 mars. 2018. [↑](#footnote-ref-3)
4. "dblp: IEEE Communications Surveys and Tutorials, Volume 17.page37" <http://dblp1.uni-trier.de/db/journals/comsur/comsur17.html>. Date de consultation : 20 mars. 2018. [↑](#footnote-ref-4)
5. "dblp: IEEE Communications Surveys and Tutorials, Volume 17. figure 2" <http://dblp1.uni-trier.de/db/journals/comsur/comsur17.html>. Date de consultation : 20 mars. 2018. [↑](#footnote-ref-5)
6. "dblp: IEEE Communications Surveys and Tutorials, Volume 17. page 37 section C 1" <http://dblp1.uni-trier.de/db/journals/comsur/comsur17.html>. Date de consultation : 20 mars. 2018. [↑](#footnote-ref-6)
7. "A Survey and a Layered Taxonomy of Software-Defined Networking." <http://www.hit.bme.hu/~jakab/edu/litr/SDN/SDN_Taxonomy_06805151.pdf>. Date de consultation : 20 mars. 2018. [↑](#footnote-ref-7)
8. (n.d.). Lightweight DDoS Flooding Attack Detection Using NOX/OpenFlow. Consulté le février 25, 2018 à l'adresse <https://pdfs.semanticscholar.org/deee/c803eaceec8cd5254c64e73b67959c5e7670.pdf> [↑](#footnote-ref-8)
9. (2014, octobre 10). File:SDN-architecture-overview-transparent.png - Wikimedia Commons. Consulté le février 25, 2018 à l'adresse <https://commons.wikimedia.org/wiki/File:SDN-architecture-overview-transparent.png> [↑](#footnote-ref-9)
10. (n.d.). SDN applications - The intent-based Northbound Interface realisation .... Consulté le février 25, 2018 à l'adresse <http://ieeexplore.ieee.org/document/7502469/> [↑](#footnote-ref-10)
11. "Scalable software defined network controllers - ACM Digital Library." 13 août. 2012, <http://dl.acm.org/citation.cfm?id=2342356.2342414>. Date de consultation : 20 mars. 2018. [↑](#footnote-ref-11)
12. "What is an OpenFlow Controller? - SDxCentral." <https://www.sdxcentral.com/sdn/definitions/sdn-controllers/openflow-controller/>. Date de consultation : 16 mars. 2018. [↑](#footnote-ref-12)
13. Tavakoli, Arsalan, et al. "Applying NOX to the Datacenter." *HotNets*. 2009. [↑](#footnote-ref-13)
14. Gude, Natasha, et al. "NOX: towards an operating system for networks." *ACM SIGCOMM Computer Communication Review* 38.3 (2008): 105-110. [↑](#footnote-ref-14)
15. "Software-Defined Networking: A Comprehensive Survey - arXiv." 31 mai. 2014, <https://arxiv.org/pdf/1406.0440>. Date de consultation : 5 mars. 2018. [↑](#footnote-ref-15)
16. Hall, Richard S., and Humberto Cervantes. "An OSGi implementation and experience report." *Consumer Communications and Networking Conference, 2004. CCNC 2004. First IEEE*. IEEE, 2004. [↑](#footnote-ref-16)
17. "All-Optical Packet/Circuit Switching-Based Data ... - Semantic Scholar." <https://pdfs.semanticscholar.org/891d/a4fc2bcf7eb01fd78be6642d8173048dddbf.pdf>. Date de consultation : 5 mars. 2018. [↑](#footnote-ref-17)
18. "All-Optical Packet/Circuit Switching-Based Data ... - Semantic Scholar." <https://pdfs.semanticscholar.org/891d/a4fc2bcf7eb01fd78be6642d8173048dddbf.pdf>. Date de consultation : 5 mars. 2018. [↑](#footnote-ref-18)
19. "Abstraction in Software-Defined Networks - acm sigcomm." <http://conferences.sigcomm.org/co-next/2013/program/p13.pdf>. Date de consultation : 5 mars. 2018. [↑](#footnote-ref-19)
20. "RFC 6241 - Network Configuration Protocol (NETCONF) - IETF Tools." <https://tools.ietf.org/html/rfc6241>. Date de consultation : 11 mars. 2018. [↑](#footnote-ref-20)
21. "RFC 4742 - Using the NETCONF Configuration Protocol ... - IETF Tools." <https://tools.ietf.org/html/rfc4742>. Date de consultation : 16 mars. 2018. [↑](#footnote-ref-21)
22. "RFC 4742 - Using the NETCONF Configuration Protocol ... - IETF Tools." <https://tools.ietf.org/html/rfc4742>. Date de consultation : 11 mars. 2018. [↑](#footnote-ref-22)
23. "RFC 6020 - YANG - A data modeling language for NETCONF." <https://tools.ietf.org/html/rfc6020>. Date de consultation : 11 mars. 2018. [↑](#footnote-ref-23)
24. "[칼럼] NETCONF-YANG의 이유있는 탄생! - The Cisco Network." 13 août. 2015, <https://apjc.thecisconetwork.com/site/content/lang/ko/id/3909>. Date de consultation : 16 mars. 2018. [↑](#footnote-ref-24)