Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Centre Universitaire de Mila
Institut des Sciences et de la Technologie

Administration des Réseaux

– Chapitre 5 –

Réseaux SDN

Département MI

Objectif du cours

1. **Se faire une idée sur le domaine**
2. **Comprendre l'architecture des réseaux SDN**
3. **L'impact du SDN sur la gestion, la maintenance et la sécurisation des réseaux**

Plan du cours

1. **Introduction**
2. **Architecture traditionnelle & SDN**
3. **Composants du SDN**
    a. **Openflow et la programmabilité des réseaux**
    b. **Contrôleurs SDN**
    c. **Applications**
4. **Virtualisation des réseaux**
5. **Challenges SDN**

L'Internet… une belle histoire

- Grand succès
  - De l'expérimentation à la réalisation
- Brillance de la sous-spécification
  - Réseaux: mode « Best Effort »
  - Hôtes: application arbitraire
- Permet l'innovation au niveau applicatif
  - Web, P2P, VoIP, Réseaux sociaux, Virtualisation, …

- Mais le changement est facile seulement au niveau de la bordure du réseaux …

L'intérieur du Net … une autre histoire

- Équipement fermé (black box)
  - Le **logiciel** et fourni avec le matériel
  - **Interfaces spécifiques** constructeurs
- Sur spécification
  - Processus de standardisation très lourd
    - OSPF (RFC 2328): 245 pages
  - Complexité liée à la distribution des algorithmes (e.g. count-to-infinity)
- L'innovation est limitée à un ensemble très limité de personnes
  - Les équipementiers écrivent le code
  - Délai très important pour l'introduction d'une nouvelle fonctionnalité … ou pour corriger un bug !!!

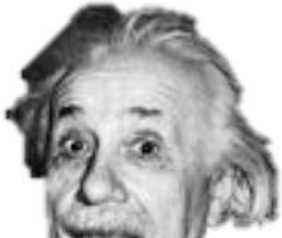
Impact sur la performance, la sécurité, la fiabilité et le coût …

Un réseau difficile à gérer …

- Les systèmes d'exploitation (OS) sont très onéreux
  - Plus de la moitié du **coût** du réseau
  - Les **erreurs humaines** sont très nombreuses

- Présence de bugs dans les équipements
  - Routeurs avec plus de **20 millions de lignes de code**
  - **Pannes** possibles, **vulnérabilités**, …
  - Les mises à jour sont très difficiles à mettre en place
    - Arrêt, nouveaux bugs, nouvelles vulnérabilités, interfaçage avec certains logiciels, …

- Problématique de la maîtrise d'équipements fermés (black box) hétérogènes inter-opérants …



Pourquoi le SDN ?

- L'idée intéresse énormément les **chercheurs**
  - Vu comme **moyen d'optimiser et de mettre à l'échelle** les réseaux ...
- L'idée est récente mais **intéresse** énormément les **industriels** et les **opérateurs**
  - Permet aux entreprises d'accélérer le **déploiement** et la **distribution** d'applications
  - Moyen de gagner de l'**argent** …
    - Réduire le **matériel** et le cout d'**administration**

Plan du cours

1. Introduction
2. **Architecture traditionnelle vs SDN**
3. Composants du SDN
    a. Openflow et la programmabilité des réseaux
    b. Contrôleurs SDN
    c. Applications
4. Virtualisation des réseaux
5. Challenges SDN

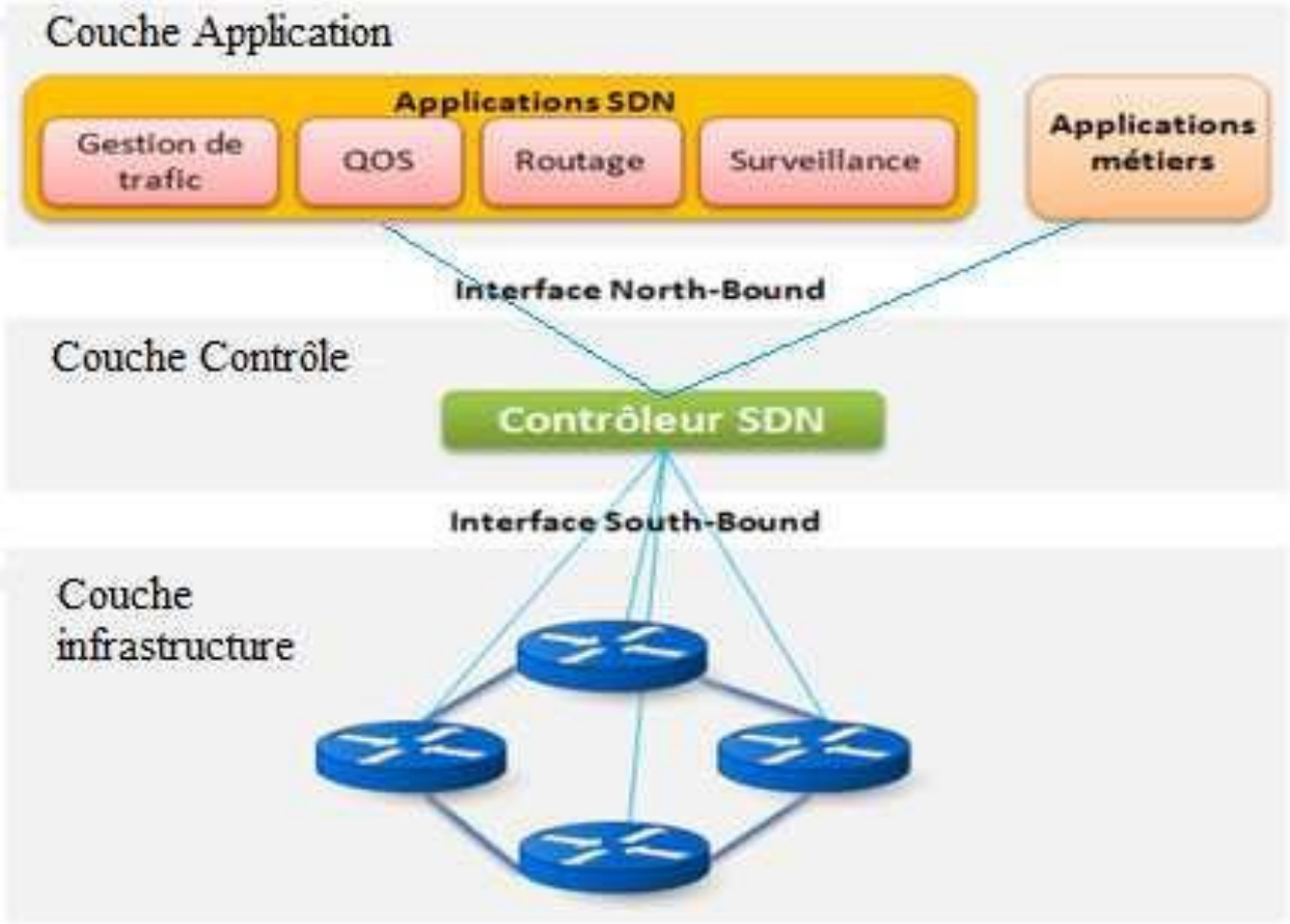
Architecture SDN (1)

Une **nouvelle architecture** réseaux :

– Facilitant la **programmabilité** des réseaux

– Avec l'idée que des **applications contrôlent**, **à distance**, des **équipements** réseaux

Architecture SDN (2)



Architecture SDN (3)

SDN contient 3 couches:

•**La couche infrastructure** :

Ici se trouve les **périphériques** qui contiennent les plans de données du réseau. Autrement dit, les **switchs SDN** responsables de **l'acheminement du trafic.**

•**La couche contrôle** :

Elle se base sur le contrôleur SDN, il contrôle le **plan de donnée** de façon réactive ou proactive en **insérant** les différentes **politiques de transfert**. Cette partie est le **cerveau** du réseau et elle englobe la plupart des opérations de calcul.

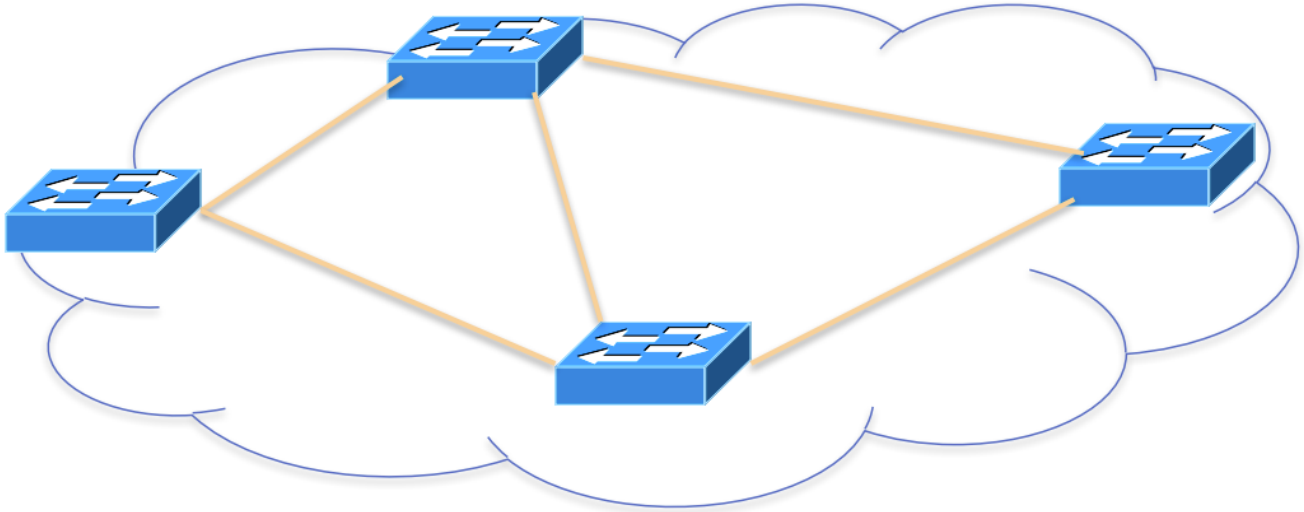
•**La couche application** :

Héberge les **applications** qui permettant d'exploiter la panoplie d'avantages qu'apporte l'architecture, en introduisant de **nouvelles fonctionnalités** réseaux.

Architecture *traditionnelle* vs *SDN*

Architecture traditionnelle

Plan données:
Diffusion de paquets



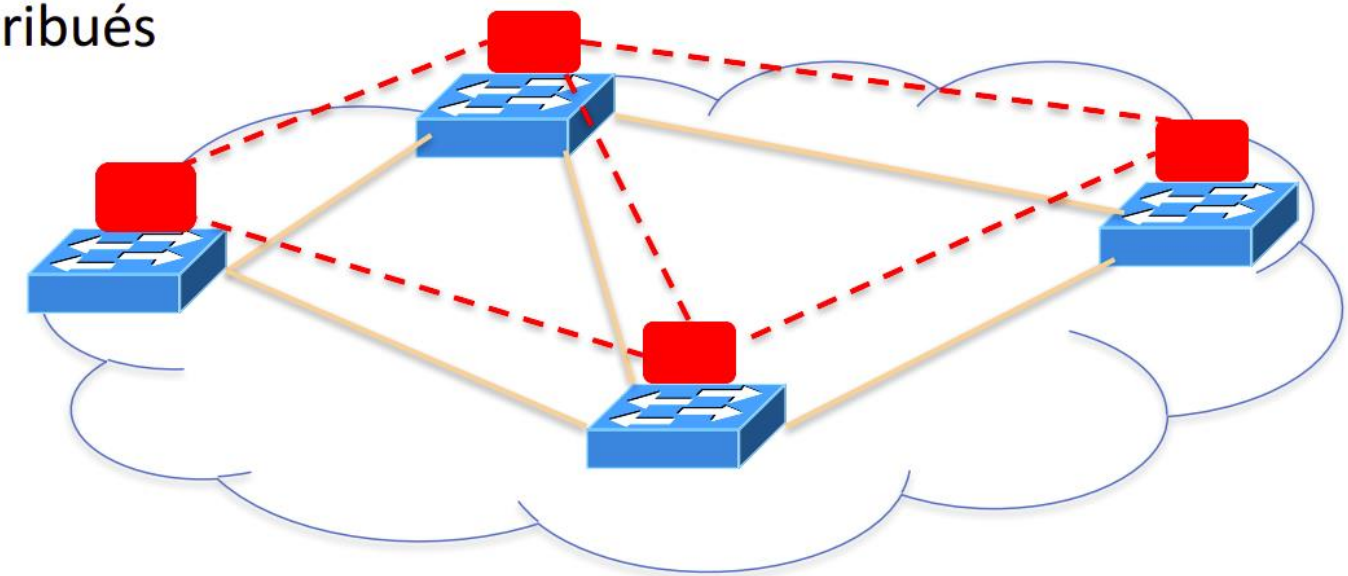
Transfert, filtrage, mise en tampon, marquage, mesures de paquets, …

Architecture ***traditionnelle*** vs *SDN*

Architecture traditionnelle

Plan contrôle:

Algorithmes distribués

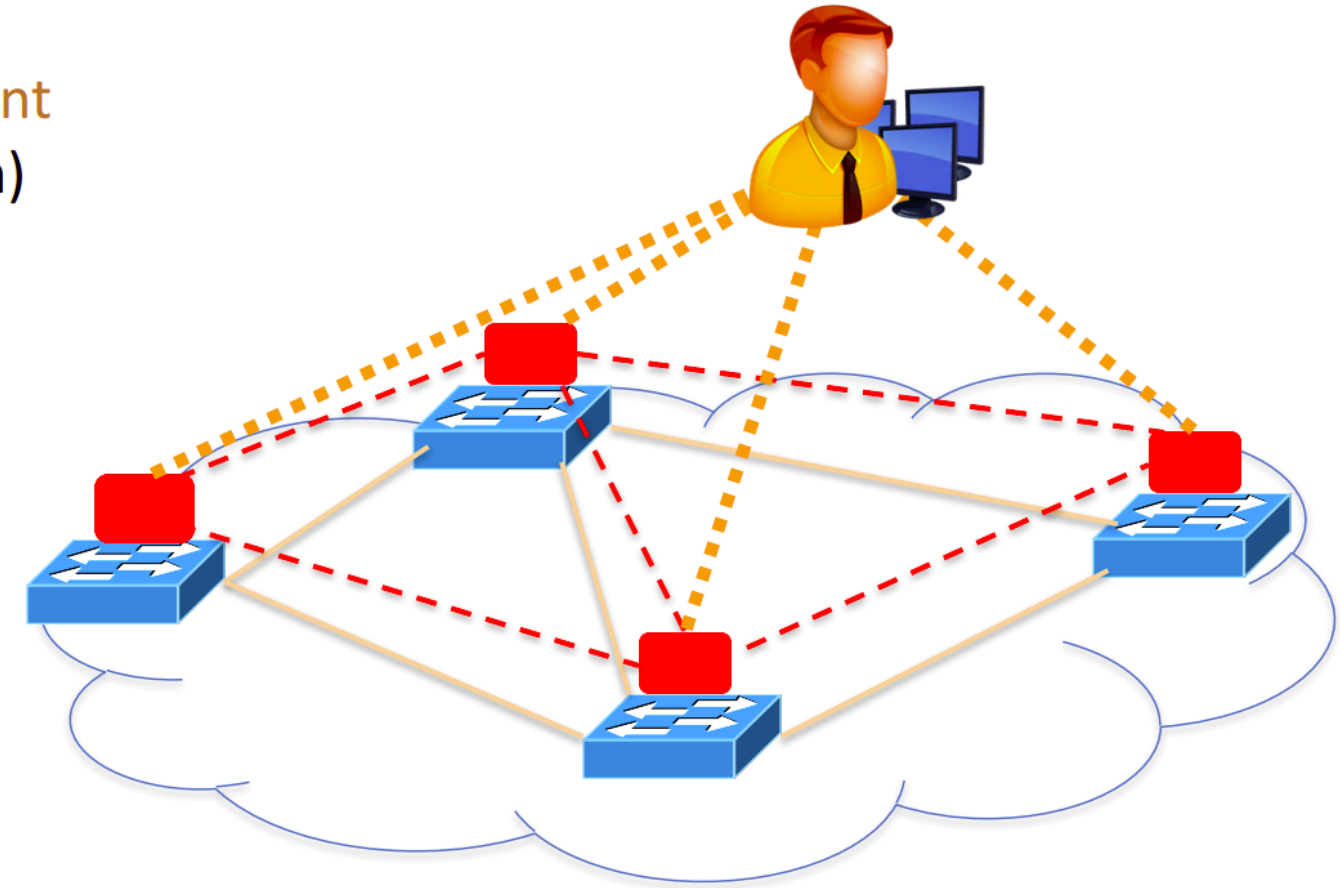


Gestion de la topologie, calcul des routes, mise en place des règles de routage, …

Architecture *traditionnelle* vs *SDN*

Architecture traditionnelle

Plan management
(Facteur humain)

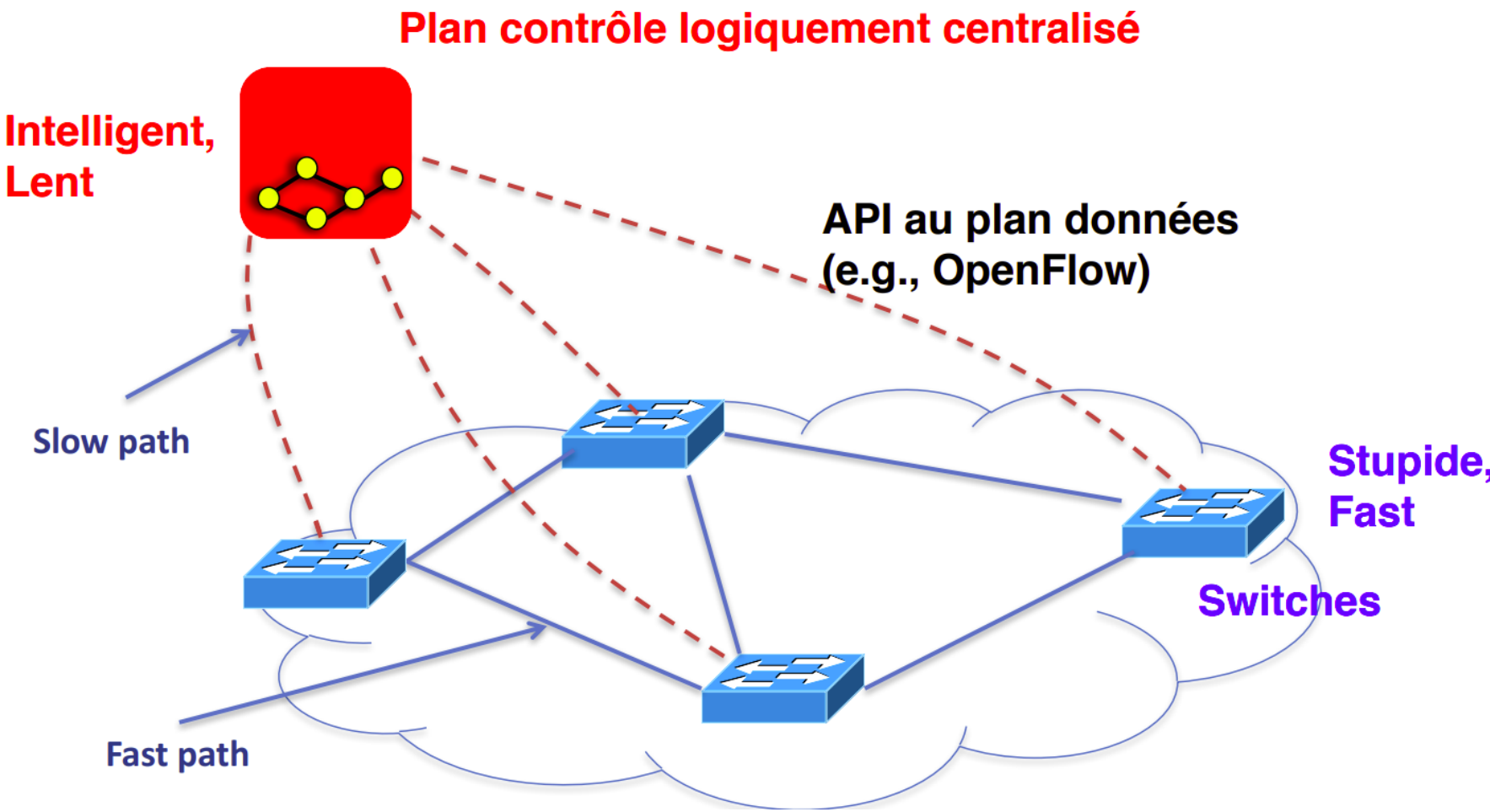


Collecte des mesures et configuration des équipements, …

Vers un plan contrôle logiquement centralisé …

- Gestion plus facile
  - Configuration des équipements
  - Vérification/débogage des contrôleurs plus facile
- Innovation facile et rapide
  - Moins de dépendances aux équipementiers et aux standards
  - Évolution du logiciel de contrôle indépendamment du hardware
- Interopération facile
  - La compatibilité est clé seulement là où il y a des protocoles
- Équipement plus simple et moins cher
  - Logiciel/OS minimaliste

SDN


Plan contrôle logiquement centralisé
Intelligent,
Lent
API au plan données
(e.g., OpenFlow)
Slow path
Stupide,
Fast
Switches
Fast path

Network OS

- Système centralisé ou distribué qui permet la création et le maintient (i.e. consistant) d'une vue du réseaux

- Utilise un protocole ouvert (e.g. **Openflow**) pour:
  - **Obtenir l'état** des éléments de forwarding
  - **Donner des directives** aux éléments de forwarding

Plan du cours

1. Introduction
2. Architecture traditionnelle vs SDN
3. **Composants du SDN**
    a. **Openflow et la programmabilité des réseaux**
    b. **Contrôleurs SDN**
    c. **Applications**
4. Virtualisation des réseaux
5. Challenges SDN

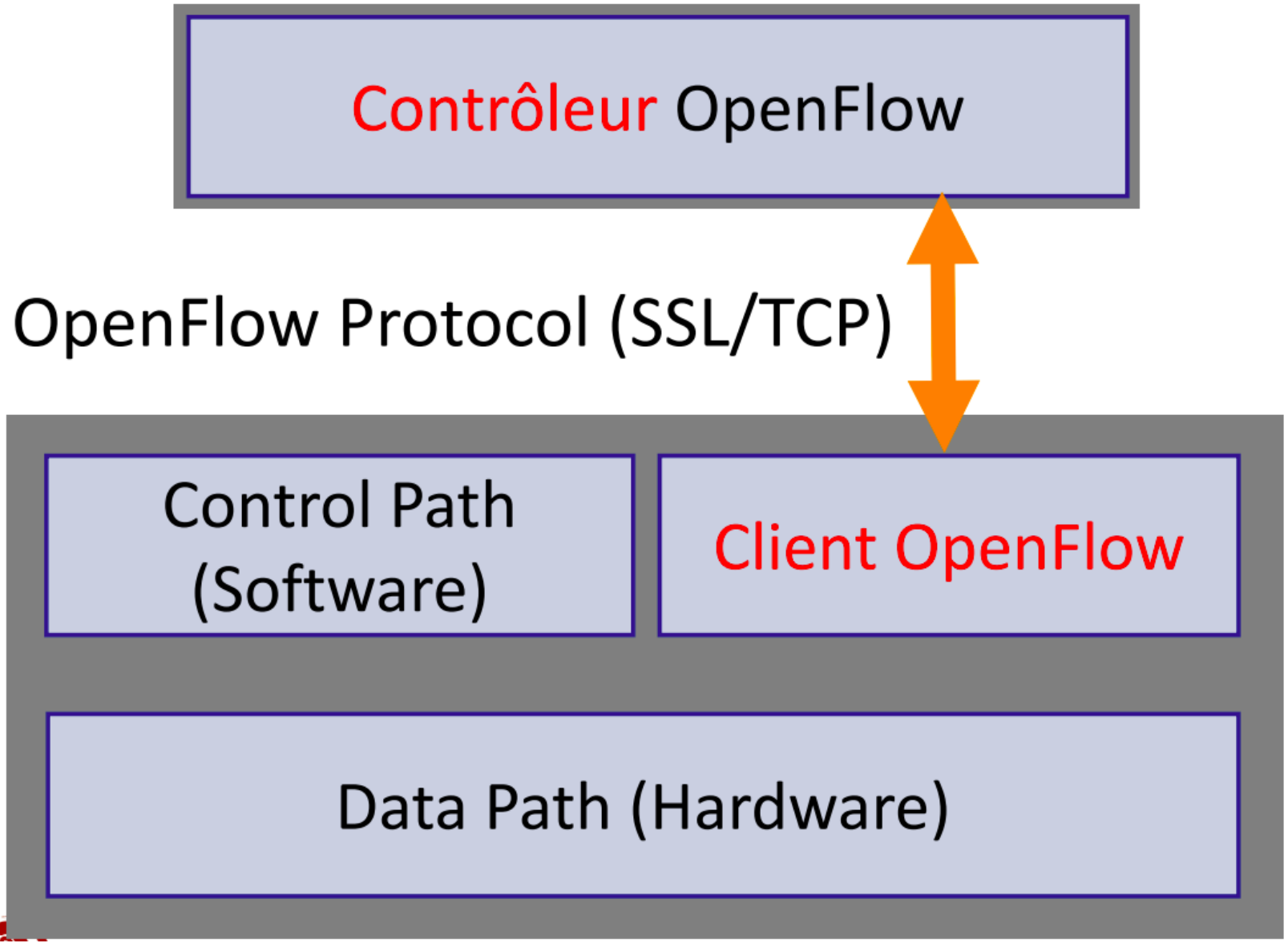
Openflow

- **Openflow** est un protocole pour le **contrôle à distance** des tables de forwarding de switches ou routeurs.

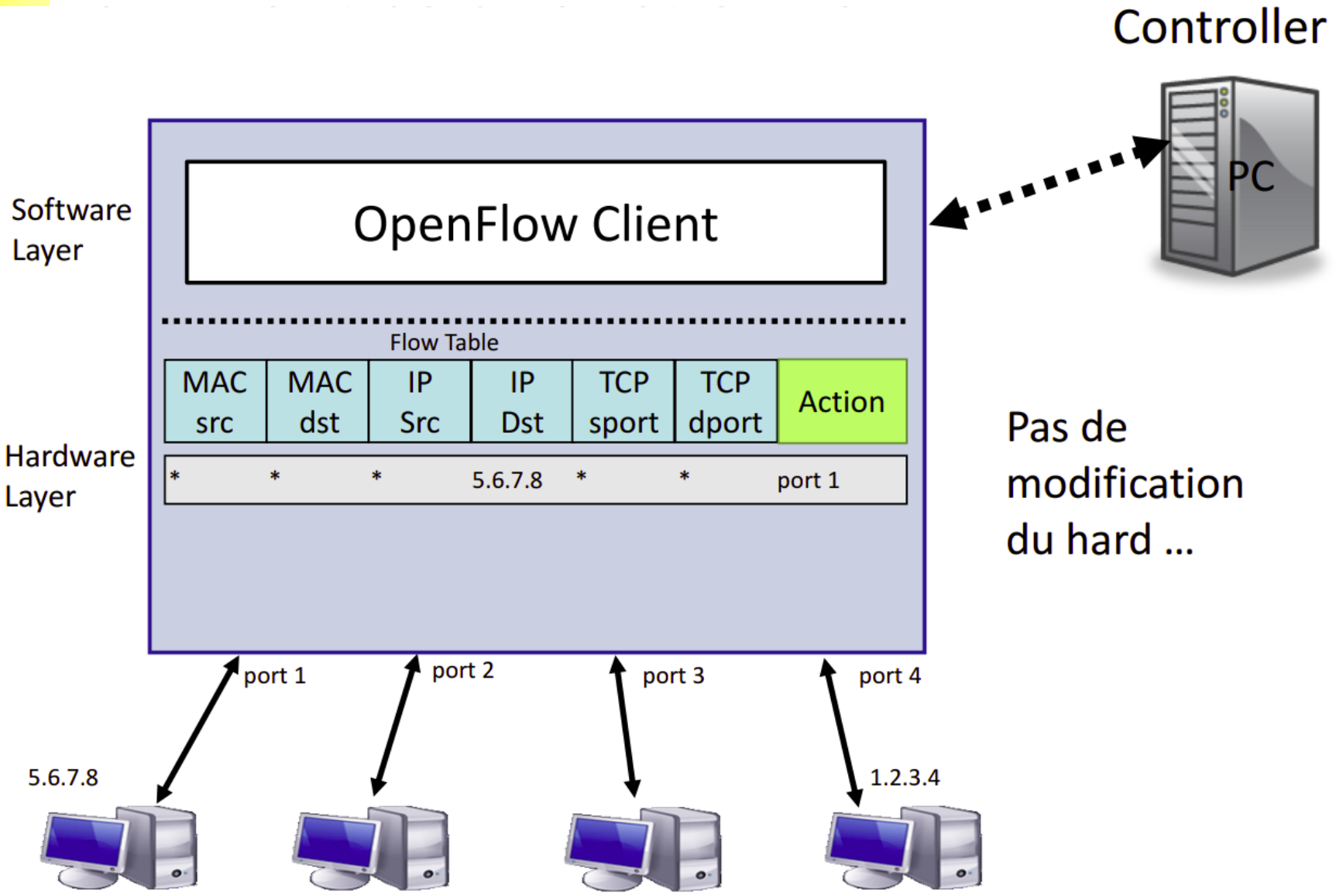
- Agit sur des **flux** non pas sur des **paquets** …
  - Un flux étant: **trafic** d'un point **A** vers un point **B**, trafic **HTTP,** …

- C'est un des éléments (optionnel) du SDN …

Comment Openflow fonctionne ?



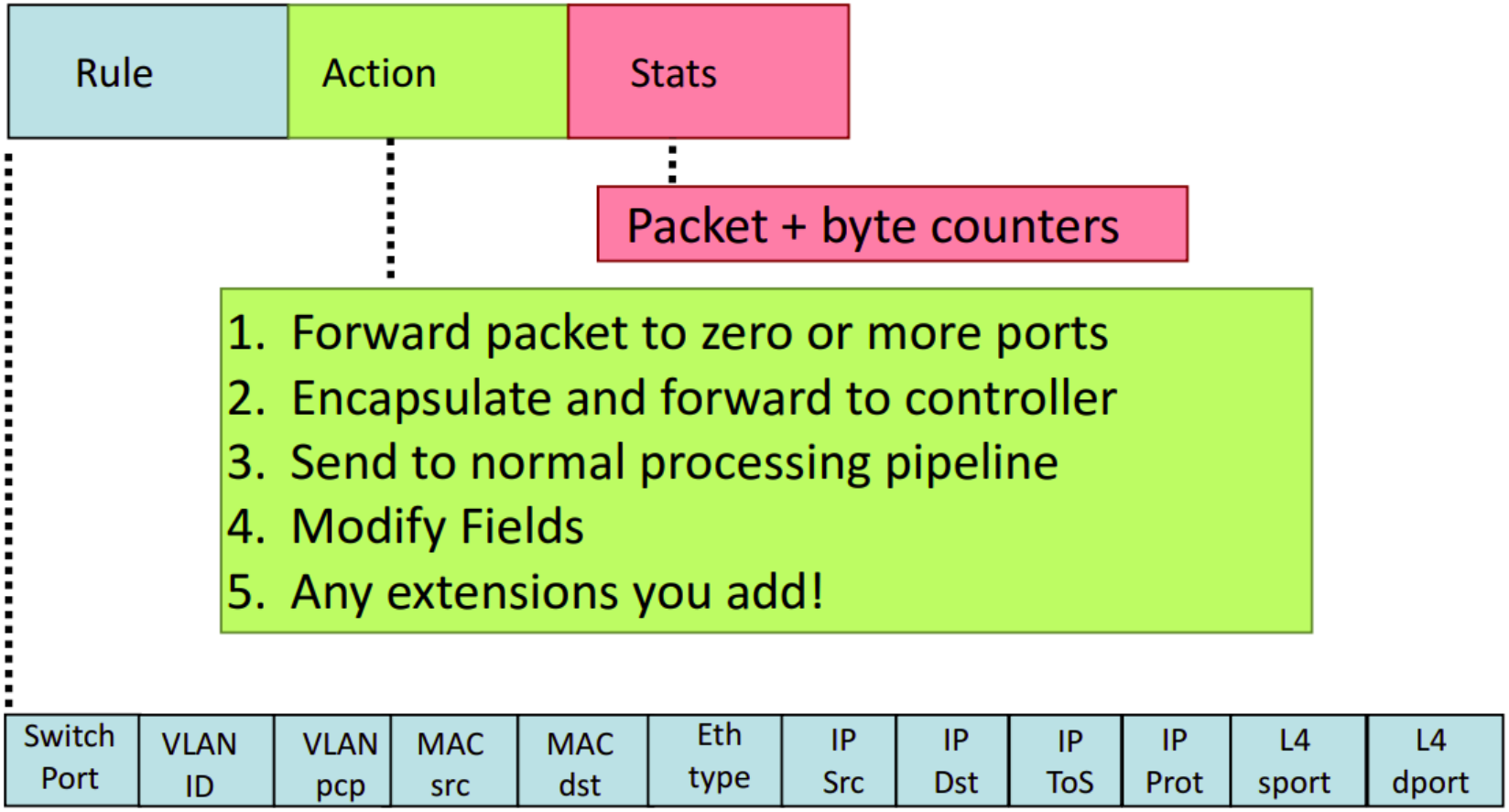
Comment Openflow fonctionne ?



Pas de modification du hard …

Bases Openflow

Les entrée des tables de flux



+ mask what fields to match

Exemple *Openflow* (1)

Switching

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
*	*	00:1f:..	*	*	*	*	*	*	*	port6

Flow Switching

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
port3	00:20..	00:1f..	0800	vlan1	1.2.3.4	5.6.7.8	4	17264	80	port6

Firewall

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
*	*	*	*	*	*	*	*	*	22	drop

Exemple *Openflow* (2)

Routing

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
*	*	*	*	*	*	5.6.7.8	*	*	*	port6

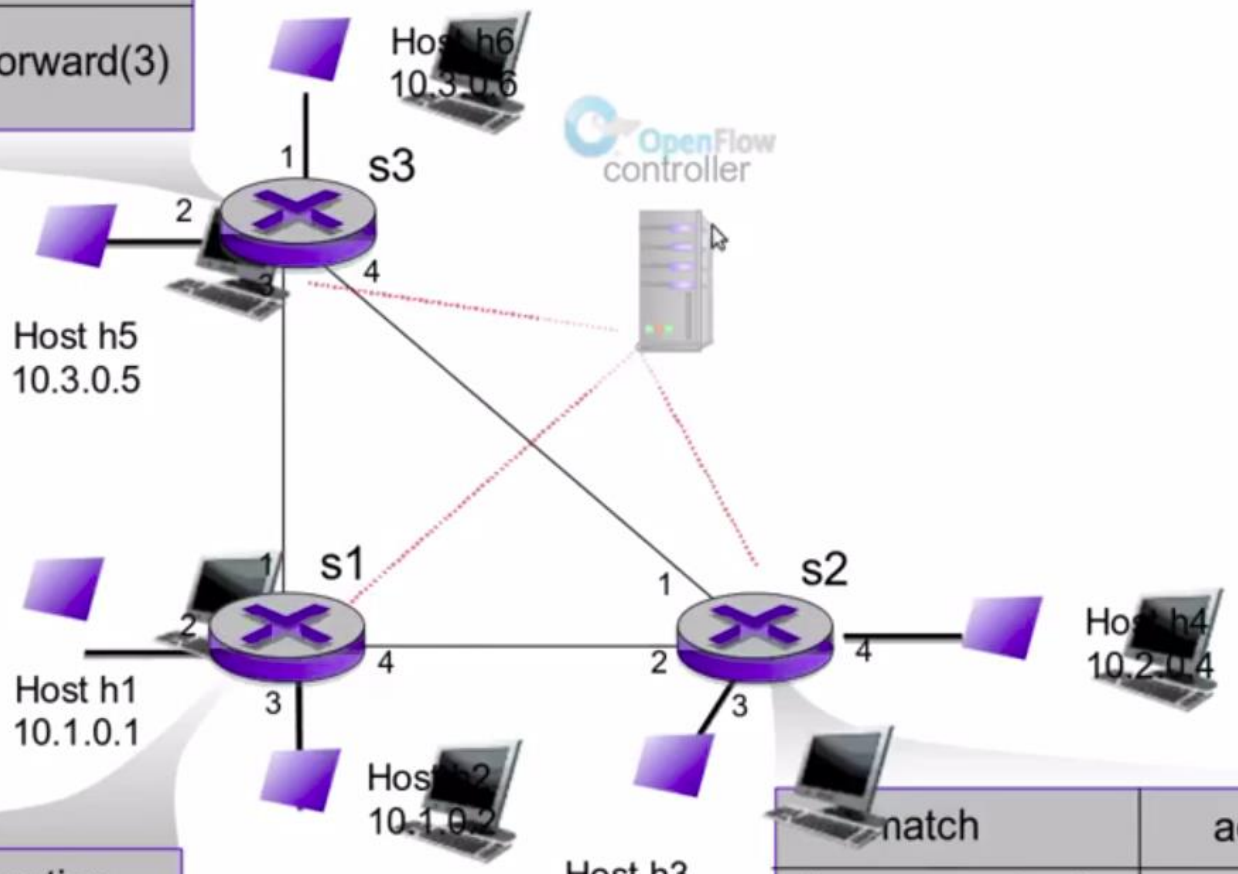
VLAN Switching

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
*	*	00:1f..	*	vlan1	*	*	*	*	*	port6, port7, port9

OpenFlow example

Example: datagrams from hosts h5 and h6 should be sent to h3 or h4, via s1 and from there to s2

match	action
IP Src = 10.3.*.* IP Dst = 10.2.*.*	forward(3)

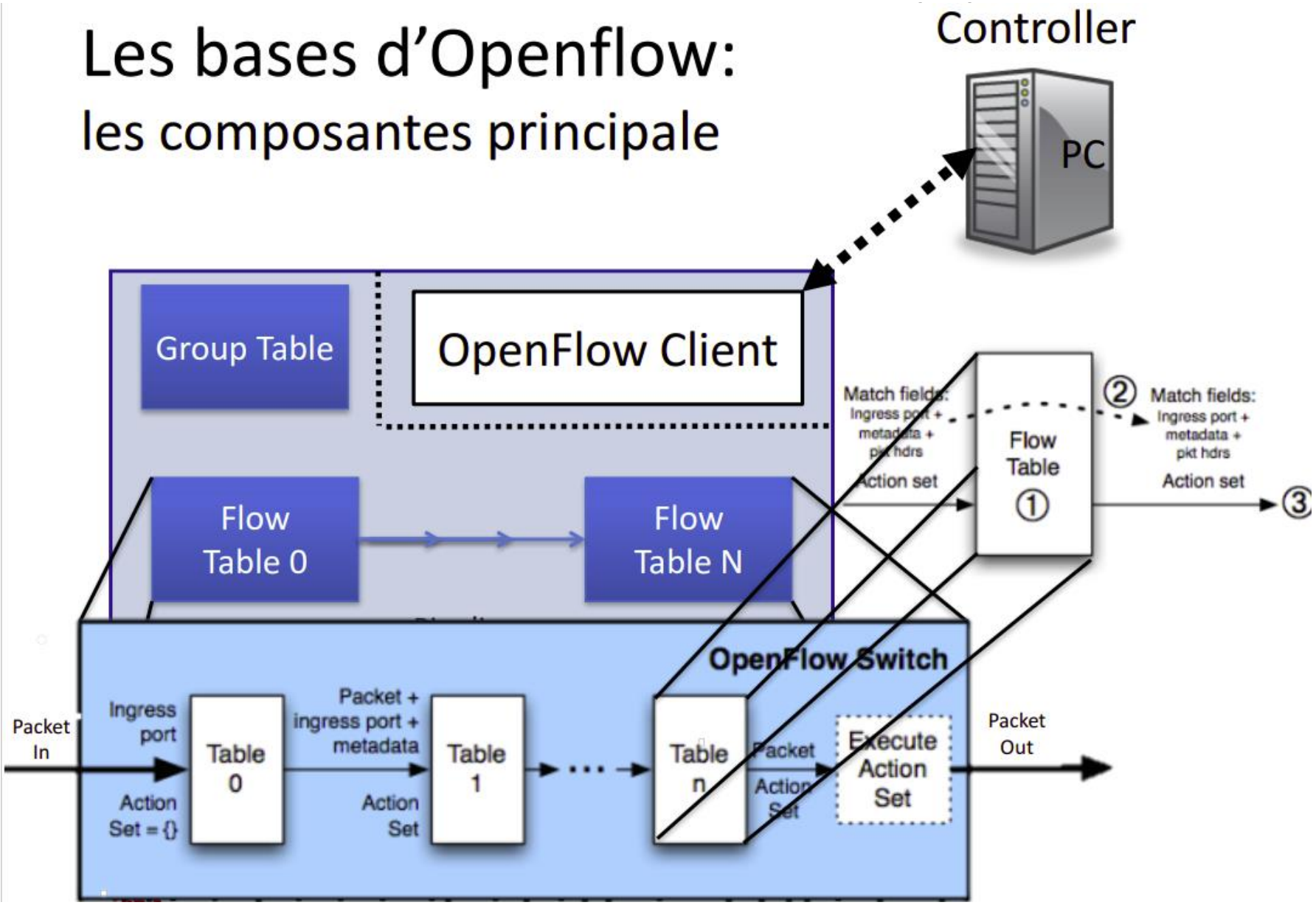


match	action
ingress port = 1 IP Src = 10.3.*.* IP Dst = 10.2.*.*	forward(4)

match	action
ingress port = 2 IP Dst = 10.2.0.3	forward(3)
ingress port = 2 IP Dst = 10.2.0.4	forward(4)

Les bases d'Openflow:

les composantes principale



Programabilité des réseaux (1)

- Openflow permet la **programmabilité** des réseaux
  - Équipement simple/disponible/interchangeable/ pas cher … « Commodity hardware »

- Existe-il d'autres outils pour le faire dans les réseaux **traditionnels** ?

Programmabilité des réseaux (2)

- **CLI à distance**
  - Difficile à mettre en place (récupération et reprise après erreur, …)
  - Équipements chers et évolution difficile
- **SNMP**
  - Pour la supervision (i.e. management)
- **NETCONF du WG NETMOD (IETF)**
- **Forces (IETF)**

Contrôleurs SDN

Plusieurs contrôleurs SDN :

- NOX/POX
- Ryu
- Floodlight
- Pyretic
- Frenetic
- RouteFlow
- Opendaylight
- ONOS

Récapitulatif sur les Contrôleurs SDN

	NOX	POX	Ryu	Floodlight	ODL
Langage	C++	Python	Python	Java	Java
Perf.	Rapide	Lent	Lent	Rapide	Rapide
OF	1.0	1.0	1.0-1.4	1.0-1.3	1.0-
Openstack	Non	Non	Oui	Oui	Oui
Maîtrise	Modérée	Facile	Modérée	Lente	Lente

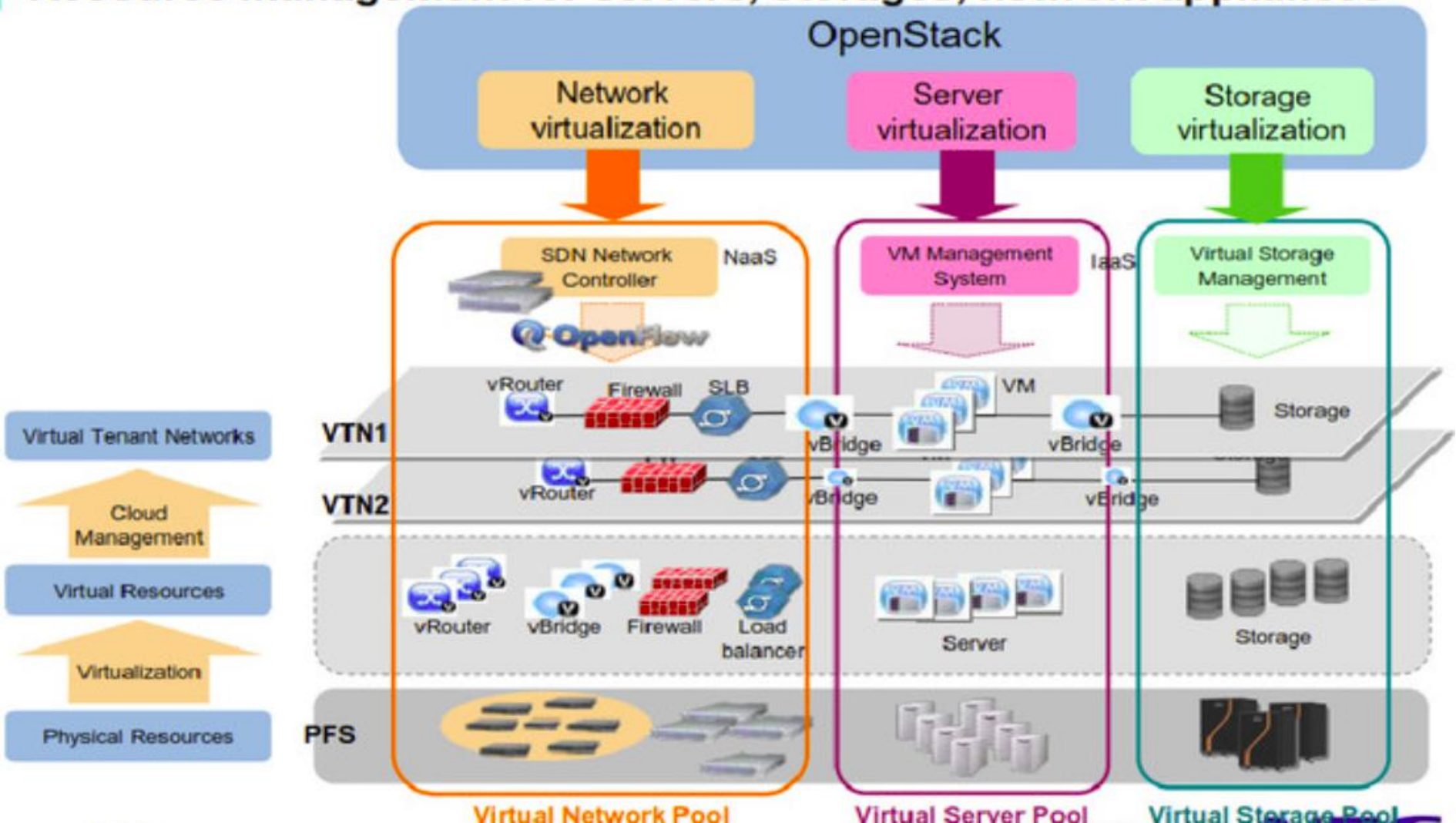
Les contrôleurs doivent offrir:

- Une interface southbound efficace
  - Permettant d'exprimer beaucoup de choses
- Une interface northbound puissante
  - Permettant d'exploiter au mieux le réseaux sans rentrer dans l'implémentation du contrôleur (e.g. REST)

Exemple Ryu avec Openstack

Agile mapping of resource pools

Resource management for servers, storages, network appliances

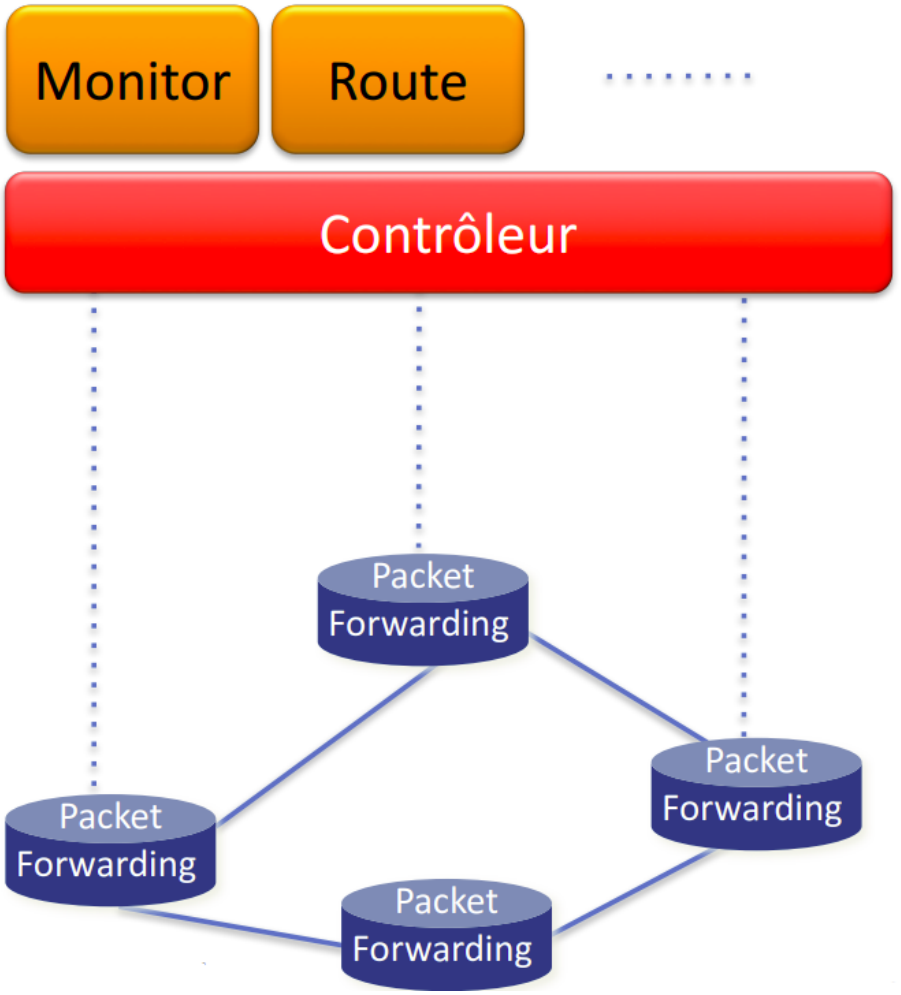


Au niveau Applicatif

Un module par tâche

Facile à programmer, à tester, à déboguer, à porter, …

Mais comment combiner les modules dans une application complète et cohérente (module affectant un même trafic)?



Plan du cours

1. Introduction
2. Architecture traditionnelle vs SDN
3. Composants du SDN
    a. Openflow et la programmabilité des réseaux
    b. Contrôleurs SDN
    c. Applications
4. **Virtualisation des réseaux**
5. Challenges SDN

Virtualisation réseaux dans **Mininet**

- Environnement **réseau virtuel** pouvant être exécuté dans un **PC**
- Exécute un **vrai Kernel**, switch et du code applicatif réel sur une même machine
  - CLI, interface python ...
- Plusieurs fonctions **Openflow** sont supportées
  - Pratique pour développer, déployer et partager
- Facile à utiliser et **open source**

Plan du cours

1. Introduction
2. Architecture traditionnelle vs SDN
3. Composants du SDN
    a. Openflow et la programmabilité des réseaux
    b. Contrôleurs SDN
    c. Applications
4. Virtualisation des réseaux
5. **Challenges SDN**

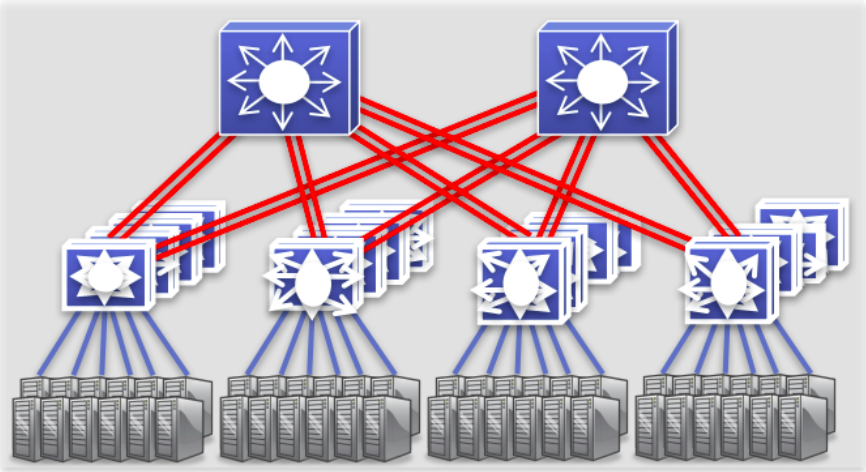
Challenges dans SDN

- Applications et API **northbound**
  - Routage **inter-domaine** (SDX), réseaux de mobiles
- Contrôle
  - Sécurité
  - Combinaison entre **Big data** et **gestion de réseaux**
  - Orchestration
  - Inconsistance possible lors de l'utilisation de plusieurs contrôleurs
- Plan données
  - Aller plus loin du match/action (détection de pattern, DPI, …)
- Programmation du plan données (e.g. Fast reroute, …)
- QoS/QoE

Bibliographie

- Cours de:
  - – Nick Feamster, Georgia Tech (Coursera)
  - – Jennifer Rexford, Princeton
  - – Nick McKeown, Stanford
  - – HADJADJ AOUL, Maître de Conférences Université de Rennes 1
- Livre SDN: Software Defined Networks Par: Thomas D. Nadeau, Ken Gray

Réduction des coûts avec SDN



Coût (CAPEX)

200,000 serveurs
Degrés de 20 ➔ 10,000 switches
$5k vendor switch = $50M
$1k commodity switch = $10M

Économie dans 1 data centers = $40M

Contrôle (OPEX)

Contrôle plus flexible
Adaptation du réseau aux services
Amélioration et innovation rapide

Moins d'administrateurs et moins d'erreurs humaines