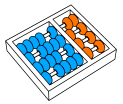
Redes Definidas por Software (SDN), OpenFlow e outros Controladores de Rede

Anderson Coelho Weller

Universidade Estadual de Campinas
Instituto de Computação

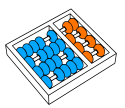
21 de outubro de 2013

Agenda



Introdução

- Redes Definidas por Software (SDN) é um novo paradigma no desenvolvimento de pesquisas em redes de computadores
- Grande interesse tanto da área acadêmica quanto industrial
- Principalmente por causa do OpenFlow
- Porém, SDN é muito mais do que OpenFlow



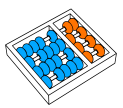
Introdução

Área de redes encontra-se em situação complexa:

- Grande sucesso da área
- Necessário estabilidade na Internet
- Pesquisas com novos protocolos tornaram-se arriscadas
- Tornando as redes pouco flexíveis

Introdução

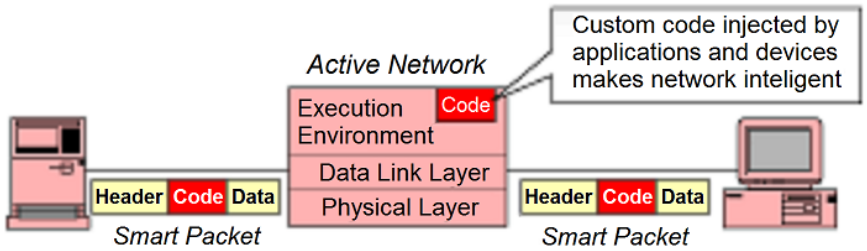
- Mesmos pesquisas como a Internet2
- Têm dificuldades em justificar a adoção em larga escala de suas novas tecnologias
- Devido ao grau de ruptura com as tecnologias atuais

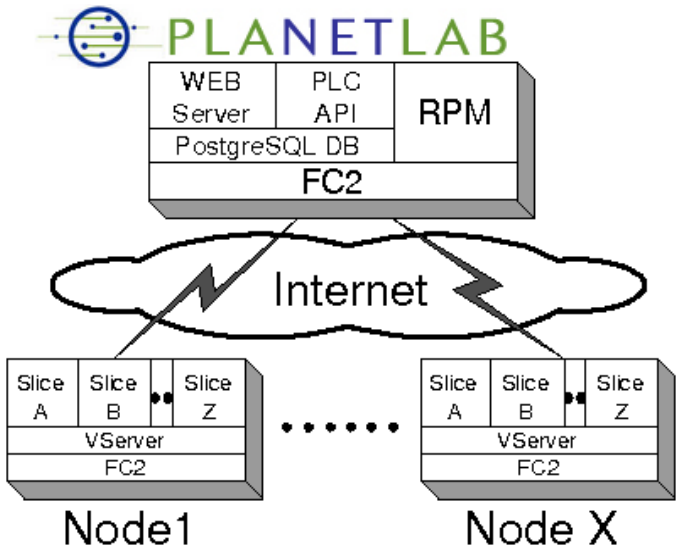


Introdução

- Existem várias pesquisas para a criação de redes com maiores recursos de programação
- Que permitam a inserção gradual de novas tecnologias

- Exemplos:
  - *Active Networks*
  - PlanetLab
  - GENI

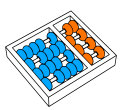




Introdução

Outra linha de pesquisa:

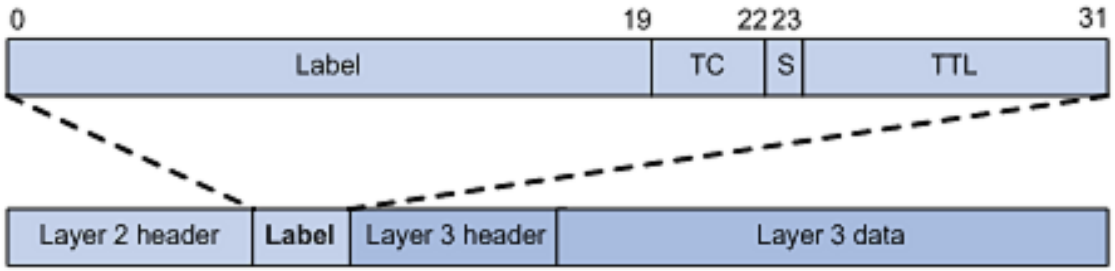
- Fazer pequenas modificações nas operações existentes
- Permitindo o desenvolvimento de hardware de alto desempenho
- Porém, possibilitando maior controle da rede (pelo administrador).

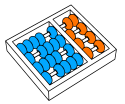


Introdução

Exemplo prático:

- Pequenas modificações na operação de encaminhamento de pacotes (que precisa de alto desempenho)
- Popularizado pelo MPLS (*Multi-protocol Label Switching*)
  - Chaveamento baseado em rótulos programáveis

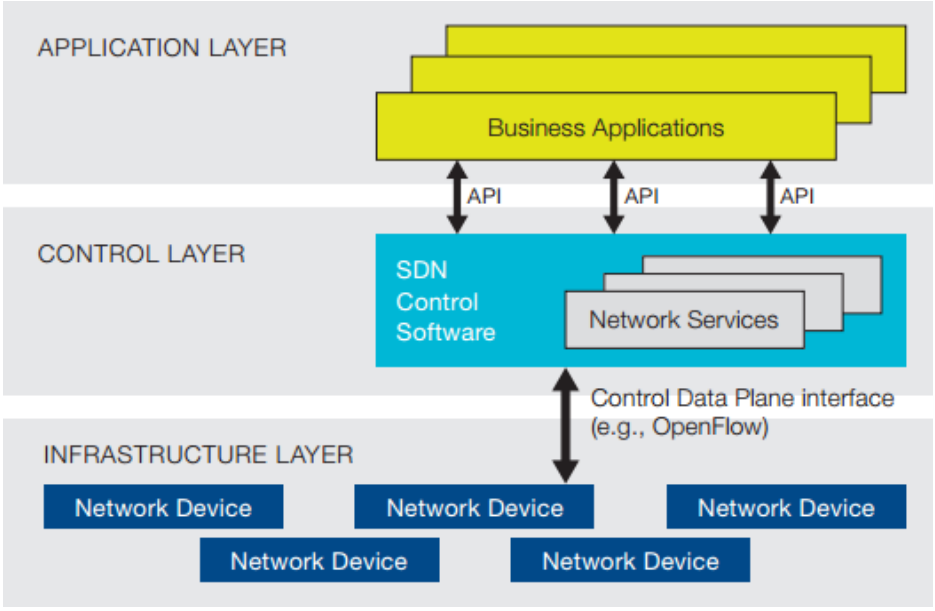




Origens

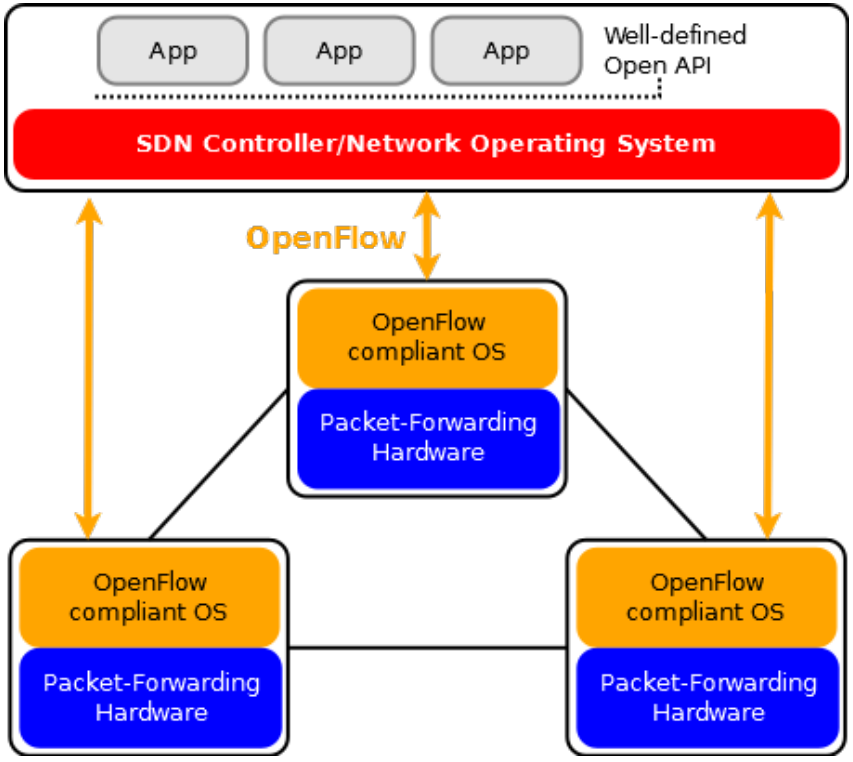
- A iniciativa mais bem sucedida foi o OpenFlow.
- Nele, os elementos de encaminhamento permitem:
- Acesso e controle da tabela de encaminhamento
  - Utilizada pelo hardware

- Porém, a decisão sobre o destino de cada pacote pode ser transferida para um nível superior



Origens

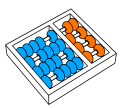
- Essa estrutura permite que a rede possa ser controlada através de aplicações (software)
- Esse novo paradigma ficou conhecido como:
  - SDN (*Software Defined Networks*)
  - Redes Definidas por Software



Motivação

Por que investir no SDN?

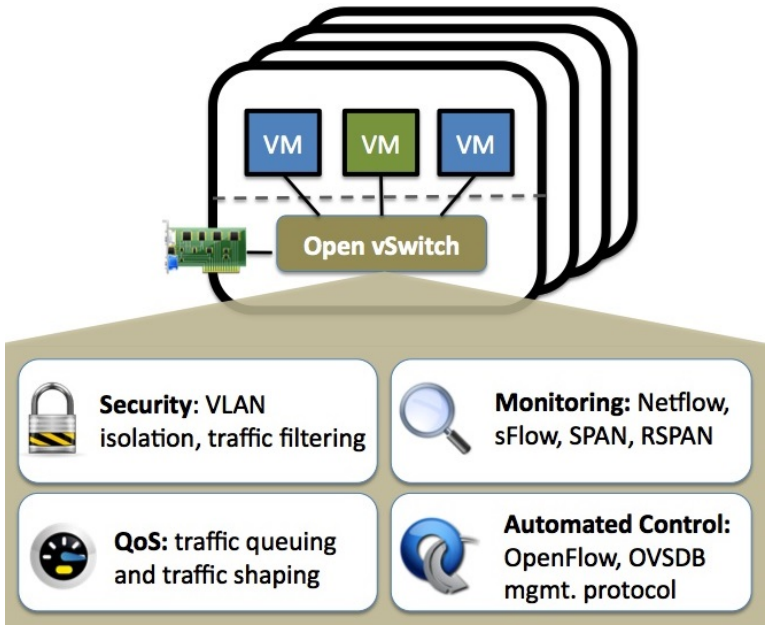
- Devido às diversas possibilidades de aplicação desse paradigma, tem atraído a atenção de pesquisadores e fabricantes.
- A comutação de pacotes não fica limitada aos princípios definidos pelo Ethernet ou IP.



Motivação

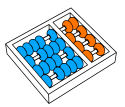
Outros motivos:

- Já existem implementações funcionais do OpenFlow:
  - Como processo de usuário
  - Ou, integrado ao Kernel, para ambientes virtualizados (Open vSwitch)
- Vários fabricantes já oferecem produtos com a interface OpenFlow: Juniper, NEC, HP, Netgear, Cisco, Ciena, etc.



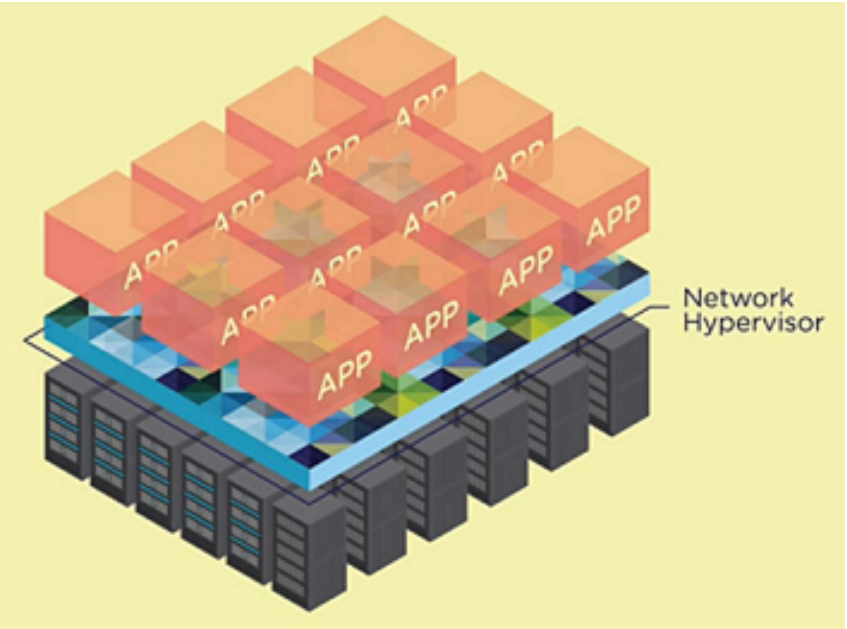
Motivação

- Porém, o padrão OpenFlow é apenas uma parte das SDNs.
- Uma SDN pode apoiar-se no OpenFlow e criar novas aplicações para controle dos elementos de comutação.



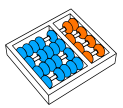
Motivação

- Essas aplicações são conhecidas como:
  - Controladores de Redes
  - Sistemas Operacionais de Redes (*Networks Hypervisors*)



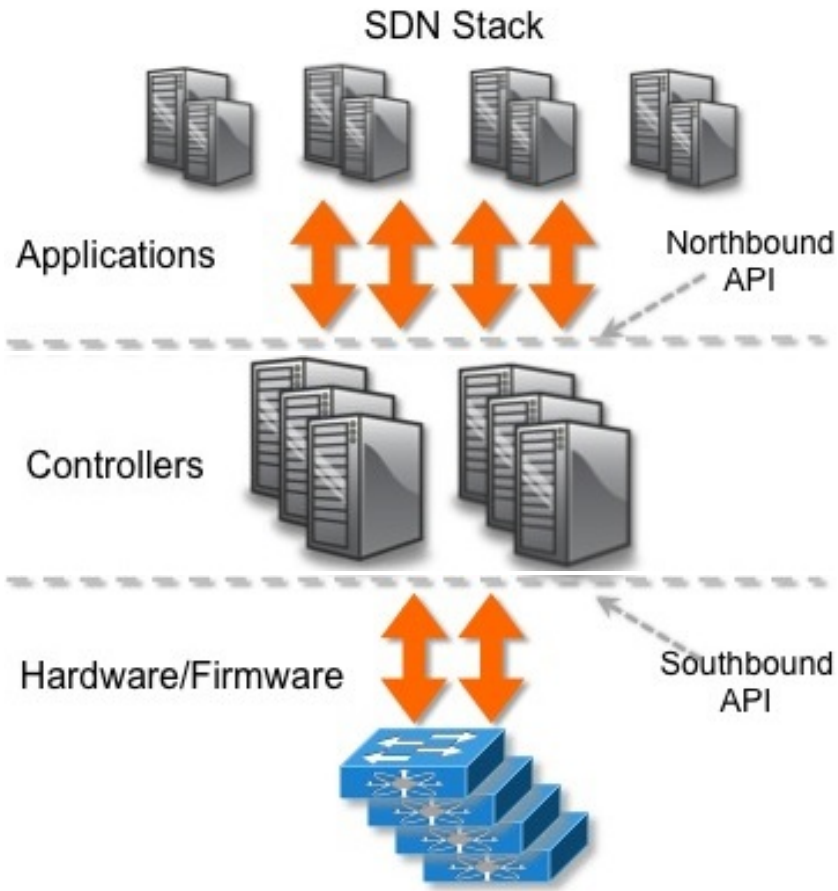
Motivação

- Os controladores de rede permitem:
  - Acessar as interfaces de rede compatíveis
  - Gerar comandos de controle da infraestrutura de chaveamento
- O que possibilita (por exemplo):
  - Criar novas políticas de segurança
  - Controle e monitoramento de tráfego mais sofisticados
  - Visões diferentes para cada usuário de um Datacenter



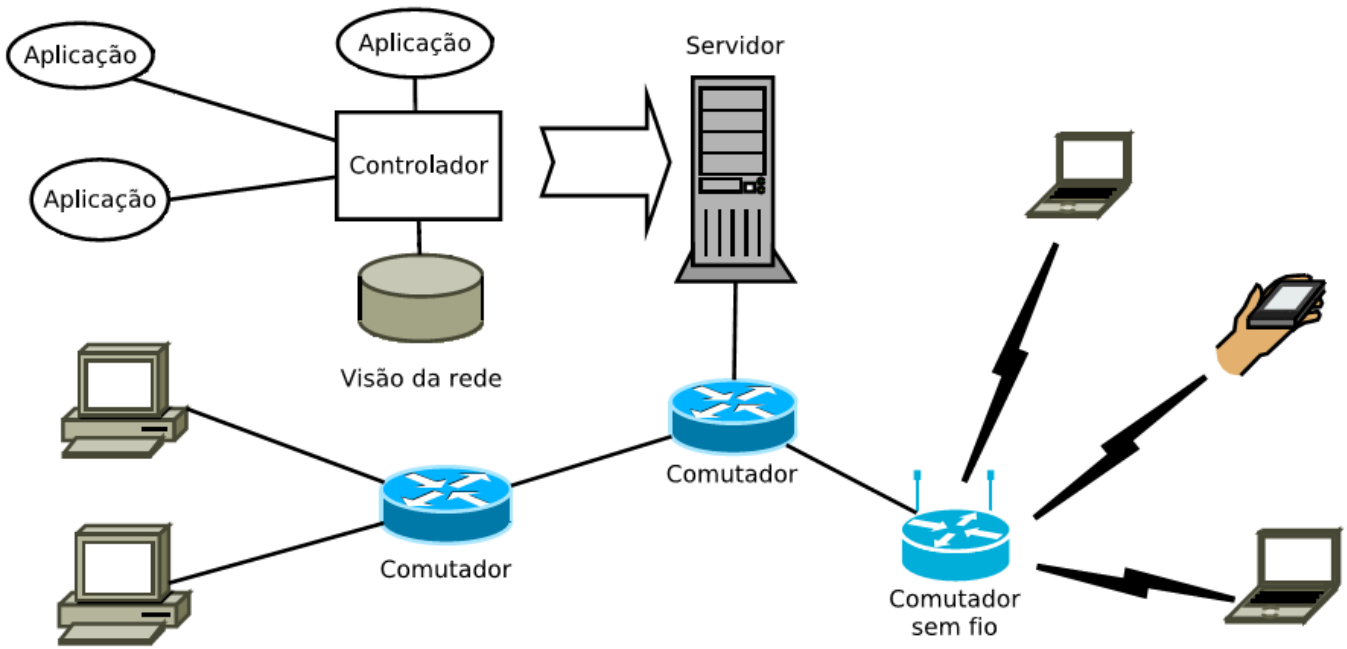
SDN

- Resumindo SDN:
  - É uma rede contendo um sistema de controle (*software*)
  - Que disponibiliza métodos para aplicativos de rede
  - Realizarem o controle do mecanismo de encaminhamento dos elementos de comutação.



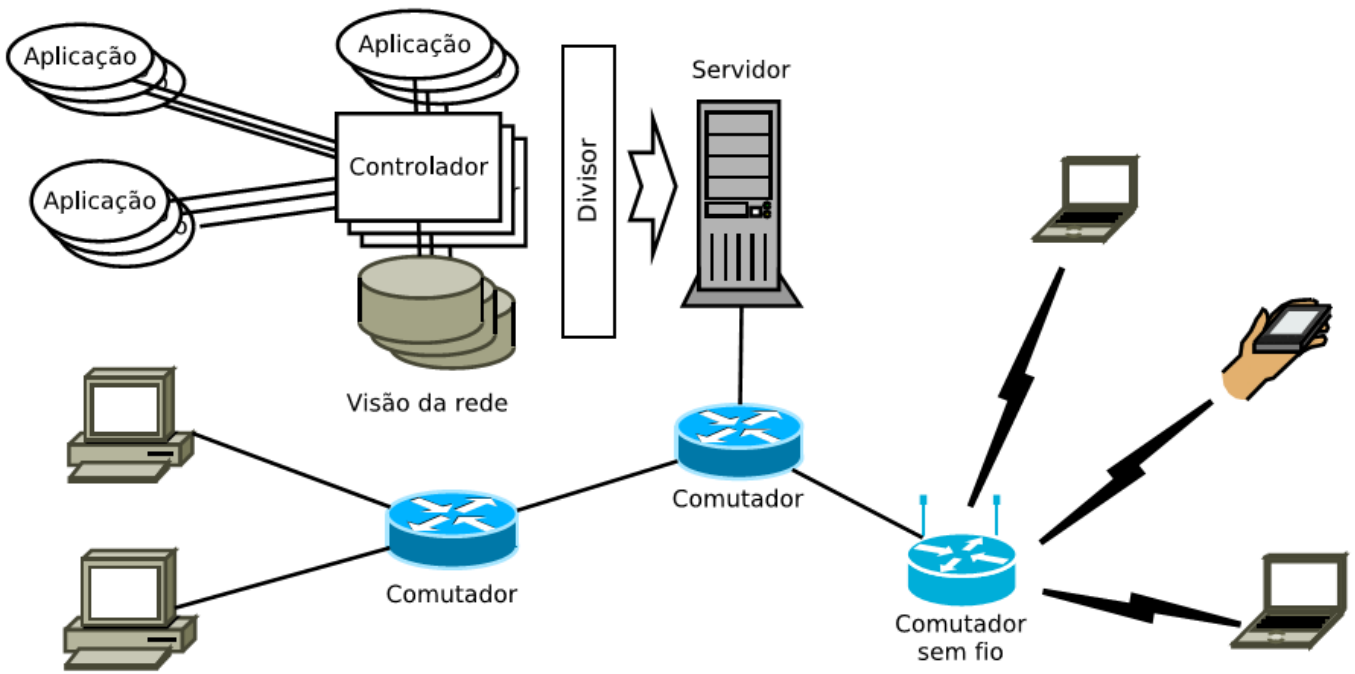
Estrutura do Software

- O software poderia ser uma aplicação monolítica.
- Mas, normalmente é estruturado com:
  - Um controlador geral
  - E várias aplicações específicas



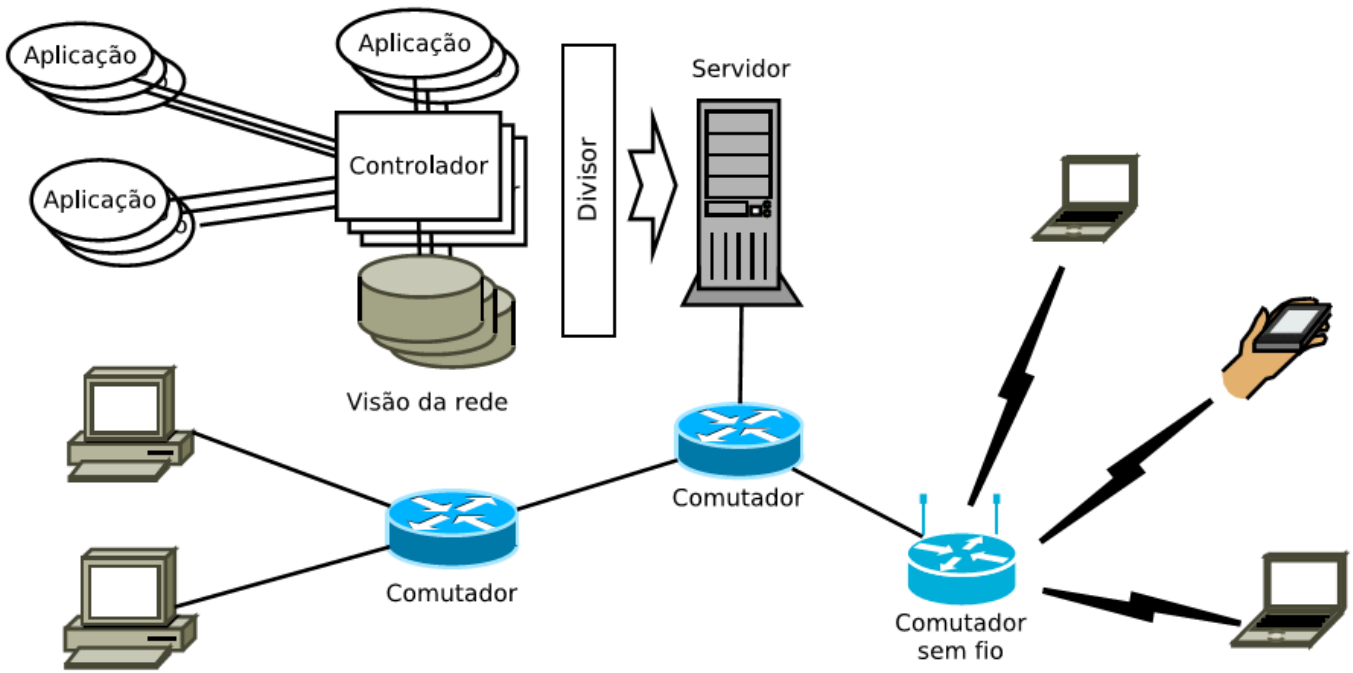
Estrutura do Software

- Outra possibilidade é ter um divisor de visões:
  - Com vários Controladores
  - E suas respectivas aplicações

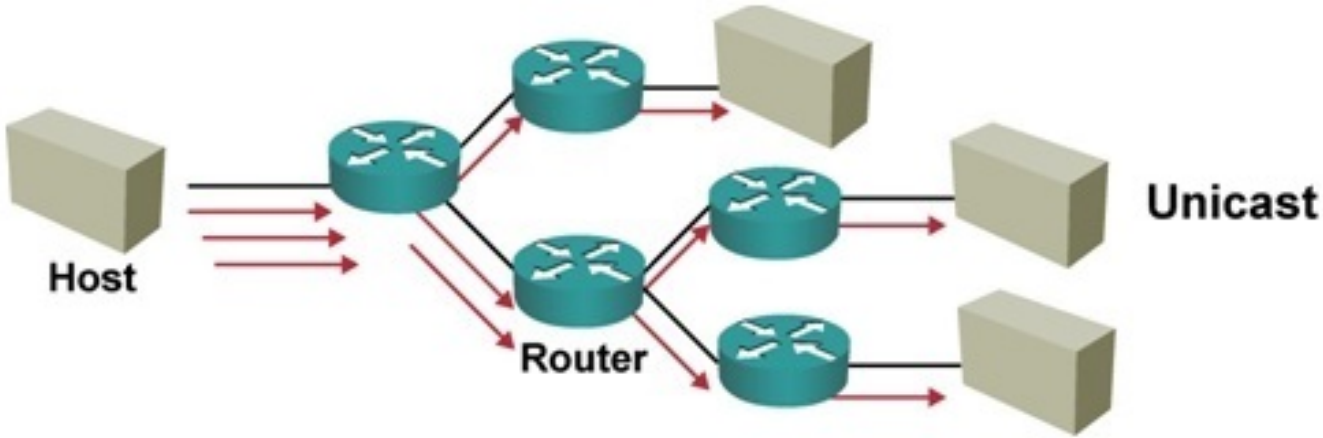


Estrutura geral de uma SDN

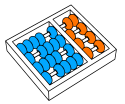
- Uma SDN é composta pelos seguintes elementos:
  - Elementos de comutação programáveis
  - Divisor de recursos / visões
  - Controladores
  - Aplicações de rede



Elementos de comutação programáveis



- Lembrando da operação de encaminhamento nas redes baseadas em pacotes:
  - O pacote é recebido pela interface
  - Depois é inspecionado
  - É feita consulta à tabela de encaminhamento
    - Qual o destino (MAC, IP, etc)
  - Se destino foi identificado:
    - Envia pacote para a porta de destino
  - Se não foi encontrado:
    - O pacote é descartado, ou
    - é realizada uma operação *Default*

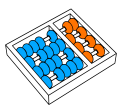


Elementos de comutação programáveis

- O hardware atual realiza o encaminhamento com alto desempenho.

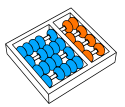
  $\therefore$ É problemático acrescentar novas funcionalidades

- Então, como uma SDN pode modificar a rede sem modificar o hardware?

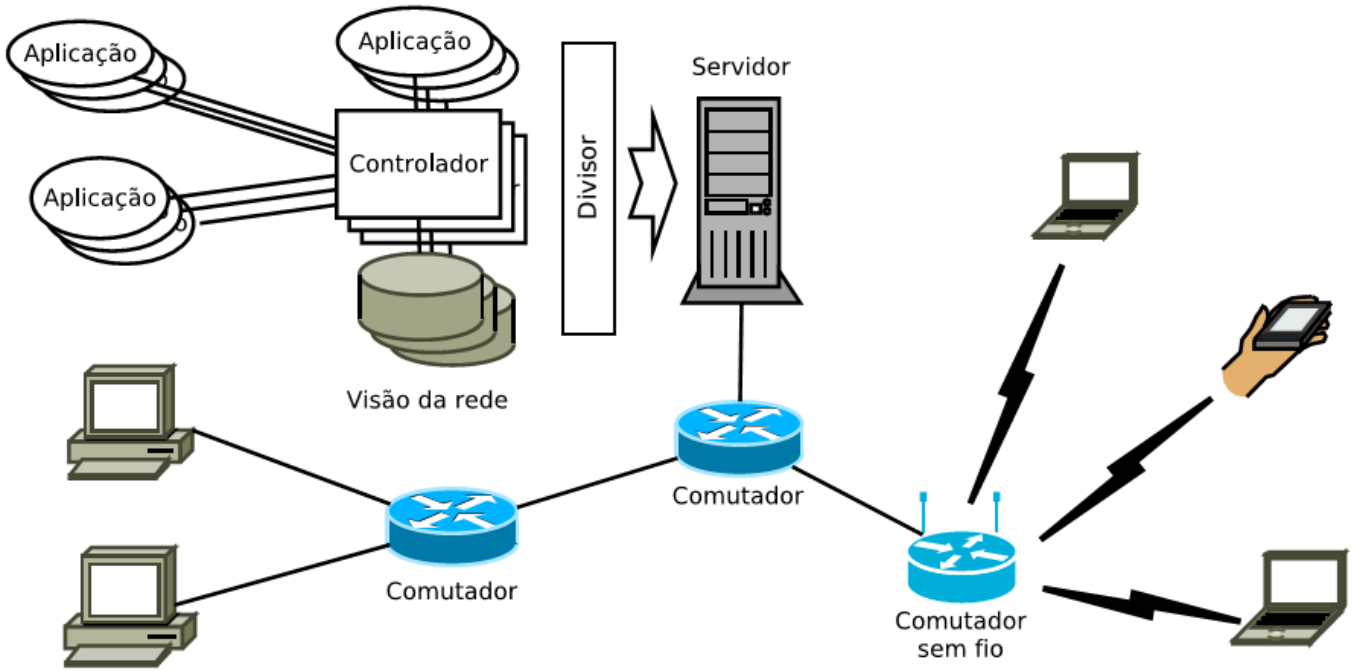


Elementos de comutação programáveis

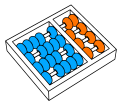
- As SDNs se restringem à manipulação simples de pacotes
- Baseado no conceito de fluxos
  - Sequência de pacotes com mesmos valores em seus atributos.
- Dessa forma, basta:
  - Controlar o conteúdo da tabela de encaminhamento.
  - E indicar ao hardware a ação a ser tomada, ao detectar um padrão.



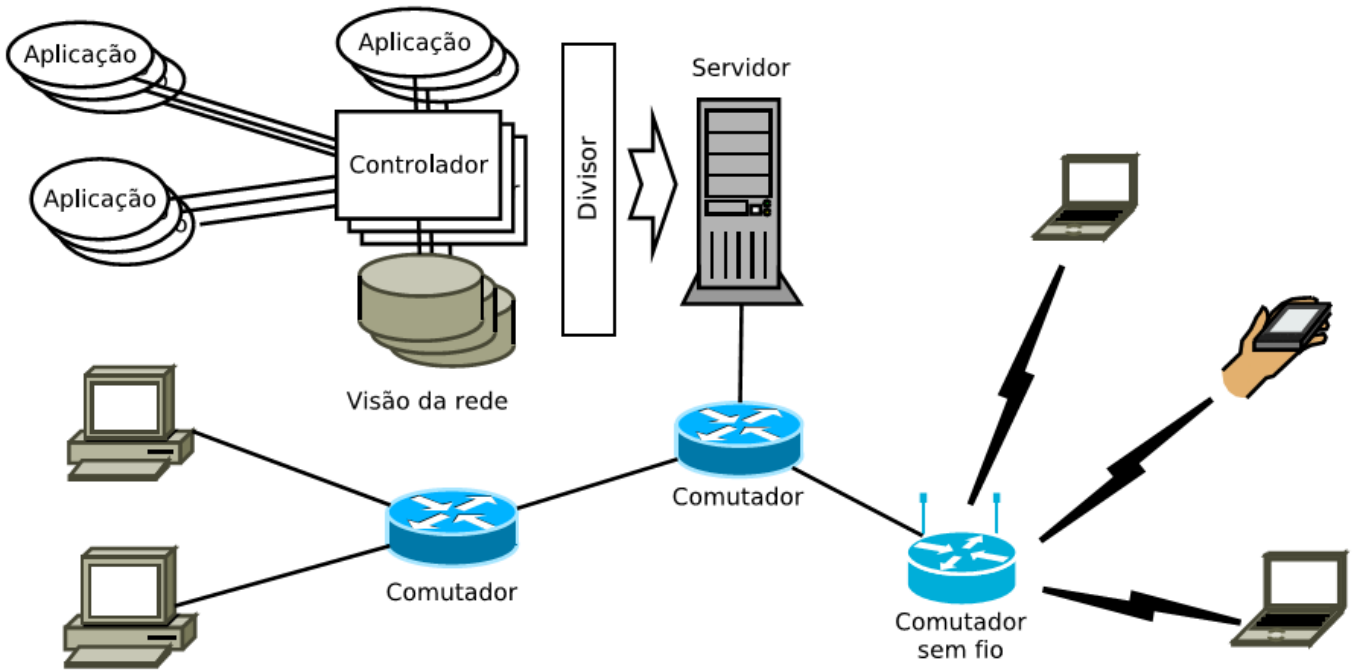
Divisores de recursos/visões



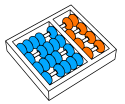
- Isso foi possível pois:
  - Manteve-se os “Fluxos de Operação” da rede
  - E estendeu-se o tratamento para os “Fluxos de Pesquisa”



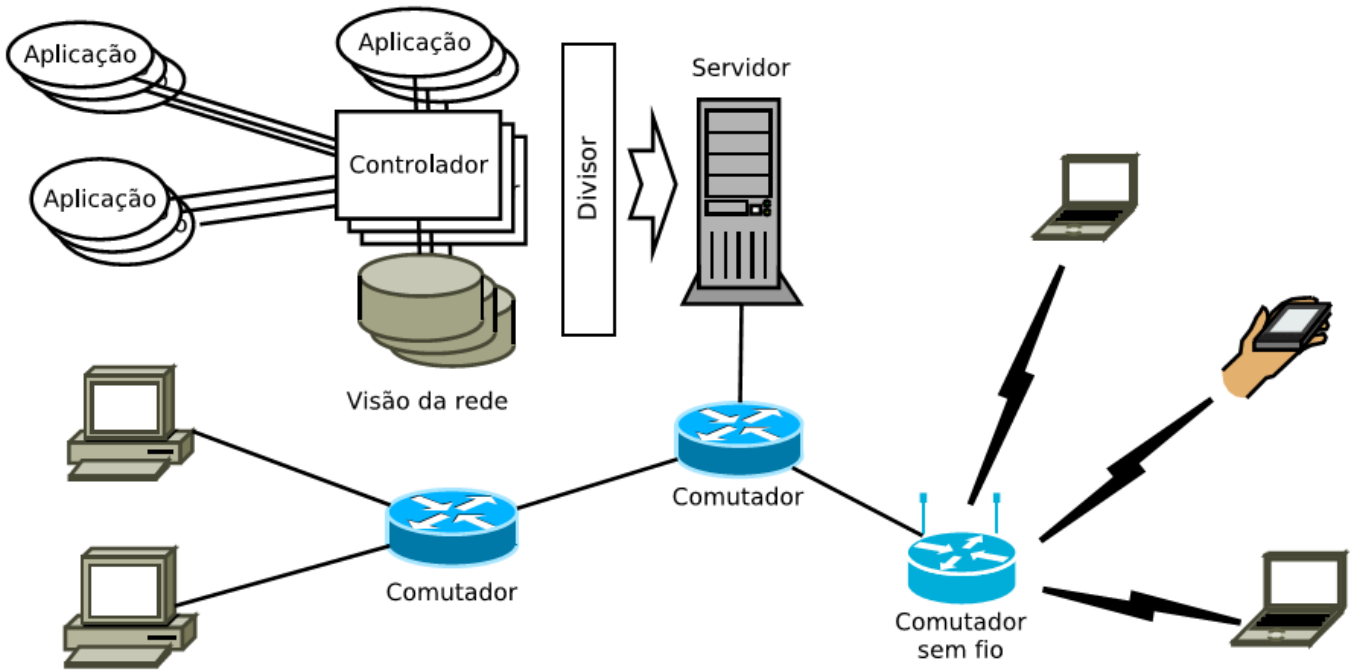
Divisores de recursos/visões



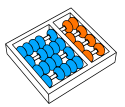
- Com novos tipos de pesquisa trabalhando em paralelo:
  - É possível criar visões diferente da rede
  - Cada uma com sua cota de recursos
- Seguindo o princípio de particionamento de tráfego Internet



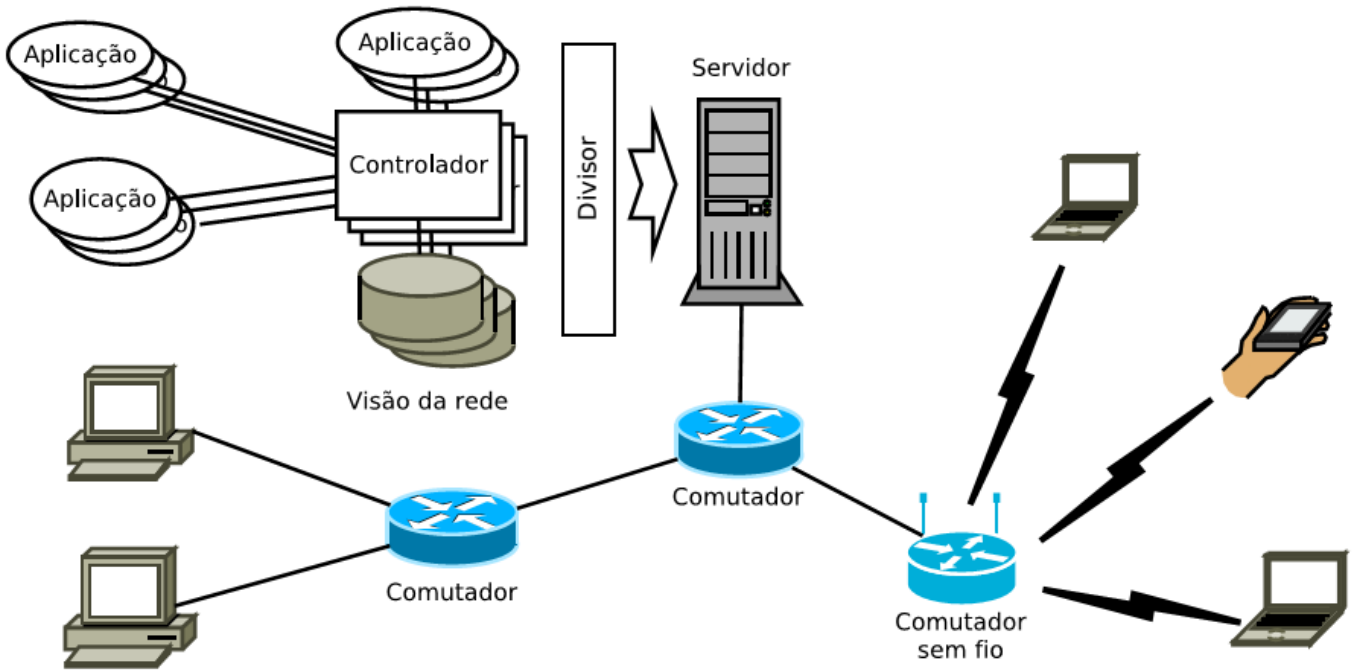
Controlador



- O que faz o elemento controlador ?
  - Oculta os detalhes internos da rede
  - Centraliza a comunicação com os elementos programáveis
  - Oferece uma visão unificada da rede



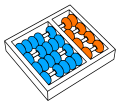
Controlador



- E pode trabalhar de forma distribuída
  - Através da divisão dos elementos de visão
  - Ou através de algoritmos distribuídos

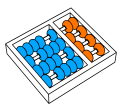
Controlador

- Existem vários tipos de controladores diferentes
  - Ryu
  - NOX
  - Flowvisor
  - Routeflow
  - Trema
  - Maestro
  - Beacon
  - Onix



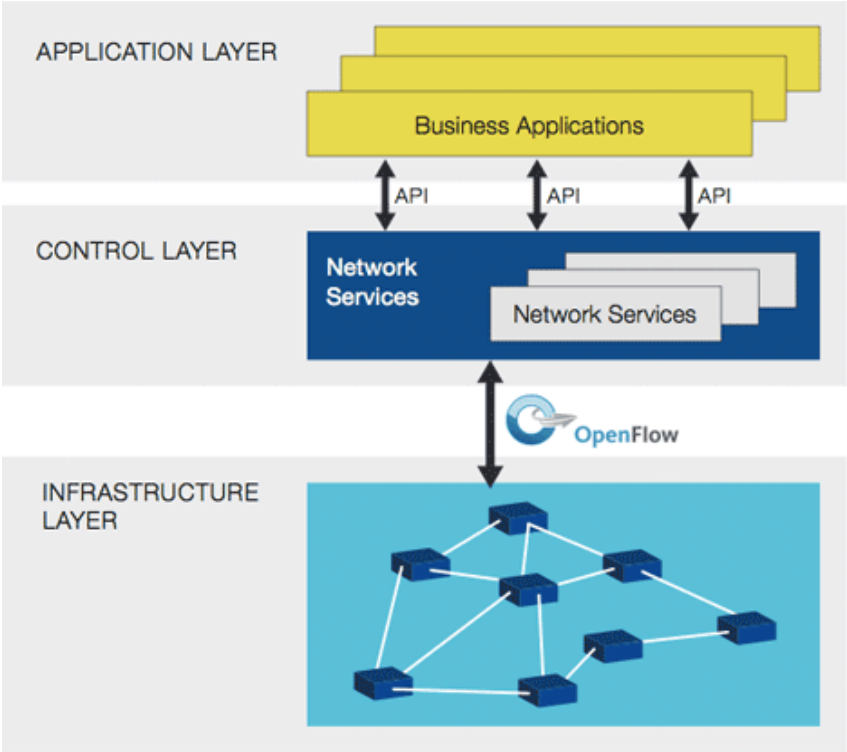
Aplicações de Rede

- São as funcionalidades acrescentadas às redes. Ex.:
  - Soluções de roteamento
  - Controle de interação entre os comutadores
    - Simulando um único *Switch* ou Roteador IP
  - Controle de acesso
  - Gerência de redes
  - Gerência de energia
  - Comutador Virtual
  - Roteador expansível de alta capacidade
  - *Datacenters* multi-usuários



OpenFlow

- É um padrão aberto para SDN
- Funciona como um protocolo de comunicação entre:
  - O controlador
  - E os equipamentos

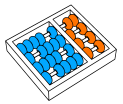


OpenFlow

- A evolução das SDNs está diretamente ligada ao sucesso do OpenFlow.

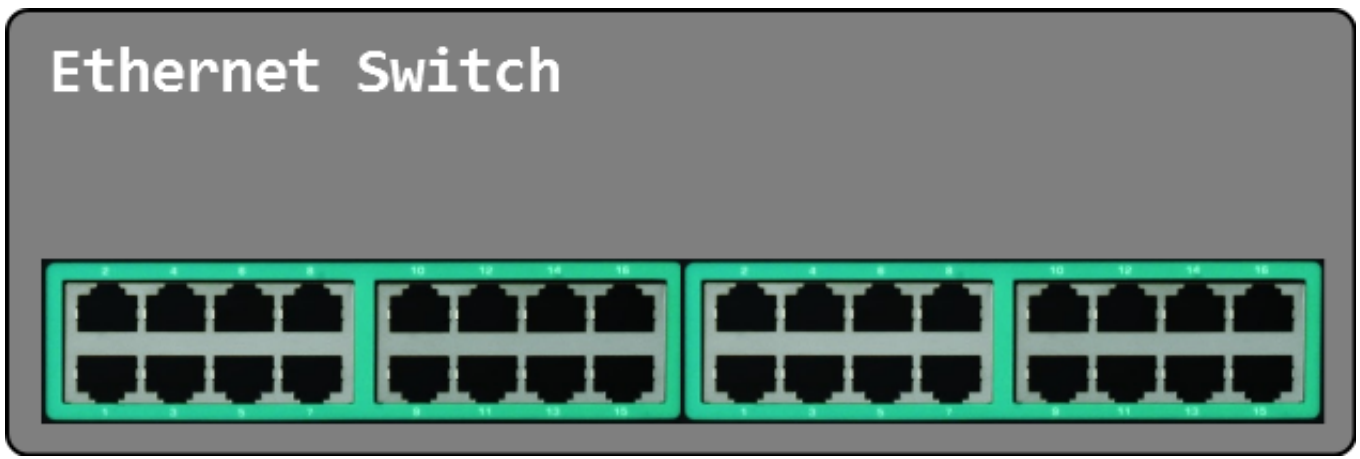
- Ele permite a realização de pesquisas e testes de novos protocolos
- Em equipamentos de redes comerciais
- Em paralelo com a operação normal das redes

- Isso é possível através de uma API
- Que permite aos programadores controlar os elementos de encaminhamento de pacotes



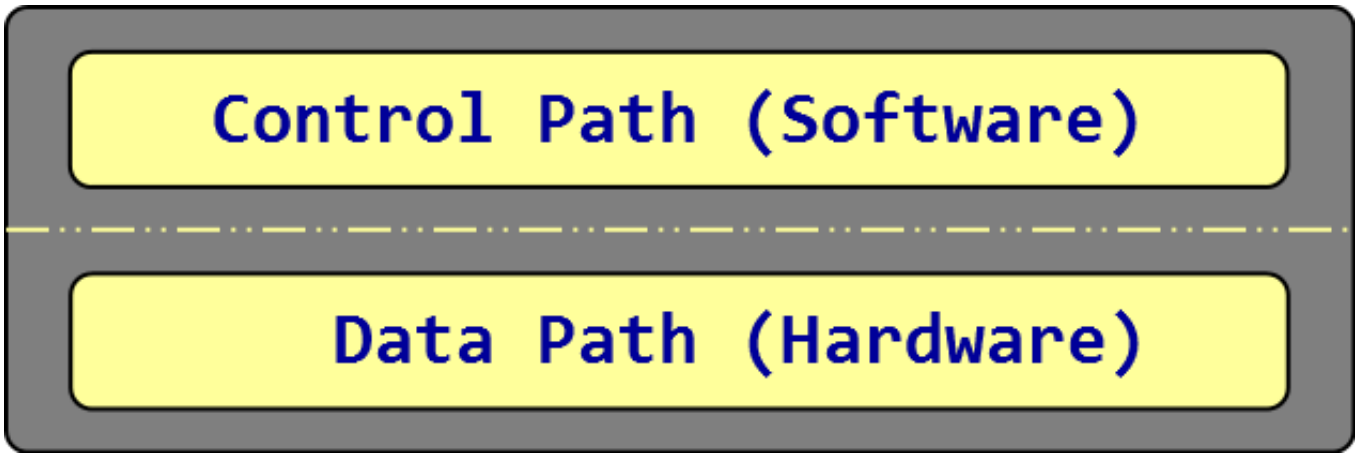
Estrutura do OpenFlow

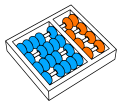
- Estrutura de um *Switch* tradicional [3]:



Estrutura do OpenFlow

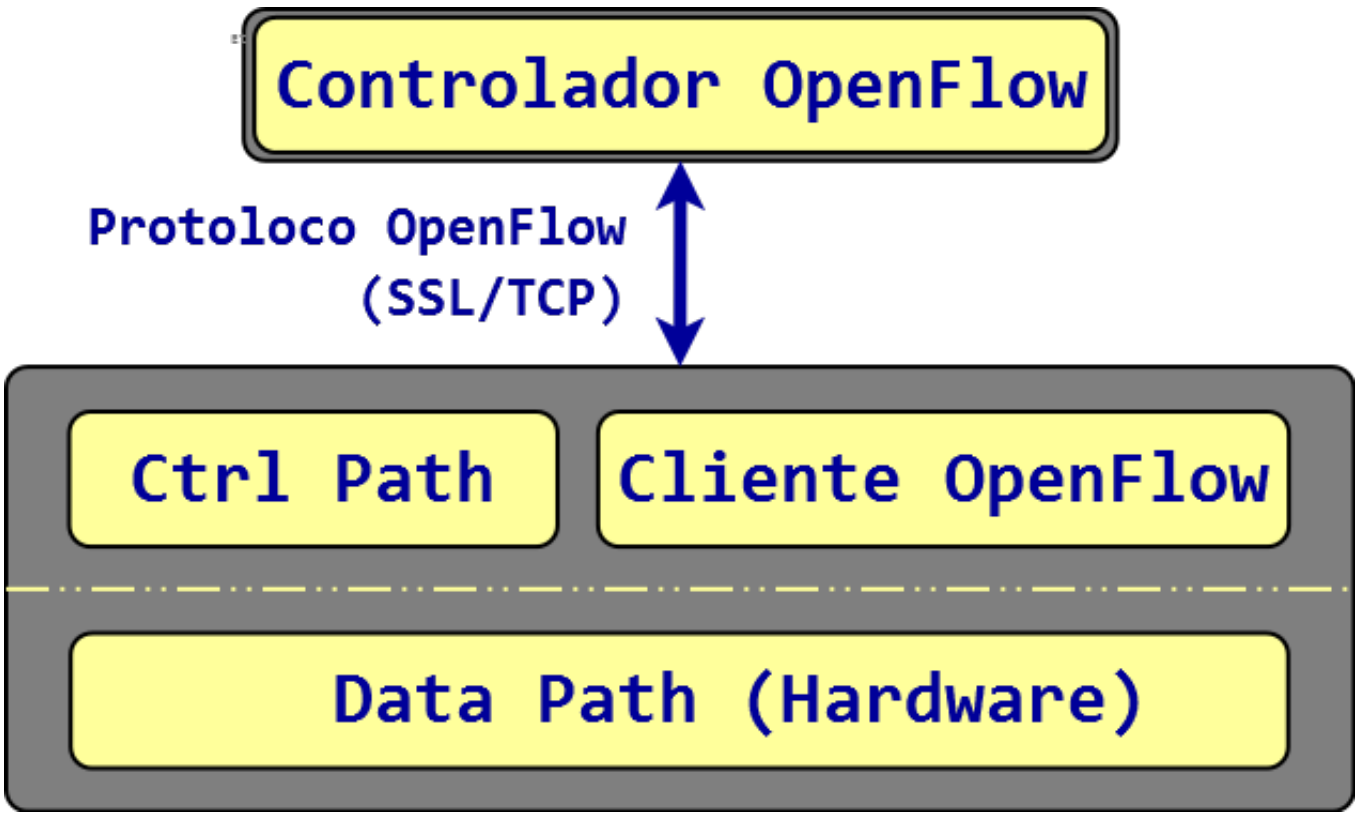
- Estrutura de um *Switch* tradicional [3]:

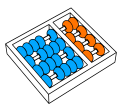




Estrutura do OpenFlow

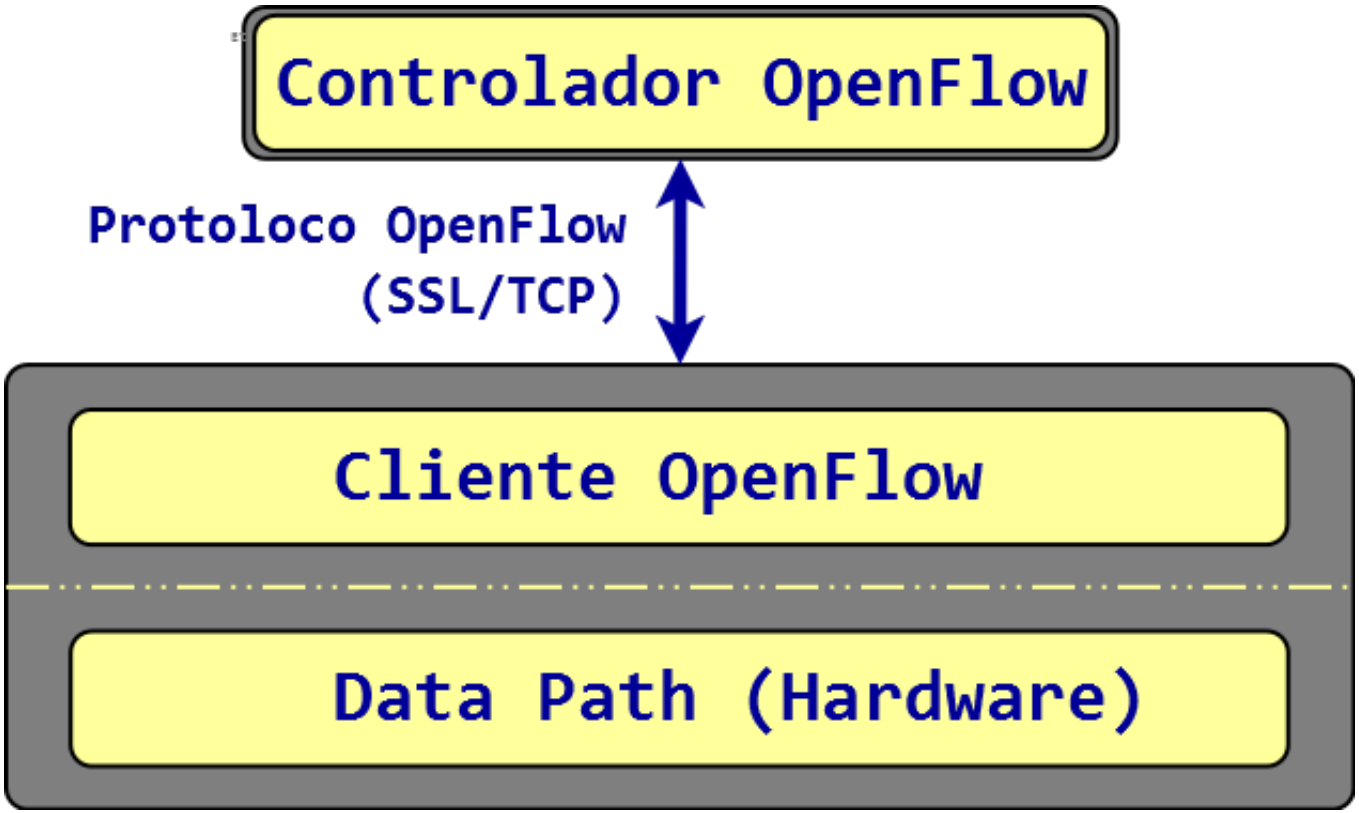
- Estrutura de um *Switch OpenFlow-Hybrid*:





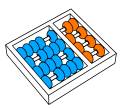
Estrutura do OpenFlow

- Estrutura de um *Switch OpenFlow-Only*:



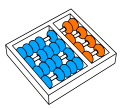
Estrutura do OpenFlow

- No OpenFlow existe uma separação bem definida entre:
  - Plano de Dados
  - Plano de Controle



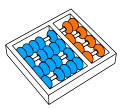
OpenFlow - Plano de Dados

- Cuida do encaminhamento de pacotes
- Utiliza regras simples (Chamada de Ações):
  - Encaminhar pacote
  - Alterar parte do cabeçalho
  - Descartar pacote
  - Encaminhar para o controlador (inspeção)



OpenFlow - Plano de Controle

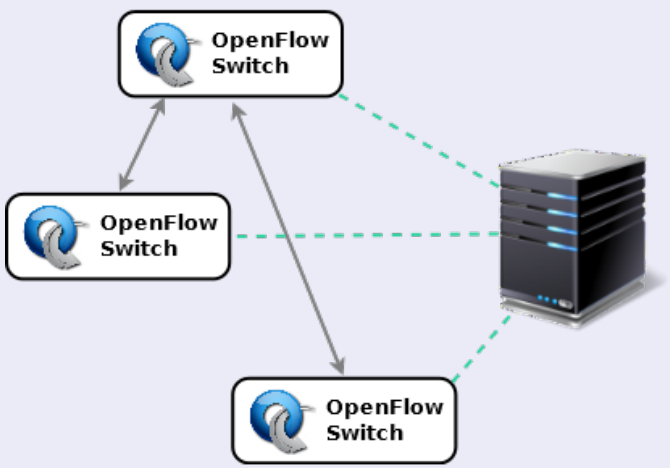
- Permite programar as entradas na tabela de encaminhamento
- Com padrões que identifiquem:
  - Os fluxos de interesse
  - E as regras associadas a eles



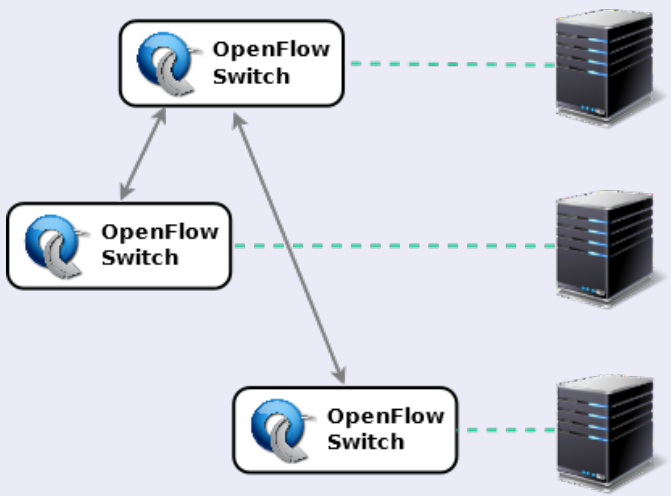
OpenFlow - Plano de Controle

- Esse módulo de software pode ser implementado de forma independente (em algum ponto da rede)
- Ou trabalhar distribuidamente

Controle Centralizado

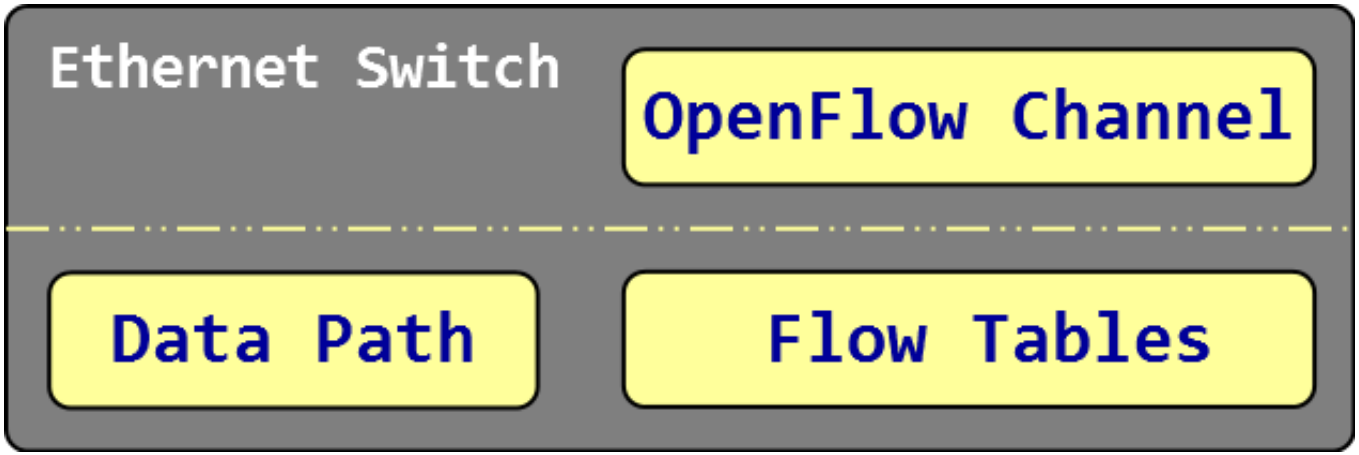


Controle Distribuído



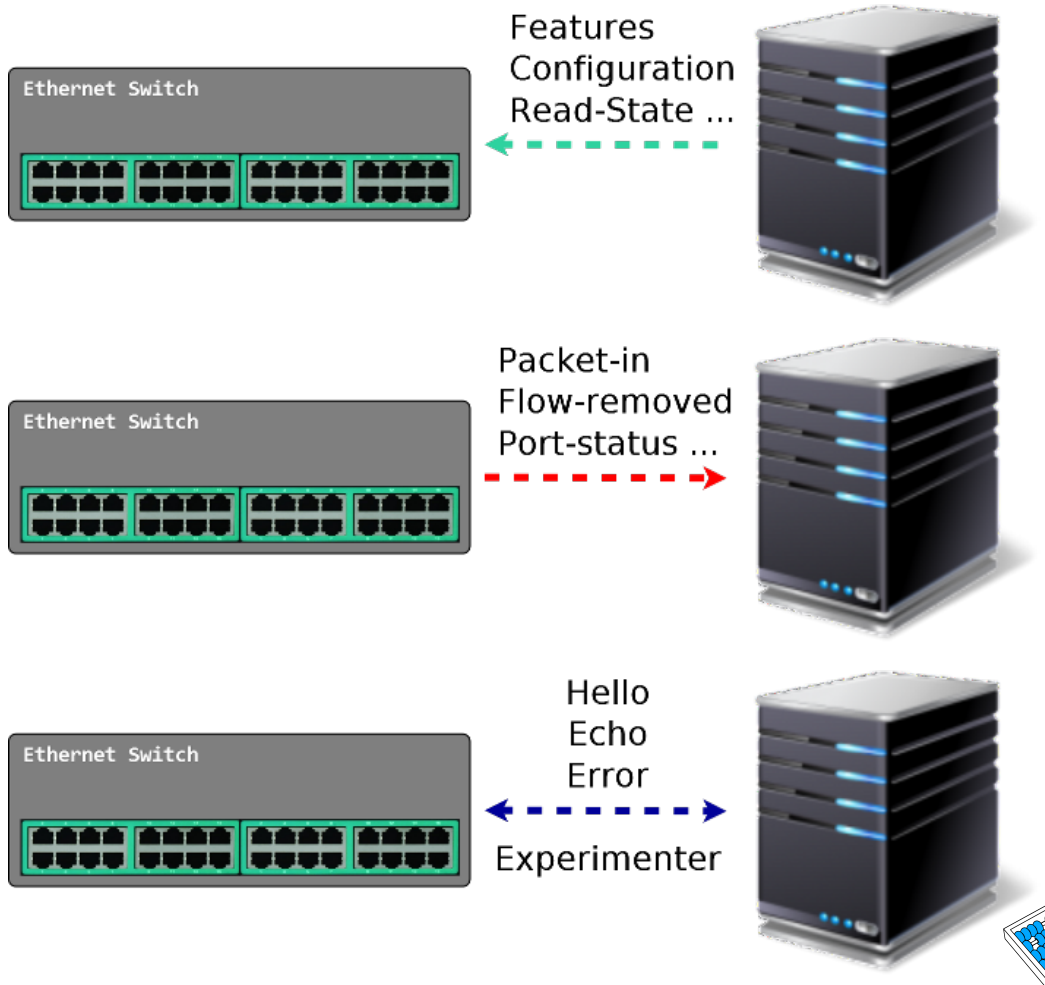
OpenFlow - Especificações

- Requisitos do *Switch* [5]:
  - *Flow Tables*
  - *OpenFlow Channel*
  - Tipos de Porta



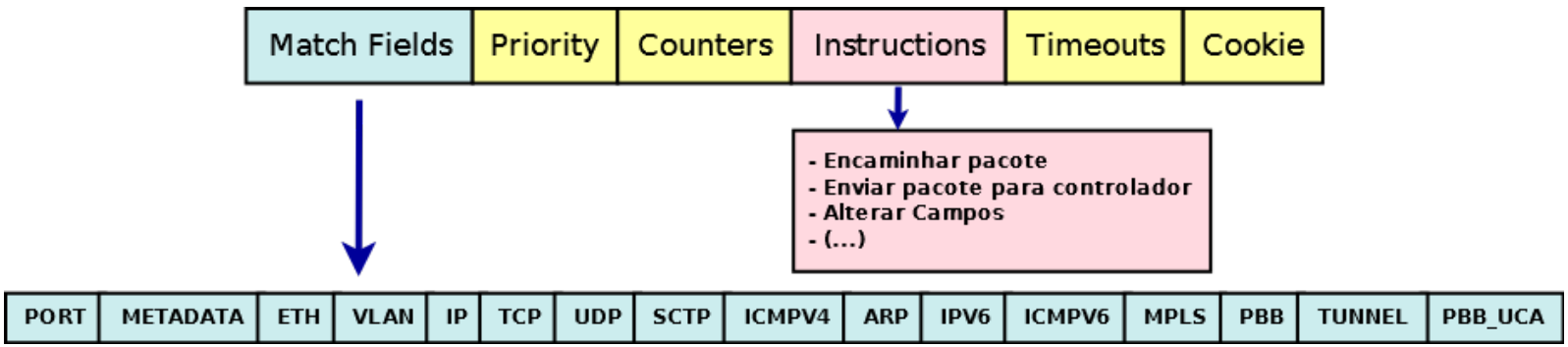
OpenFlow - Especificações

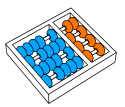
- Tipos de Mensagens:
  - *Controller-to-Switch*
  - *Asynchronous*
  - *Symmetric*



Estrutura

- Cada entrada da tabela de fluxo contém [5]:
  - ***Match Fields***: para comparar com o pacotes.
  - ***Priority***: prioridade do fluxo de entrada.
  - ***Counters***: estatísticas para o fluxo.
  - ***Instructions***: ações a realizar.
  - ***Timeouts***: tempo máximo ou tempo ocioso para excluir da tabela.
  - ***Cookie***: utilizado somente pelo controlador.





Exemplo da Tabela de Fluxo

- Um *Switch* OpenFlow utiliza memórias TCAM (*Ternary Content-Addressable Memory*) para as tabelas de fluxo.

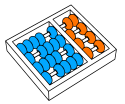
Nelas os bits podem ser representados com:

0 (Zero)

1 (Um)

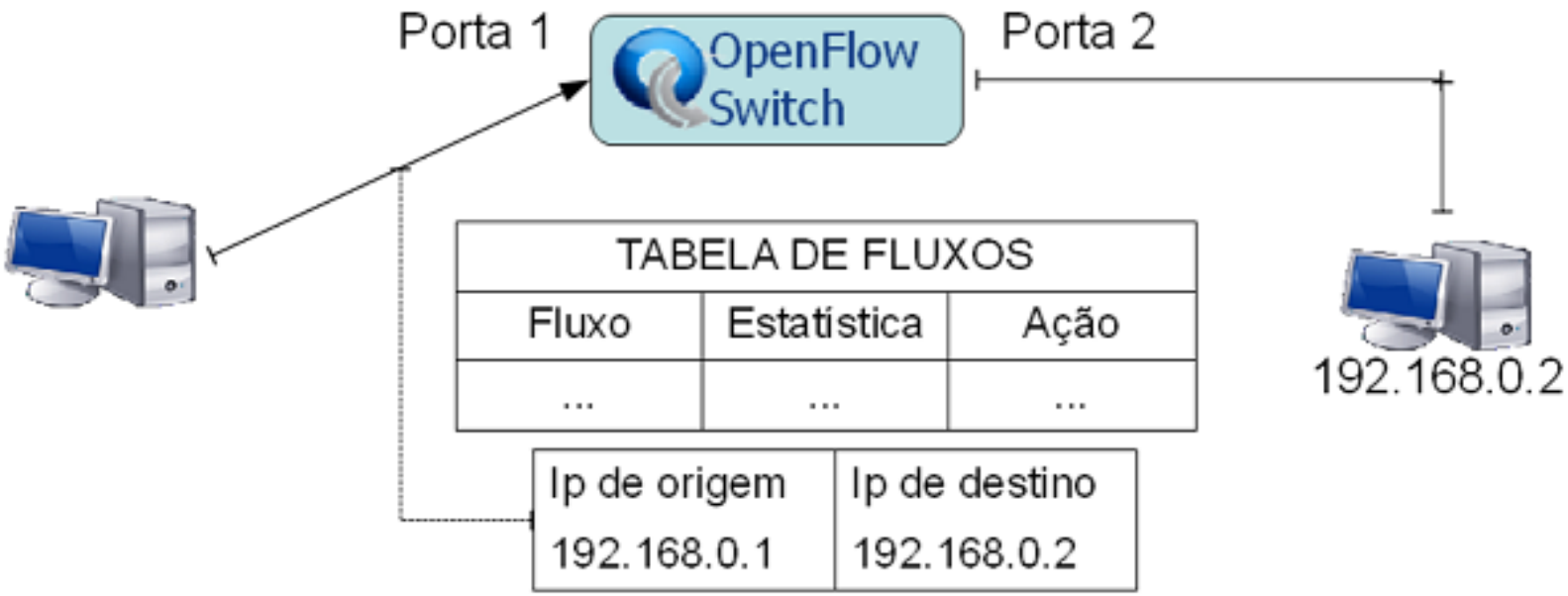
* (Não importa - *Don't care*)

IN_PORT	...	ETH_DST	ETH_SRC	...	IPV4_SRC	IPV4_DST	...	INSTRUCTION
5	*	00:24:D7:63:2C:14	58:B0:35:F6:12:F1	*	*	01*	*	Action 1
1	*	*	*	*	010*	100*	*	Action 2
*	*	*	*	*	101*	011*	*	Action 3
*	*	*	*	*	*	*	*	Action 4



Funcionamento

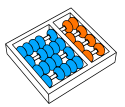
- Quando o *switch* recebe um pacote ele deve compará-lo com a sua tabela de fluxo.



Funcionamento

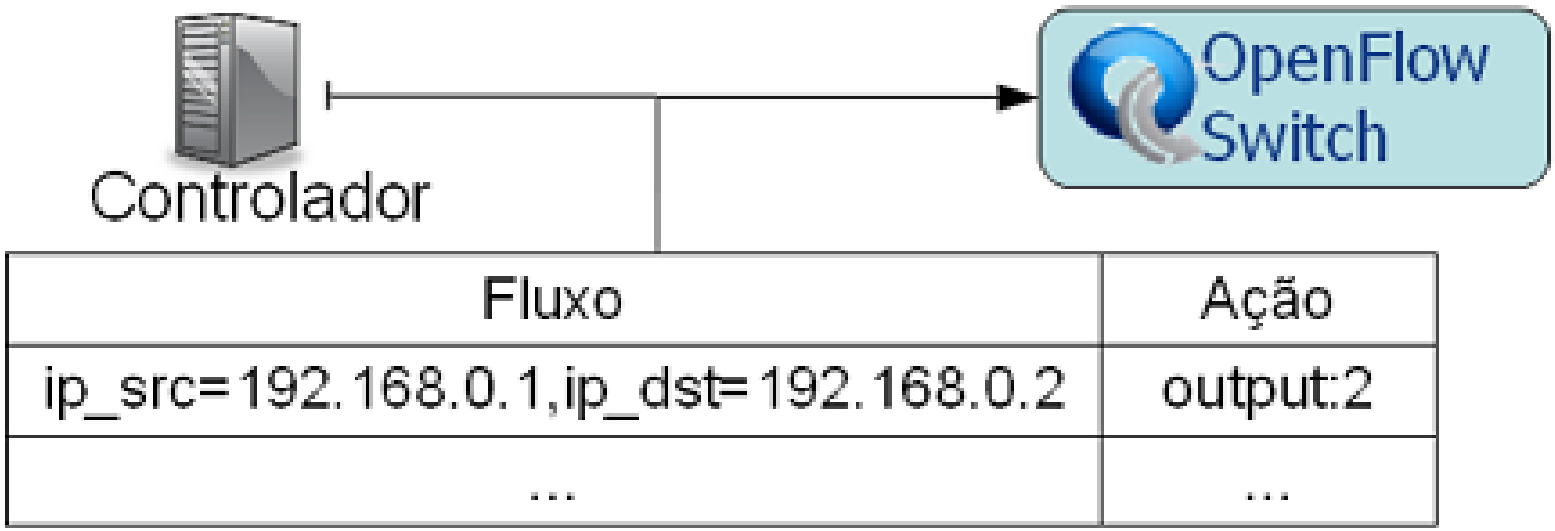
- Se o cabeçalho não for compatível com nenhum fluxo em sua tabela, ele deve encaminhar o cabeçalho para o controlador.

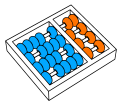
Ip de origem	Ip de destino
192.168.0.1	192.168.0.2



Funcionamento

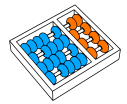
- Ao receber o cabeçalho, o controlador toma uma decisão:
  - Enviar uma tabela de fluxos para o *switch*;
  - Ou indicar uma ação para o pacote (por exemplo: descartá-lo).





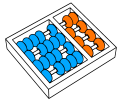
Estrutura de *Match Fields* I

```
/* OXM Flow match field types for OpenFlow basic class
   (oxm_class=OFPXMC_OPENFLOW_BASIC) */
enum oxm_ofb_match_fields {
  OFPXMT_OFB_IN_PORT      = 0,  // Switch input port
  OFPXMT_OFB_IN_PHY_PORT  = 1,  // Switch physical input port
  OFPXMT_OFB_METADATA     = 2,  // Metadata passed between
                                // tables
  OFPXMT_OFB_ETH_DST      = 3,  // Ethernet destination
                                // address
  OFPXMT_OFB_ETH_SRC      = 4,  // Ethernet source address
  OFPXMT_OFB_ETH_TYPE     = 5,  // Ethernet frame type
  OFPXMT_OFB_VLAN_VID     = 6,  // VLAN id
  OFPXMT_OFB_VLAN_PCP     = 7,  // VLAN priority
  OFPXMT_OFB_IP_DSCP      = 8,  // IP DSCP (6 bits in ToS
                                // field)
  OFPXMT_OFB_IP_ECN       = 9,  // IP ECN (2 bits in ToS
                                // field)
  OFPXMT_OFB_IP_PROTO     = 10, // IP protocol
  OFPXMT_OFB_IPV4_SRC     = 11, // IPv4 source address
  OFPXMT_OFB_IPV4_DST     = 12, // IPv4 destination address
```



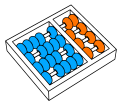
Estrutura de *Match Fields* II

```
OFPXMT_OFB_TCP_SRC      = 13, // TCP source port
OFPXMT_OFB_TCP_DST      = 14, // TCP destination port
OFPXMT_OFB_UDP_SRC      = 15, // UDP source port
OFPXMT_OFB_UDP_DST      = 16, // UDP destination port
OFPXMT_OFB_SCTP_SRC     = 17, // SCTP source port
OFPXMT_OFB_SCTP_DST     = 18, // SCTP destination port
OFPXMT_OFB_ICMPV4_TYPE  = 19, // ICMP type
OFPXMT_OFB_ICMPV4_CODE  = 20, // ICMP code
OFPXMT_OFB_ARP_OP       = 21, // ARP opcode
OFPXMT_OFB_ARP_SPA      = 22, // ARP source IPv4 address
OFPXMT_OFB_ARP_TPA      = 23, // ARP target IPv4 address
OFPXMT_OFB_ARP_SHA      = 24, // ARP source hardware
                              // address
OFPXMT_OFB_ARP_THA      = 25, // ARP target hardware
                              // address
OFPXMT_OFB_IPV6_SRC     = 26, // IPv6 source address
OFPXMT_OFB_IPV6_DST     = 27, // IPv6 destination address
OFPXMT_OFB_IPV6_FLABEL  = 28, // IPv6 Flow Label
OFPXMT_OFB_ICMPV6_TYPE  = 29, // ICMPv6 type
OFPXMT_OFB_ICMPV6_CODE  = 30, // ICMPv6 code
```



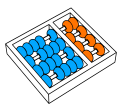
Estrutura de *Match Fields* III

```
  OFPXMT_OFB_IPV6_ND_TARGET= 31, // Target address for ND
  OFPXMT_OFB_IPV6_ND_SLL   = 32, // Source link-layer for ND
  OFPXMT_OFB_IPV6_ND_TLL   = 33, // Target link-layer for ND
  OFPXMT_OFB_MPLS_LABEL    = 34, // MPLS label
  OFPXMT_OFB_MPLS_TC       = 35, // MPLS TC
  OFPXMT_OFP_MPLS_BOS      = 36, // MPLS BoS bit
  OFPXMT_OFB_PBB_ISID      = 37, // PBB I-SID
  OFPXMT_OFB_TUNNEL_ID     = 38, // Logical Port Metadata
  OFPXMT_OFB_IPV6_EXTHDR   = 39, // IPv6 Extension Header
                                 // pseudo-field
  OFPXMT_OFB_PBB_UCA       = 41, // PBB UCA header field
};
```



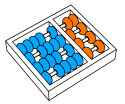
Limitações e futuras versões

- Desde a versão 1.0.0 (31/12/2009) até a atual 1.4.0 (14/10/2013, ver [5]), a especificação do OpenFlow já sofreu várias modificações.
- Porém, ainda existem algumas limitações, por exemplo:
  - Definição do padrão para circuitos ópticos e
  - Definição de fluxos que englobe protocolos fora do modelo TCP/IP.
- A versão 2.0 está sendo formulada com o intuito de eliminar essas limitações (entre outras).



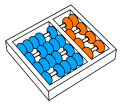
Controladores de Rede

- Como apresentado, existem vários tipos de controladores de rede, como exemplo temos:
- OpenFlow Reference
  - Stanford/Nicira
  - Não é projetado para ser extensível
- RouteFlow
  - CPqD (Brasil)
  - Encaminhamento IP como um serviço (quagga)
- NOX
  - Nicira
  - Desenvolvendo ativamente
- Ryu
  - Nippon Telegraph and Telephone Corporation
  - Framework



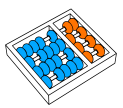
NOX

- É o controlador original do OpenFlow.
- Possibilita o desenvolvimento de controladores em C++ ou Python
- Trabalha sobre o conceito de fluxos de dados
  - Checa o primeiro pacote de cada fluxo
  - E determina a política a ser aplicada



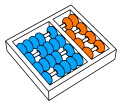
NOX

- É um sistema operacional de rede simples
- Provê primitivas para:
  - Gerenciamento dos eventos
  - Funções para a comunicação com os *switches*
- Os mesmos desenvolvedores do NOX desenvolveram o POX, com a premissa dele ser completamente escrito em Python.



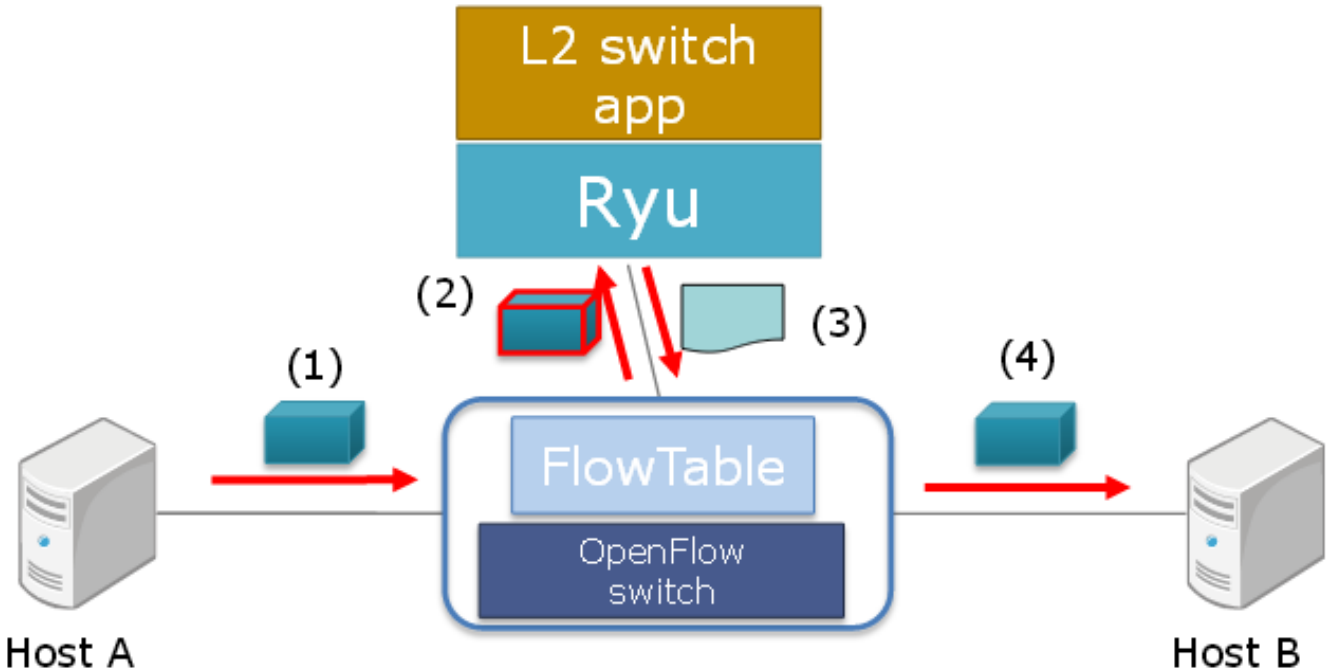
Ryu

- É um *Framework* para desenvolvimento de aplicações SDN, ao invés de um controlador monolítico [7].
- Tem a filosofia de permitir um desenvolvimento ágil.
- É um software Open Source (Apache v2), totalmente escrito em Python.



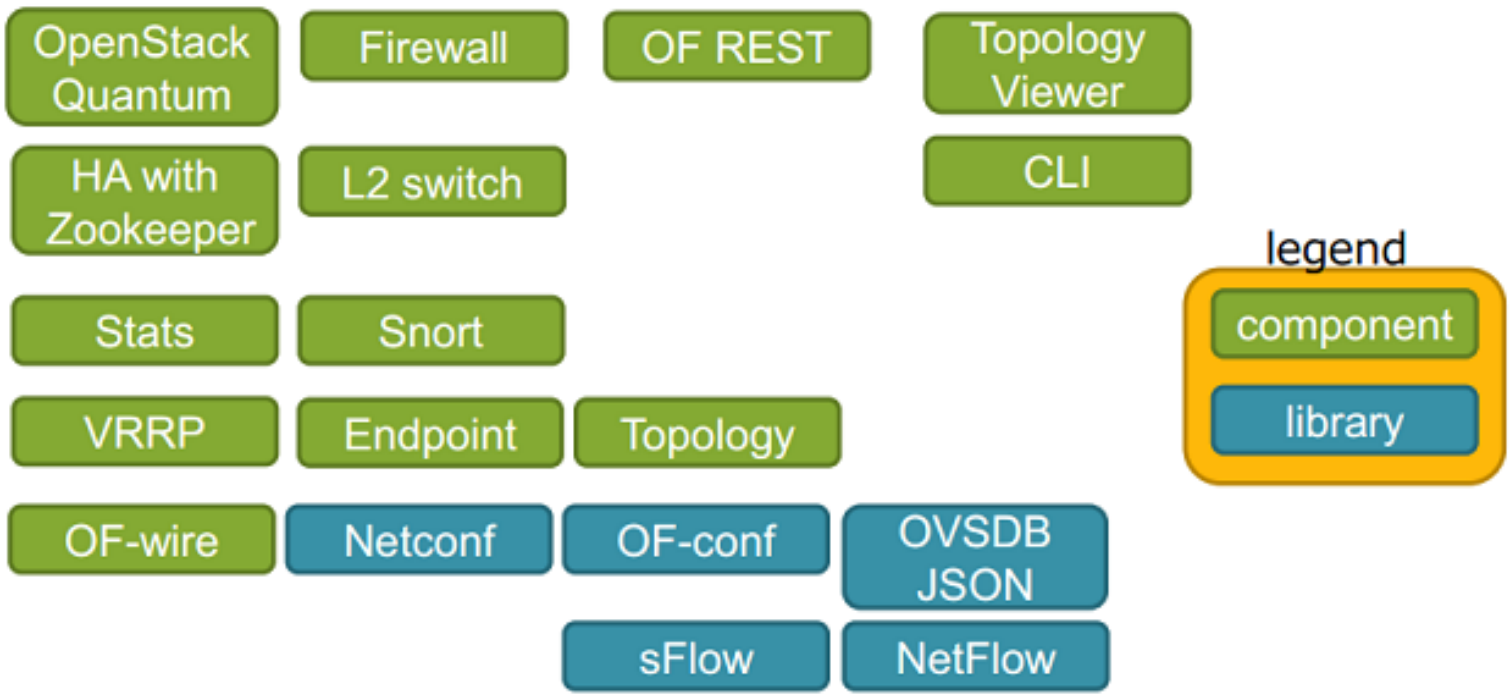
Ryu

- Disponibiliza um conjunto de componentes para criação das aplicações SDN
- Esses componentes disponibilizam:
  - Interface para controle, consulta e geração de eventos.
  - Comunicação por troca de mensagens



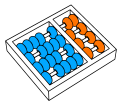
Ryu

- As componentes e bibliotecas incluídas no Ryu são:



Considerações Finais

- A arquitetura SDN está apenas iniciando, porém há um grande interesse tanto acadêmico quanto empresarial, devido as possibilidades que ela abre para o futuro das redes de computadores.
- OpenFlow foi a peça que alavancou a SDN e serve como base para vários controladores atuais.
- Porém, não é interesse dos fabricantes deixar seus equipamentos abertos para software de outra empresa.
- Consequentemente, todas estão trabalhando em seus próprios projetos.
- Mais detalhes sobre o assunto podem ser encontrados a partir das referências bibliográficas.



Referências I


D. Guedes, L. F. M. Vieira, M. M. Vieira, H. Rodrigues, and R. V. Nunes, "Redes definidas por software: uma abordagem sistêmica para o desenvolvimento de pesquisas em redes de computadores," in *Minicursos do XXX Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos - SBRC 2012*, Ouro Preto, MG, April 2012, pp. 160–210. [Online]. Available: http://sbrc2012.dcc.ufmg.br/app/p-04-f.html [Accessed: Sep. 16, 2013]

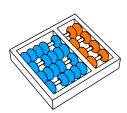

N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "Openflow: Enabling innovation in campus networks," *SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, pp. 69–74, March 2008. [Online]. Available: http://doi.acm.org/10.1145/1355734.1355746 [Accessed: Oct. 07, 2013]


L. Bertholdo, "Tecnologias, conceitos e serviços emergentes: OpenFlow," PoP-RS/UFRGS, Ouro Preto - MG, April 2012, 13º WRNP - Workshop RNP. [Online]. Available: http://www.pop-rs.rnp.br/arquivos/2012/WRNP_Openflow.pdf [Accessed: Oct. 17, 2013]


ONF, "OpenFlow - Specifications," Open Networking Foundation, 2013. [Online]. Available: https://www.opennetworking.org/sdn-resources/onf-specifications/openflow [Accessed: Sep. 30, 2013]


——, "OpenFlow Switch Specification: Version 1.4.0 (Wire Protocol 0x05)," Open Networking Foundation, October 2013. [Online]. Available: https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-spec-v1.4.0.pdf [Accessed: Oct. 20, 2013]

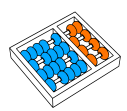

C. Marcondes, "Projeto de Desenvolvimento em OpenFlow: Tutorial de OpenFlow," Universidade Federal de São Carlos (UFSCar), Julho 2011. [Online]. Available: http://www.inf.ufes.br/~magnos/IF/if_files/Tutorial.pdf [Accessed: Oct. 18, 2013]



Referências II


OSRG, "Build SDN agilely: Ryu, a component-based software-defined networking framework," Nippon Telegraph and Telephone Corporation - Open Source Software Computing Group, 2013. [Online]. Available: http://osrg.github.io/ryu/ [Accessed: Oct. 07, 2013]


K. Ohmura, "OpenStack/Quantum SDN-based network virtulization with Ryu," Nippon Telegraph and Telephone Corporation - Open Source Software Computing Group, May 2013, 31. [Online]. Available: http://osrg.github.io/ryu/slides/LinuxConJapan2013.pdf [Accessed: Oct. 07, 2013]



Redes Definidas por Software (SDN), OpenFlow e outros Controladores de Rede

Anderson Coelho Weller

Pergunta sobre SDN

- A arquitetura SDN define uma nova forma de estruturar um sistema em rede, com isso, várias pesquisas são realizadas para aproveitar essa organização em aplicações de redes de computadores.

- Sabendo disso, pesquise um desses tipos de aplicação que pode ser melhorado com a utilização de uma SDN, e descreva sucintamente quais são as vantagens em relação a sua implementação tradicional.

- Para a pesquisa, utilize o artigo de Guedes et al. [1].