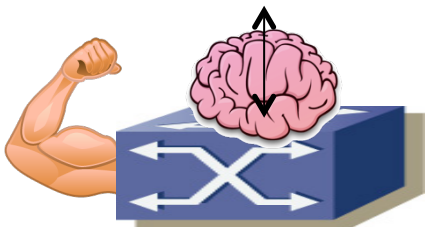
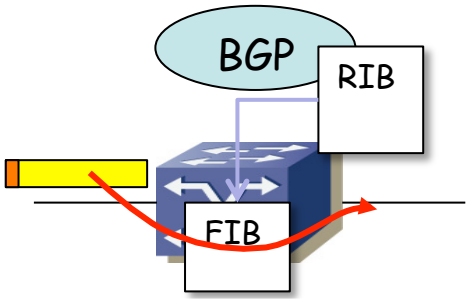
Elementos en SDN

Control vs Data Plane

- En Internet el desarrollo se hizo basado en un control distribuido
- Es decir, ambos están en el mismo equipo, implementados por el fabricante
- *Software Defined Networking* (SDN) se basa en la separación de ambos y comunicación mediante un interfaz abierto (…)
- La propuesta del Open Networking Forum (ONF) es OpenFlow
- Eso no quiere decir que SDN sea igual a OF

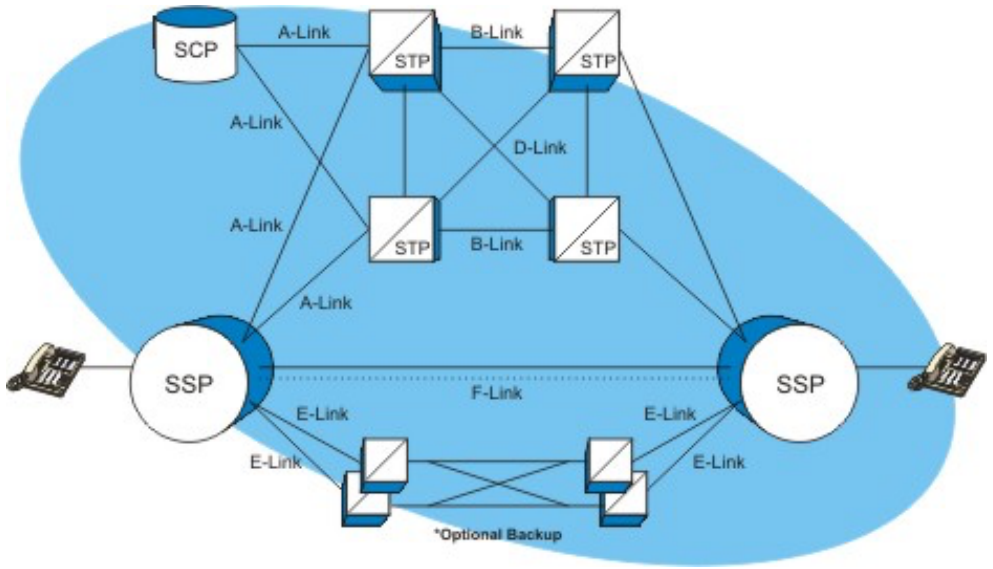
¿ OpenFlow ?





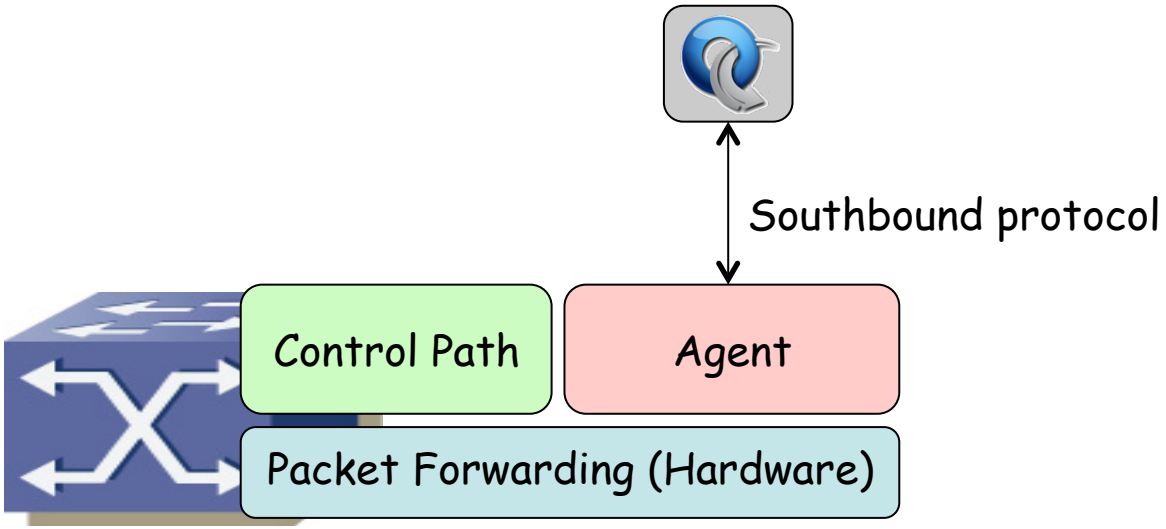
Separación control-datos

- Simplifica la evolución del control pues no está “atado” al hardware
- Permite el desarrollo de software de más alto nivel, así como su depuración, testing, simulación, etc
- La red telefónica ya tenía separado el control a elementos de señalización y control de red
- Especialmente útil en data centers y en IXPs
- Permite la optimización de los flujos
- También para una arquitectura con middleboxes
- También en el entorno WAN controlado por la misma empresa



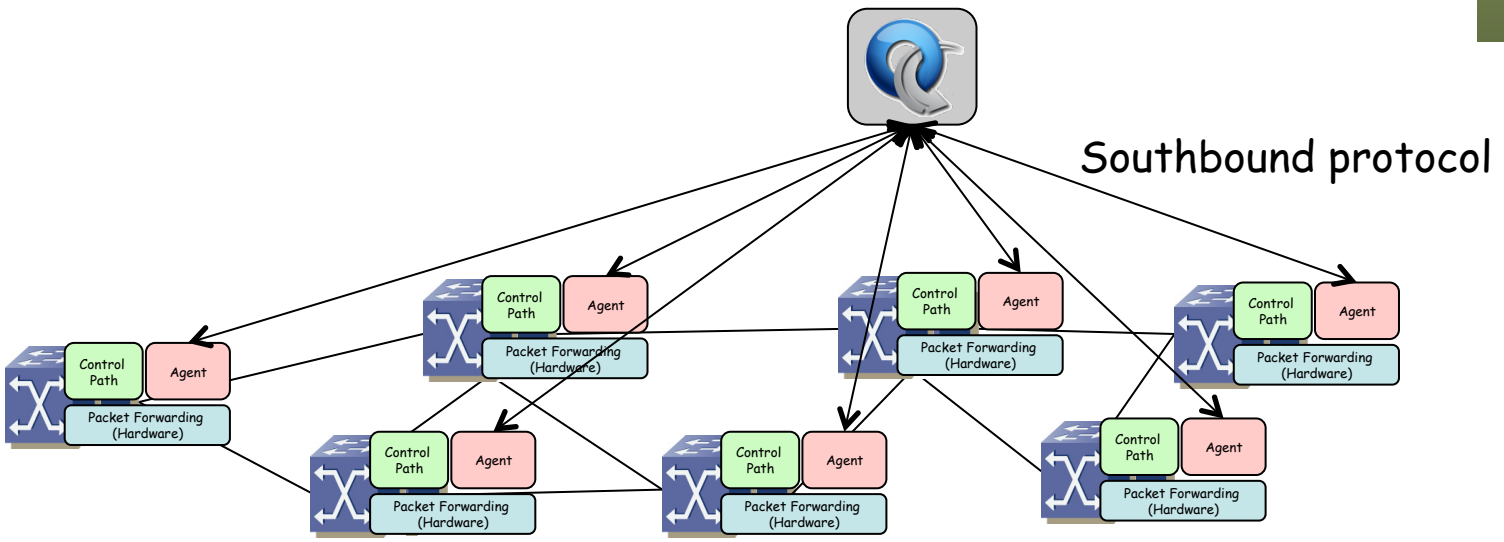
Controlador

- Esta arquitectura permite tener centralizado el plano de control
- Hoy en día el concepto de SDN no obliga a tenerlo centralizado
- Se comunica con el dispositivo mediante un *Southbound protocol*
- Para un gran número de dispositivos (…)



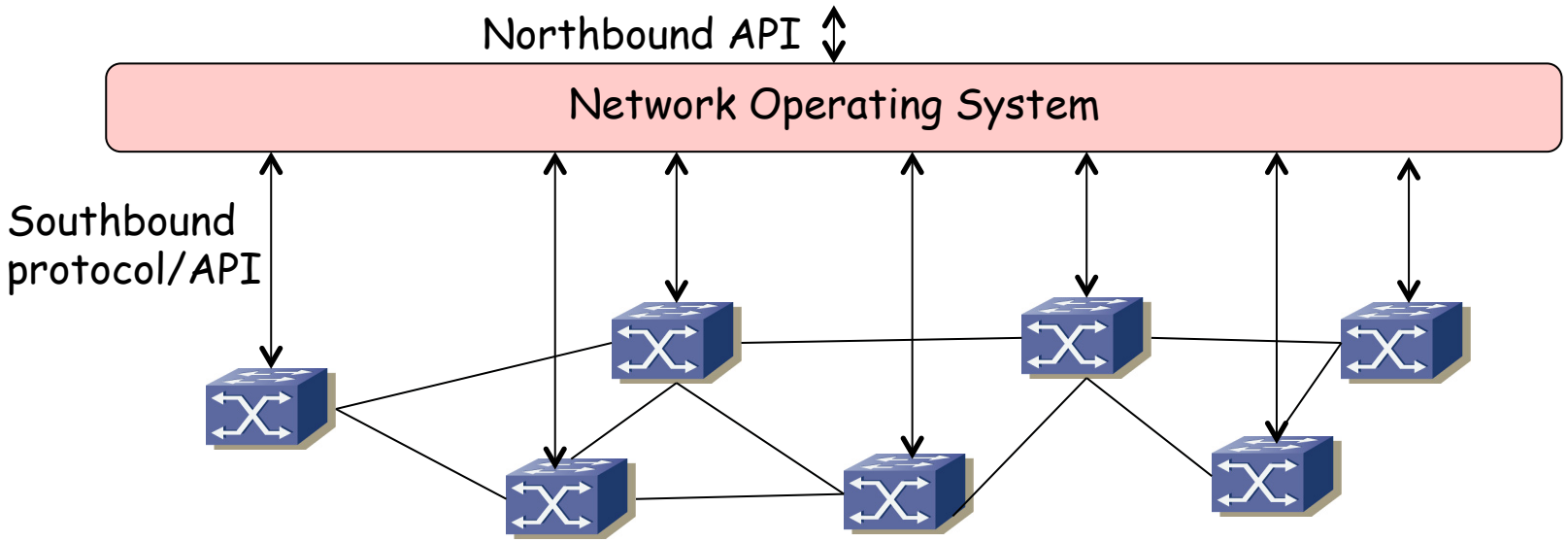
Controlador

- Esta arquitectura permite tener centralizado el plano de control
- Hoy en día el concepto de SDN no obliga a tenerlo centralizado
- Se comunica con el dispositivo mediante un *Southbound protocol*
- Para un gran número de dispositivos



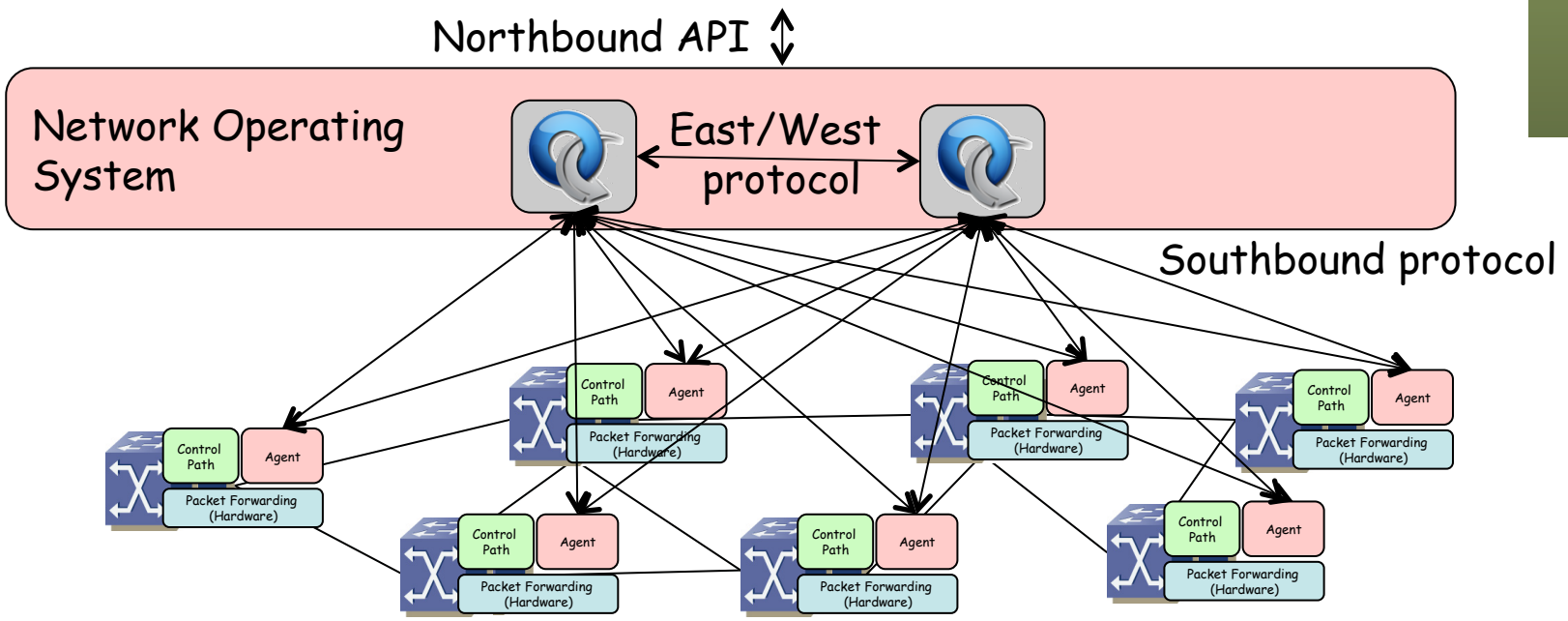
Network Operating System

- Tenemos una visión global de la red
- Mediante lo que se está viniendo a llamar un NOS
- El NOS es software en servidores que habla con los conmutadores
- El NOS da una visión virtualizada de la red, un grafo y un API ("*northbound*")
- Sobre ella podemos escribir los programas de control
- Nos aísla del hardware, igual que un OS del hardware del PC



High Availability

- Para mayor disponibilidad no tendremos un solo controlador sino varios
- La comunicación entre los controladores se lleva a cabo mediante lo que se llama un *East/West Protocol*

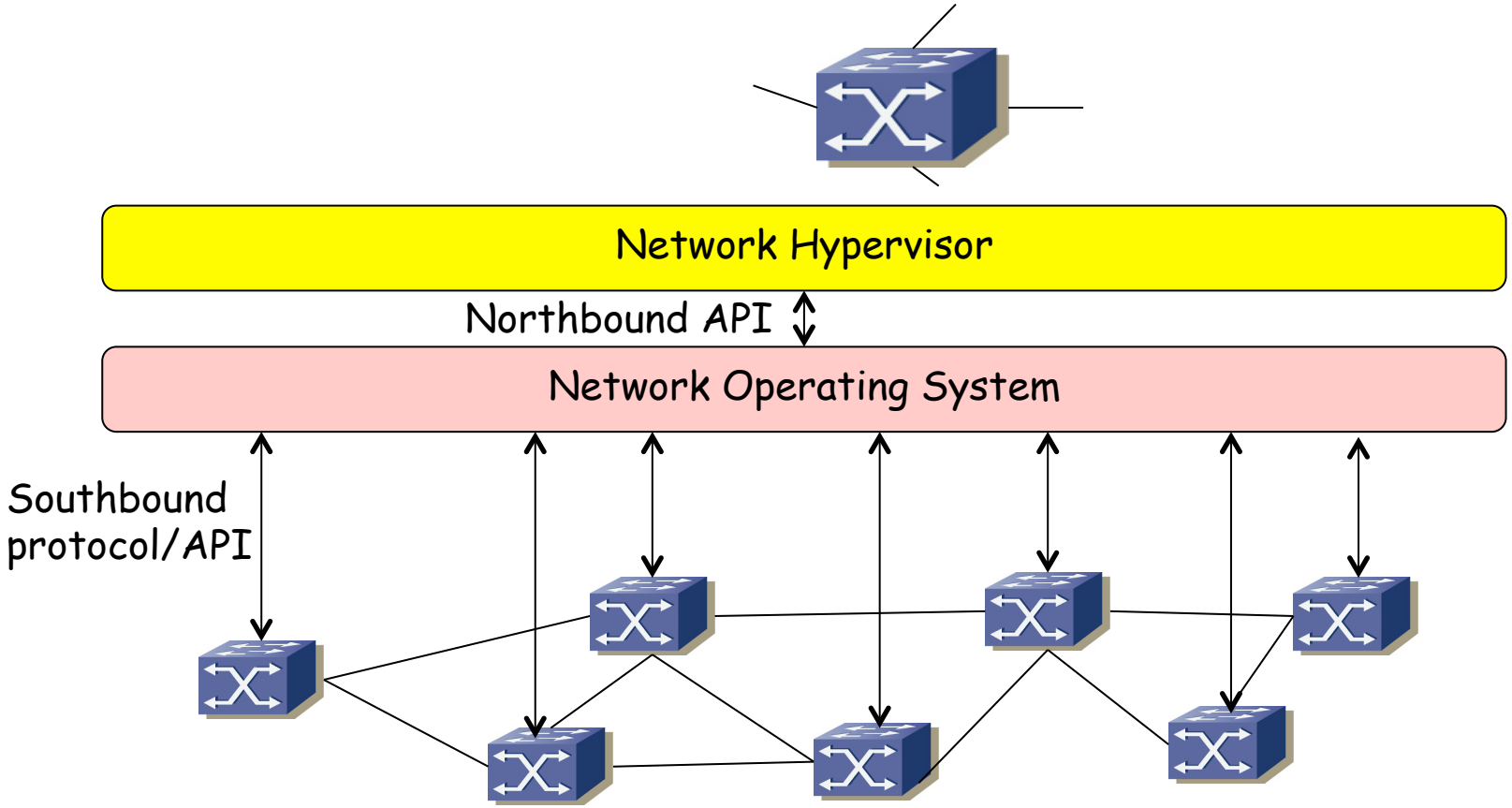


upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Redes de Nueva Generación
Área de Ingeniería Telemática

Network Virtualization

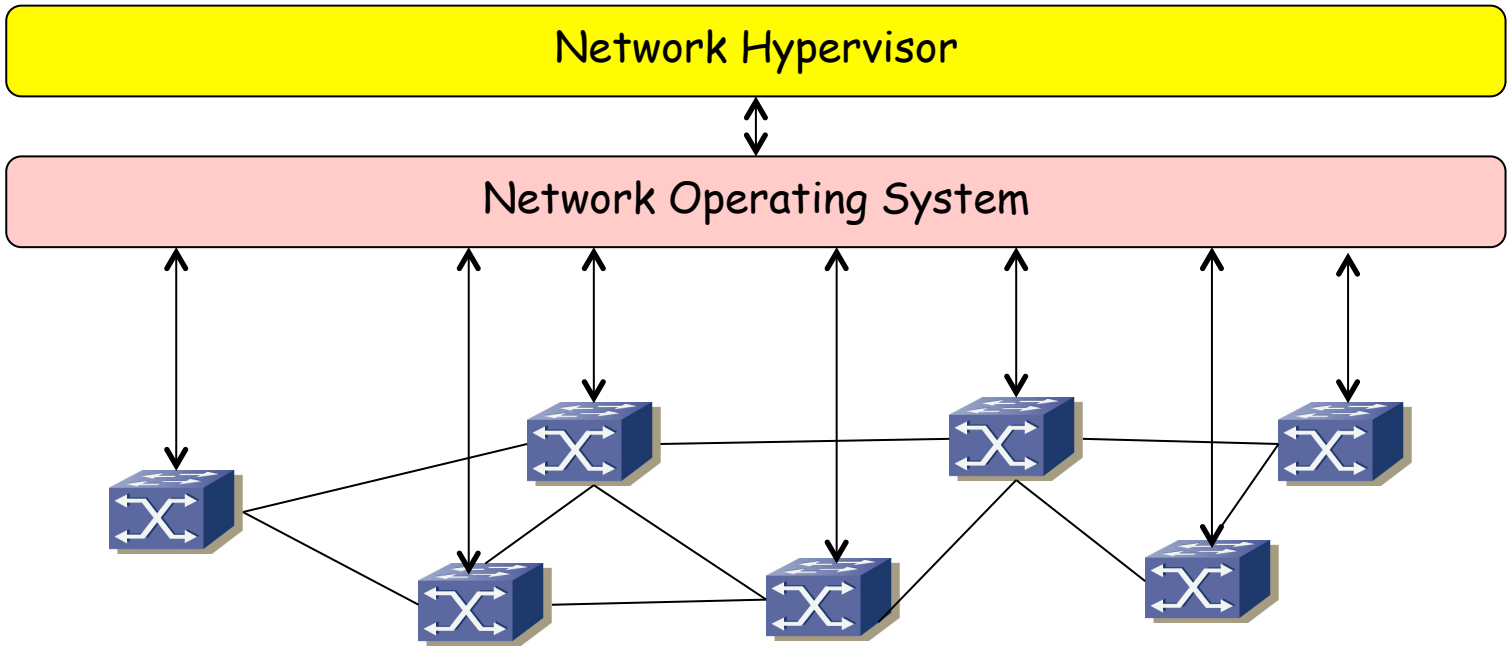
Network Virtualization

- La visión que da el NOS es virtual
- Puede ser la más adecuada para el problema que tenga que resolver el programa de control
- Sobre el NOS un *Network Hypervisor*



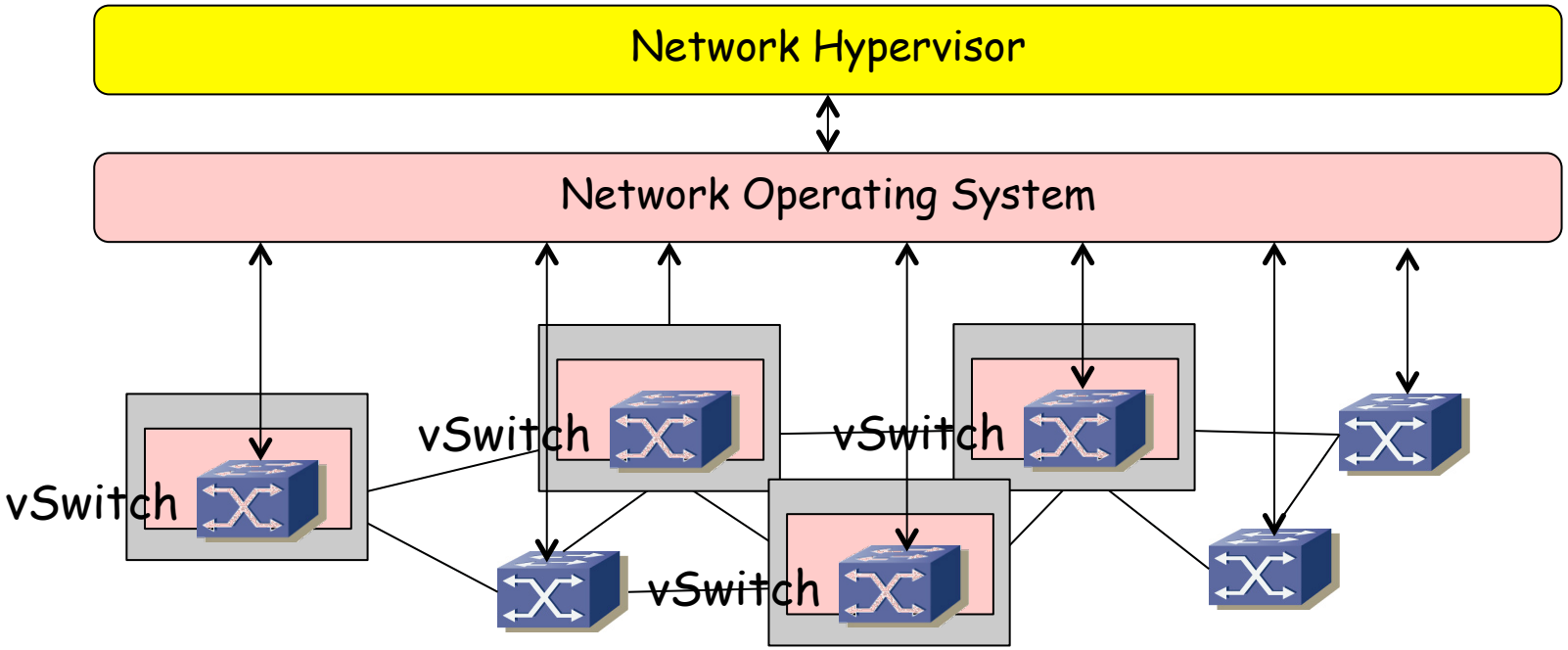
Network Virtualization

- ¿Y esos conmutadores?
- (…)



Network Virtualization

- ¿Y esos conmutadores?
- Ya no son simplemente conmutadores hardware, también vSwitches
- Hoy en día tenemos ya más puertos de hosts virtuales que físicos
- Un core x86 puede reenviar más de 20Mpps IPv4
- 1Mpps de 64bytes = 500Mbps; 1Mpps de 1518bytes = 12Gbps
- La frontera (edge) puede implementarse en software
- Podemos simplificar el core y volver el edge controlado por software



Ejemplo

- Middleboxes: lo más frecuente es que estén basados en arquitectura x86
- Están en general en el camino del tráfico
- Hacen mucho más que simple reenvío capa 2 ó 3
- Y pueden con ello
- Por cierto, ¿hay muchos?

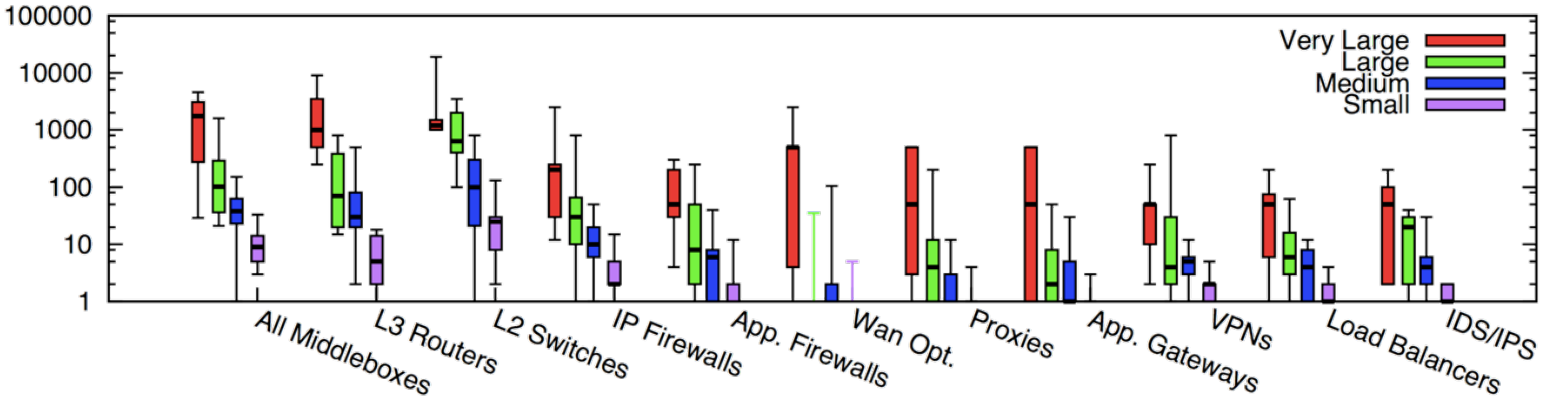


Figure 1: Box plot of middlebox deployments for small (fewer than 1k hosts), medium (1k-10k hosts), large (10k-100k hosts), and very large (more than 100k hosts) enterprise networks. Y-axis is in log scale.

J.Sherry et al., "Making Middleboxes Someone Else's Problem: Network Processing as a Cloud Service", ACM SIGCOMM 2012

OpenFlow

OpenFlow

- Su origen en proyectos de investigación en la Universidad de Stanford
- En 2011 se funda el consorcio ONF
  - *Open Networking Foundation*
  - https://www.opennetworking.org
  - Más de 140 empresas (fabricantes, operadoras, ISPs, startups, etc)
- OpenFlow es un protocolo “southbound”
- No hace “nada” sin una aplicación que lo emplee

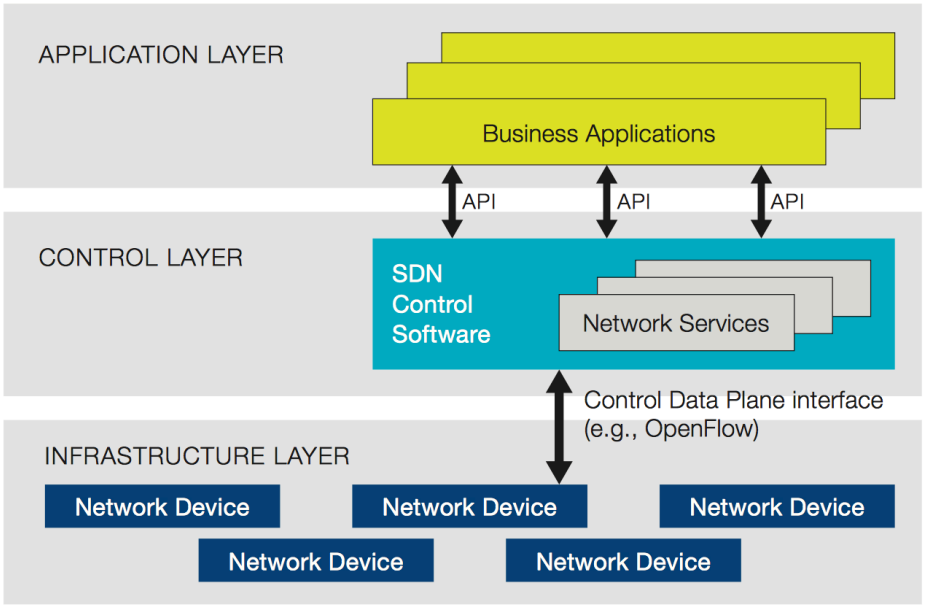
OPEN NETWORKING FOUNDATION

Martin Casado

ONF y SDN

- "The aim of SDN is to provide open interfaces that enable the development of software that can control the connectivity provided by a set of network resources and the flow of network traffic though them, along with possible inspection and modification of traffic that may be performed in the network."
- "In the SDN architecture, the control and data planes are decoupled, network intelligence and state are logically centralized, and the underlying network infrastructure is abstracted from the applications. "

OpenFlow



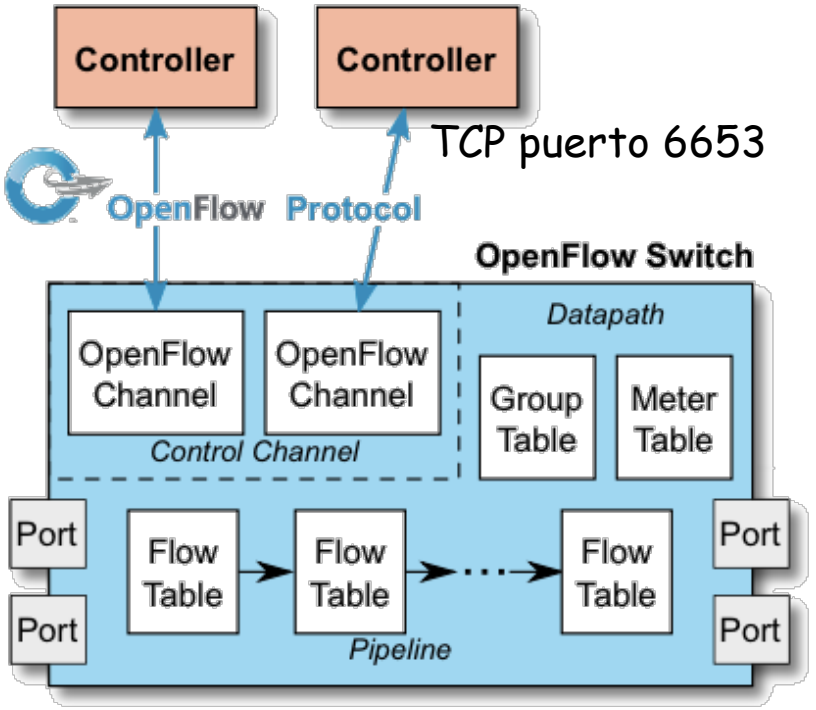
https://www.opennetworking.org/images/stories/downloads/sdn-resources/white-papers/wp-sdn-newnorm.pdf

OpenFlow

- Dos tipos de conmutadores:
  - *OpenFlow-only*: solo soportan el modo de funcionamiento OpenFlow
  - *OpenFlow-hybrid*: también soportan funcionamiento "normal" (conmutación L2, conmutación L3, VLANs, ACLs, etc)
  - Los híbridos deberán tener alguna forma de clasificar si los paquetes pasan por procesado "normal" u OpenFlow

Flow Tables

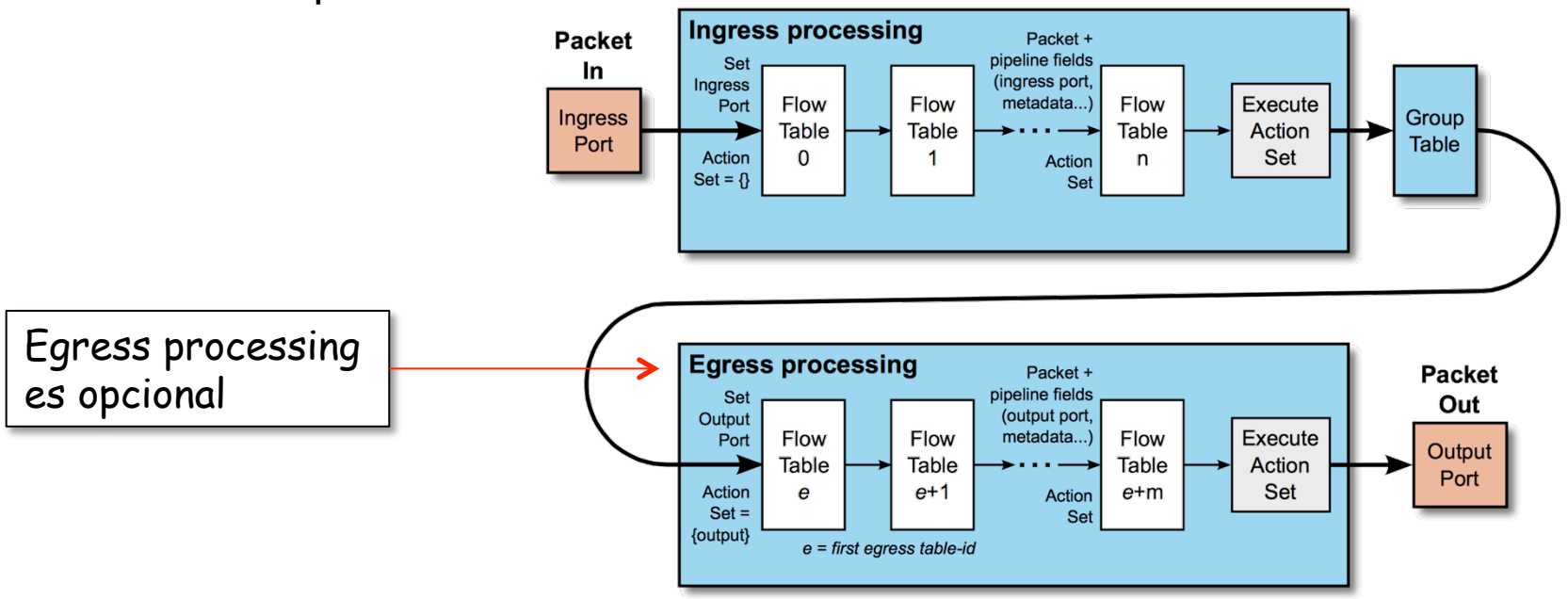
- Contienen la información sobre los campos a comprobar (*match fields*) en los paquetes y qué hacer con ellos
- El controlador puede añadir, modificar y borrar entradas empleando OF
- Las "acciones" son las operaciones en caso de que el paquete verifique la entrada en la tabla
- Puede reenviar el paquete, mandárselo al controlador, pasarlo a otra tabla, actualizar contadores, etc



https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-switch-v1.5.0.noipr.pdf

OpenFlow pipeline

- Debe tener al menos una tabla aunque pueden ser más (desde 1.1, permite procesado de etiquetas MPLS)
- Hay procesado a la entrada del paquete (al menos una tabla)
- Si se decide reenviarlo pasa por tablas de salida (desde 1.5)
- Las tablas se comprueban en orden
- Si el paquete verifica una regla se ejecuta la acción que indique
- Si no verifica ninguna es un *"table miss"* y hay una acción por defecto en la tabla para este caso



https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-switch-v1.5.0.noipr.pdf

Acciones

- Incluimos aquí la acción por defecto para el caso de *"table miss"*
- La acción puede ser pasar a otra tabla posterior (no anterior)
- Puede ser hacer inundación
- O reenviar por un puerto en concreto
- O puede ser reenviar el paquete al controlador (dentro de un mensaje OF)
- O pasar el paquete a un reenvío tradicional si es un conmutador híbrido
- O modificar campos de cabeceras del paquete (una modificación afecta a las comprobaciones en egress tables)
- etc

Entradas en las tablas

- *Match Fields:*
  - Puede valer ANY (comodín) o soportarse bitmasks
  - Hasta la versión 1.1 se miraban ciertos campos:
    - Puerto de entrada, metadatos provenientes de tabla anterior
    - Direcciones MAC origen y destino, Ethertype, VLAN ID, PCP
    - Etiqueta MPLS, TC
    - Direcciones IP origen y destino, protocolo, ToS
    - Puertos origen y destino TCP/UDP/SCTP
    - Tipo y código ICMP
  - Otros que se han ido añadiendo:
    - Bits ECN
    - Flags TCP
    - Código de opción de ARP, direcciones MAC e IP en el mensaje ARP
    - Direcciones IPv6, flow label IPv6, tipo y código ICMPv6
    - Etc
  - (…)

Match Fields	Priority	Counters	Instructions	Timeouts	Cookie	Flags

Entradas en las tablas

- Prioridad:
  - Pueden verificarse varias entradas de la tabla
  - En ese caso se selecciona solo la de mayor prioridad
- (…)

Match Fields	Priority	Counters	Instructions	Timeouts	Cookie	Flags

Entradas en las tablas

- Contadores:
  - Se actualizan cuando la entrada es seleccionada
- (…)

Counter	**Bits**	
Per Flow Table		
Reference Count (active entries)	32	*Required*
Packet Lookups	64	*Optional*
Packet Matches	64	*Optional*
Per Flow Entry		
Received Packets	64	*Optional*
Received Bytes	64	*Optional*
Duration (seconds)	32	*Required*
Duration (nanoseconds)	32	*Optional*
Per Port		
Received Packets	64	*Required*
Transmitted Packets	64	*Required*
Received Bytes	64	*Optional*
Transmitted Bytes	64	*Optional*
Receive Drops	64	*Optional*
Transmit Drops	64	*Optional*
Receive Errors	64	*Optional*
Transmit Errors	64	*Optional*
Receive Frame Alignment Errors	64	*Optional*
Receive Overrun Errors	64	*Optional*
Receive CRC Errors	64	*Optional*
Collisions	64	*Optional*
Duration (seconds)	32	*Required*
Duration (nanoseconds)	32	*Optional*
Per Queue		
Transmit Packets	64	*Required*
Transmit Bytes	64	*Optional*
Transmit Overrun Errors	64	*Optional*
Duration (seconds)	32	*Required*
Duration (nanoseconds)	32	*Optional*
Per Group		
Reference Count (flow entries)	32	*Optional*
Packet Count	64	*Optional*
Byte Count	64	*Optional*
Duration (seconds)	32	*Required*
Duration (nanoseconds)	32	*Optional*
Per Group Bucket		
Packet Count	64	*Optional*
Byte Count	64	*Optional*
Per Meter		
Flow Count	32	*Optional*
Input Packet Count	64	*Optional*
Input Byte Count	64	*Optional*
Duration (seconds)	32	*Required*
Duration (nanoseconds)	32	*Optional*
Per Meter Band		
In Band Packet Count	64	*Optional*
In Band Byte Count	64	*Optional*

Entradas en las tablas

- *Instructions:*
  - Cambio al paquete, acciones, etc, cuando se selecciona la entrada
  - Las hay de implementacion requerida y opcional
  - Ejemplos:
    - Enviar a un puerto de salida, descartar, asignar cola en el puerto out
    - Añadir/retirar etiquetas (MPLS, VLAN, PBB)
    - Modificar valor de un campo de cabecera
- (…)

Match Fields	Priority	Counters	Instructions	Timeouts	Cookie	Flags

Entradas en las tablas

- *Timeouts:*
  - Máximo tiempo inactiva antes de expirar
- (…)

Match Fields	Priority	Counters	Instructions	Timeouts	Cookie	Flags

Entradas en las tablas

- *Cookie*:
  - Ahí el controlador puede guardar un valor
  - El switch no lo emplea para nada
- (…)

Match Fields	Priority	Counters	Instructions	Timeouts	Cookie	Flags

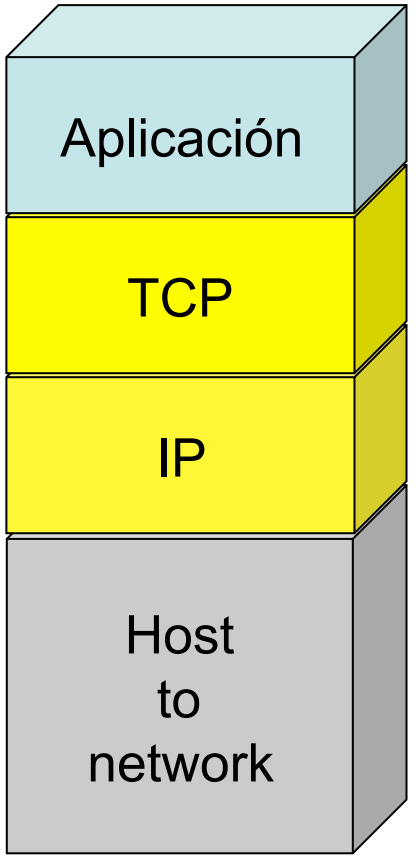
Entradas en las tablas

- *Flags*:
  - Diferentes opciones
  - Ejemplo:
    - Que envíe un mensaje al controlador al eliminarse o expirar una entrada
    - Que no lleve contadores de bytes o de paquetes

Match Fields	Priority	Counters	Instructions	Timeouts	Cookie	Flags

El protocolo

- TCP (puerto 6653), opcionalmente empleando TLS
- Hay mensajes:
  - De controlador a conmutador (…)
  - Asíncronos (desde el conmutador)
  - Simétricos



El protocolo

- TCP (puerto 6653), opcionalmente empleando TLS
- Hay mensajes:
  - De controlador a conmutador
    - Petición de capacidades
    - Establecer o preguntar por configuración o estado
    - Entregarle un paquete para enviar por un puerto
  - Asíncronos (desde el conmutador) (…)
  - Simétricos

El protocolo

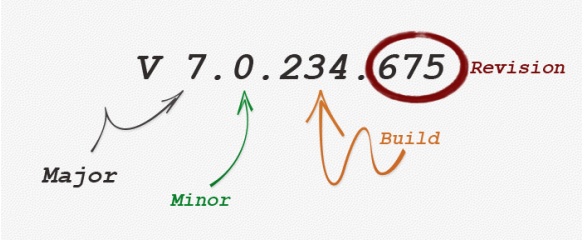
- TCP (puerto 6653), opcionalmente empleando TLS
- Hay mensajes:
  - De controlador a conmutador
  - Asíncronos (desde el conmutador)
    - Envío al controlador de un paquete recibido
    - Notificación de entrada en tabla eliminada
    - Notificación de cambio de estado de un puerto
  - Simétricos (…)

El protocolo

- TCP (puerto 6653), opcionalmente empleando TLS
- Hay mensajes:
  - De controlador a conmutador
  - Asíncronos (desde el conmutador)
  - Simétricos
    - Hello, al establecer la conexión
    - Echo, para comprobar que el otro extremo está vivo y tal vez para medir latencia o bw
    - Error

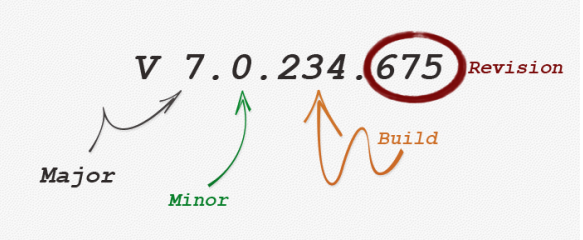
Versiones

- https://www.opennetworking.org/sdn-resources/technical-library
- Versión 1.5.1 Abril de 2015
- Probablemente OF 1.0 sea lo más implementado en hardware
- Las siguientes versiones han ido introduciendo mejoras, más flexibilidad, pero también haciéndolo más complejo
- OF 1.1
  - Múltiples tablas
  - Soporte de acciones para MPLS (soporta multi-etiqueta)
  - Acciones sobre el TTL
  - Soporte de VLANs en QinQ
  - Soporte para agrupar puertos de cara a acciones
- (…)



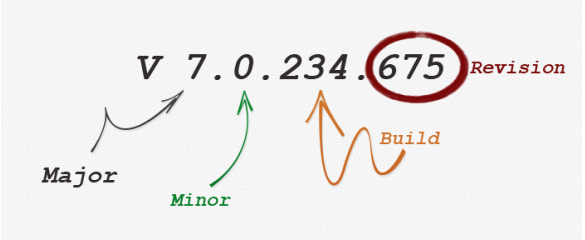
Versiones

- https://www.opennetworking.org/sdn-resources/technical-library
- Versión 1.5.1 Abril de 2015
- Probablemente OF 1.0 sea lo más implementado en hardware
- Las siguientes versiones han ido introduciendo mejoras, más flexibilidad, pero también haciéndolo más complejo
- OF 1.1
- OF 1.2
  - Soporte de campos de IPv6, ICMPv6, ND
  - Mejora la extensibilidad de las reglas de *match*
- (…)



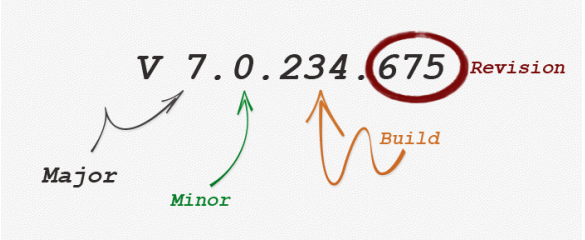
Versiones

- https://www.opennetworking.org/sdn-resources/technical-library
- Versión 1.5.1 Abril de 2015
- Probablemente OF 1.0 sea lo más implementado en hardware
- Las siguientes versiones han ido introduciendo mejoras, más flexibilidad, pero también haciéndolo más complejo
- OF 1.1
- OF 1.2
- OF 1.3.x
  - *Meters* por flujo (limitadores para QoS)
  - Soporte de PBB
- (…)



Versiones

- https://www.opennetworking.org/sdn-resources/technical-library
- Versión 1.5.1 Abril de 2015
- Probablemente OF 1.0 sea lo más implementado en hardware
- Las siguientes versiones han ido introduciendo mejoras, más flexibilidad, pero también haciéndolo más complejo
- OF 1.1
- OF 1.2
- OF 1.3.x
- OF 1.4
  - Mayor extensibilidad
  - Soporte de puertos ópticos (frecuencias, potencia, etc)
- (…)

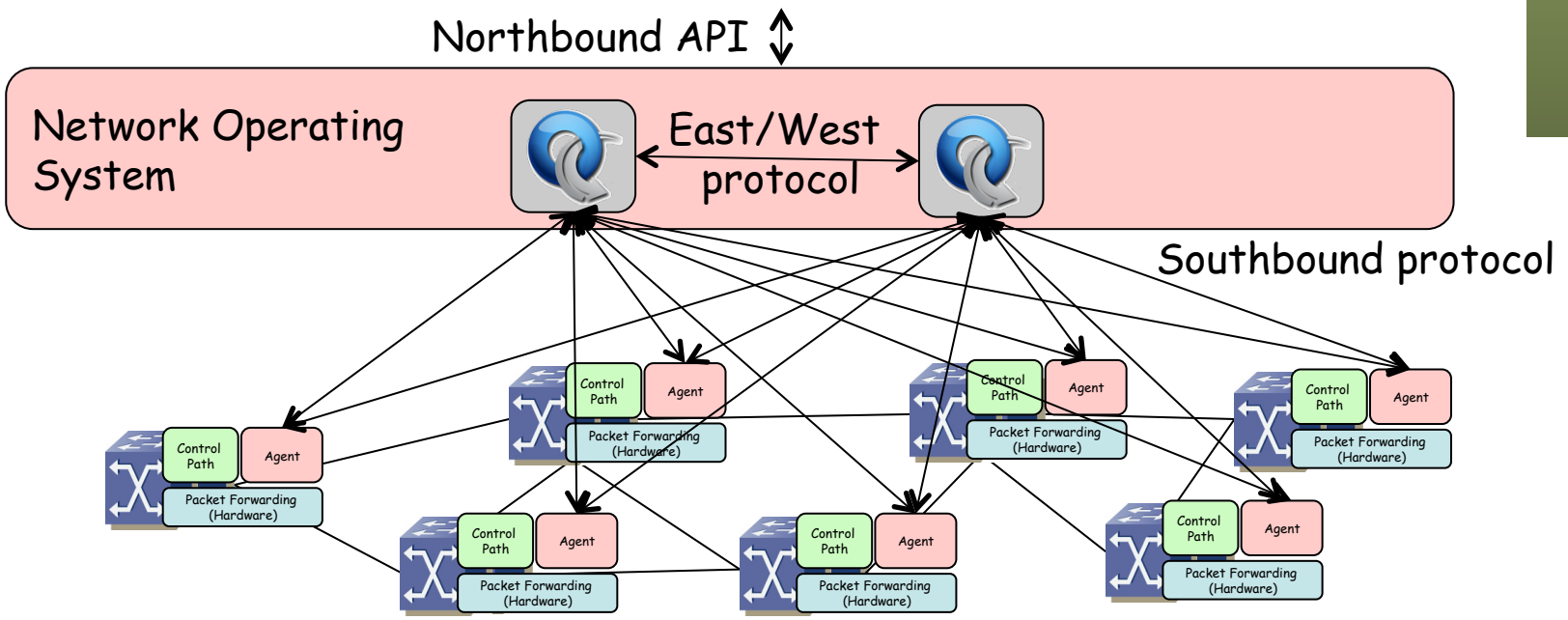


Versiones

- https://www.opennetworking.org/sdn-resources/technical-library
- Versión 1.5.1 Abril de 2015
- Probablemente OF 1.0 sea lo más implementado en hardware
- Las siguientes versiones han ido introduciendo mejoras, más flexibilidad, pero también haciéndolo más complejo
- OF 1.1
- OF 1.2
- OF 1.3.x
- OF 1.4
- OF 1.5
  - *Egress tables*
  - Soporte para más que Ethernet
  - Flags TCP

APIs

- OpenFlow es un *Southbound API*
- El ONF asocia OpenFlow a SDN pero una SDN no necesita emplear necesariamente OpenFlow
- Podríamos considerar OF a día de hoy el API south estándar
- No hay *Northbound API* estandarizada, ni *de facto*
- No hay *East/West API* estandarizada

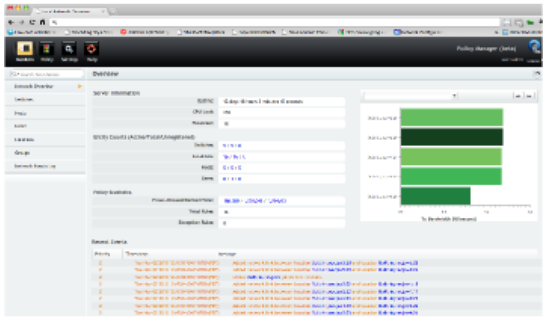


Controladores

- NOX
  - http://www.noxrepo.org
  - Desarrollado por Nicira, cedido el código en 2008
  - Ofrece un API C++ para OF 1.0
  - Muchos otros heredan de su código
  - Incluye componentes de ejemplo para descubrir la topología, implementar un puente transparente y un switch distribuido
  - Open Source
- POX
  - Hereda de NOX
  - Permite el desarrollo en Python
  - Open Source
- Beacon
  - https://openflow.stanford.edu/display/Beacon/Home
  - Java (desarrollo con eclipse)
  - Open Source

Controladores

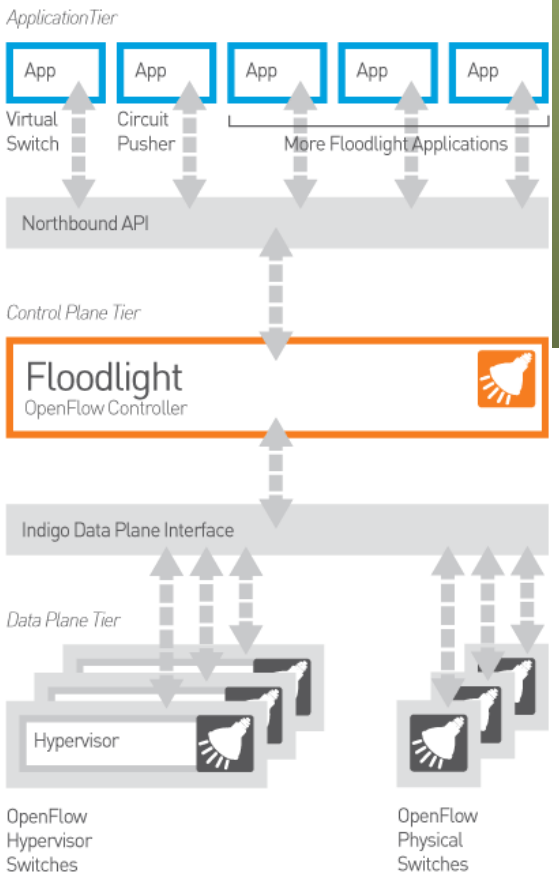
- SNAC
  - http://www.openflowhub.org/display/Snac/SNAC+Home
  - Incluye GUI web
  - Incluye un lenguaje de definición de políticas
  - Open Source



- FloodLight
  - http://www.projectfloodlight.org/floodlight/
  - Basado en Java (basado en Beacon)
  - Apoyado por Big Switch Networks
  - Lo emplean para construir su controlador
  - Open Source

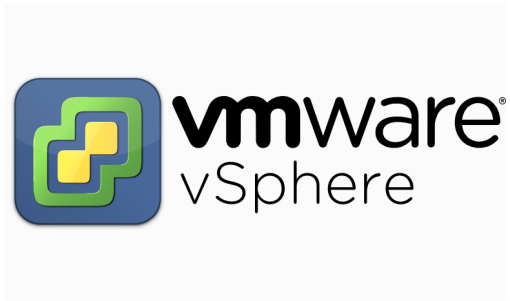
Guido Appenzeller

big switch networks



VMware

- Controlador propietario
- vCenter Server controla los VDS (Virtual Distributed Switches)
- Otros componentes: vSphere, vCloud Director, vCloud Networking and Security, vCloud Automation Center, vCenter Site Recovery Manager, vCenter Operations Management Suite, vFabric Application Director for Provisioning
- Máximos vSphere 6.0:
  - 1024 VMs por host
  - 10 vNICs por VM
  - 1000 hosts por VDS
  - 1016 puertos de VDS activos por host
  - 60.000 puertos por VDS
  - 1000 hosts, 10.000 VMs en funcionamiento y 128 VDS por vCenter
  - 65.536 direcciones MAC por vCenter
  - 4/8 operaciones vMotion simultáneas por host por NIC 1/10Gbps
  - 16 VDS por host
  - etc



http://www.vmware.com/pdf/vsphere6/r60/vsphere-60-configuration-maximums.pdf

Nicira

- Fundada en 2007
- Miembro fundador del ONF
- En 2011 empieza a distribuir su NVP (*Network Virtualization Platform*)
- Es un controlador para OVS (Open vSwitch)
- No emplea solo OF sino OVSDB (Open vSwitch DataBase Management Protocol)
- Adquirida en 2013 por VMware (por unos 1260 millones de $)

Martin Casado

OvS
Open vSwitch

Otro software

- Frameworks
  - Onix, Trema, Maestro, Ryu
  - Indigo (para añadir OF a switches)
- FlowVisor:
  - https://github.com/OPENNETWORKINGLAB/flowvisor/wiki
  - Actúa como un proxy entre los switches y los controladores OF
  - Permite repartir recursos de la red entre varios controladores
- Avior, Oflops, Cbench, Twister, FortNOX, LINC, Pantou, Of13softwitch, Cisco OnePK, Plexxi, etc etc etc

- ¡Se abrió la veda al software!