Contenidos

Universidad Rey Juan Carlos

1. Redes y enlaces inalámbricos
   - Redes LAN inalámbricas 802.11 (Wifi)
   - Redes inalámbricas personales 802.15 (Bluetooth o Zigbee)
   - Redes móviles
   - Redes vía satélite
2. Redes definidas por software (Software Defined Networks)

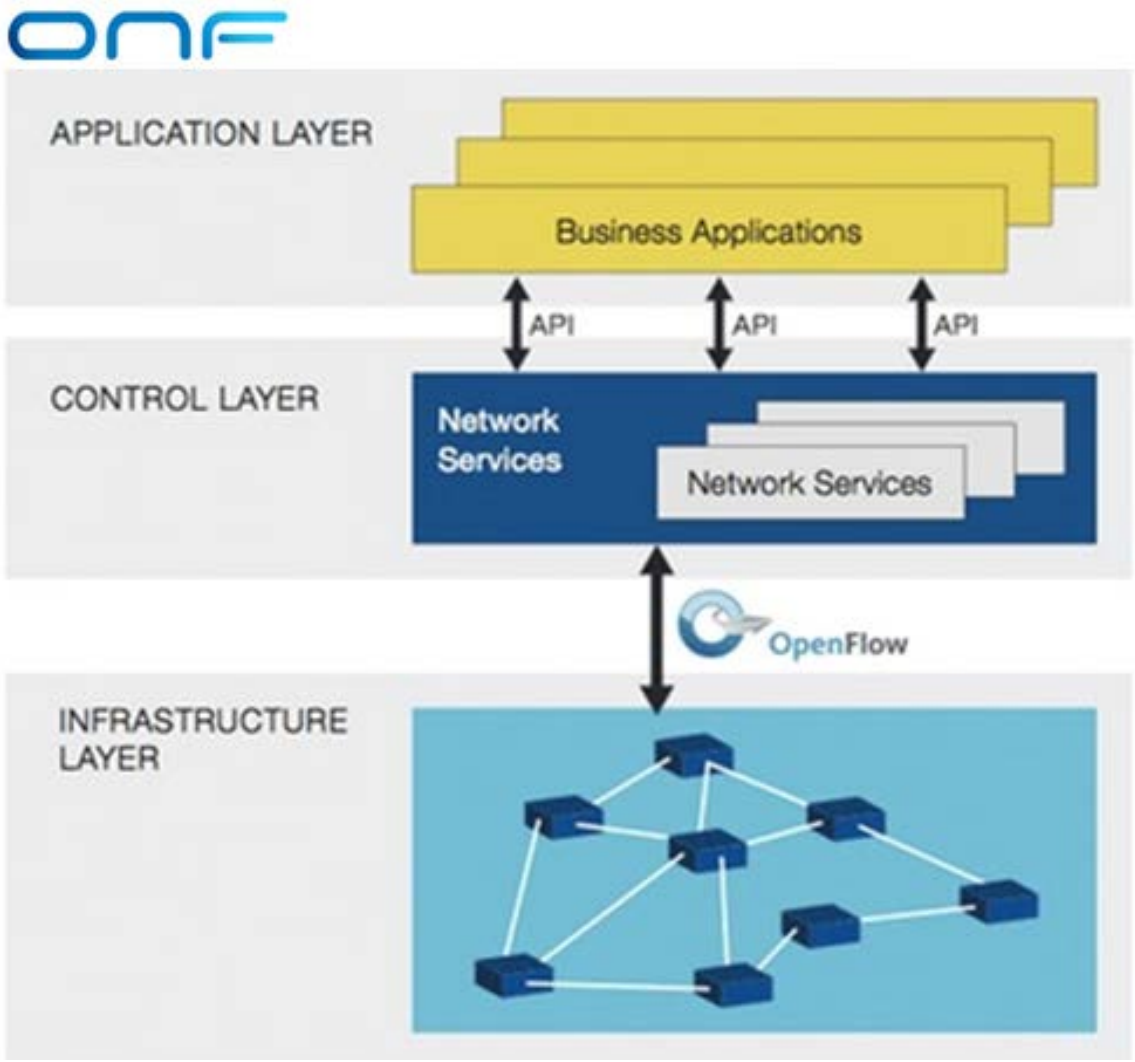
Materiales basados principalmente en:

- Kurose-Ross. Redes de Computadoras. 7ª ed. 2017. ©
- N. Feamster. Software Defined Networking Course.
- Intel Network Academy. Network transformation 101

Qué es una Software Defined Network (SDN)

- Arquitectura de red que separa el plano de datos (dispositivos de red que reenvían el tráfico) del plano de control (lógica software que controla cómo fluye el tráfico)
- Ventaja: Se controla el comportamiento de la red desde un único programa de control a alto nivel
- Dos partes en la infraestructura:
  - Plano de control: controladores y apps (SW)
  - Plano de datos: switches programables (HW)

2. Redes definidas por software (SDN)



- OpenFlow
- ONOS
- OpenDaylight
- POX
- RYU

- OpenFlow
- NetConf
- BGP-LS
- SNMP

Del enfoque tradicional a las SDN

La capa de red tiene dos funciones principales:

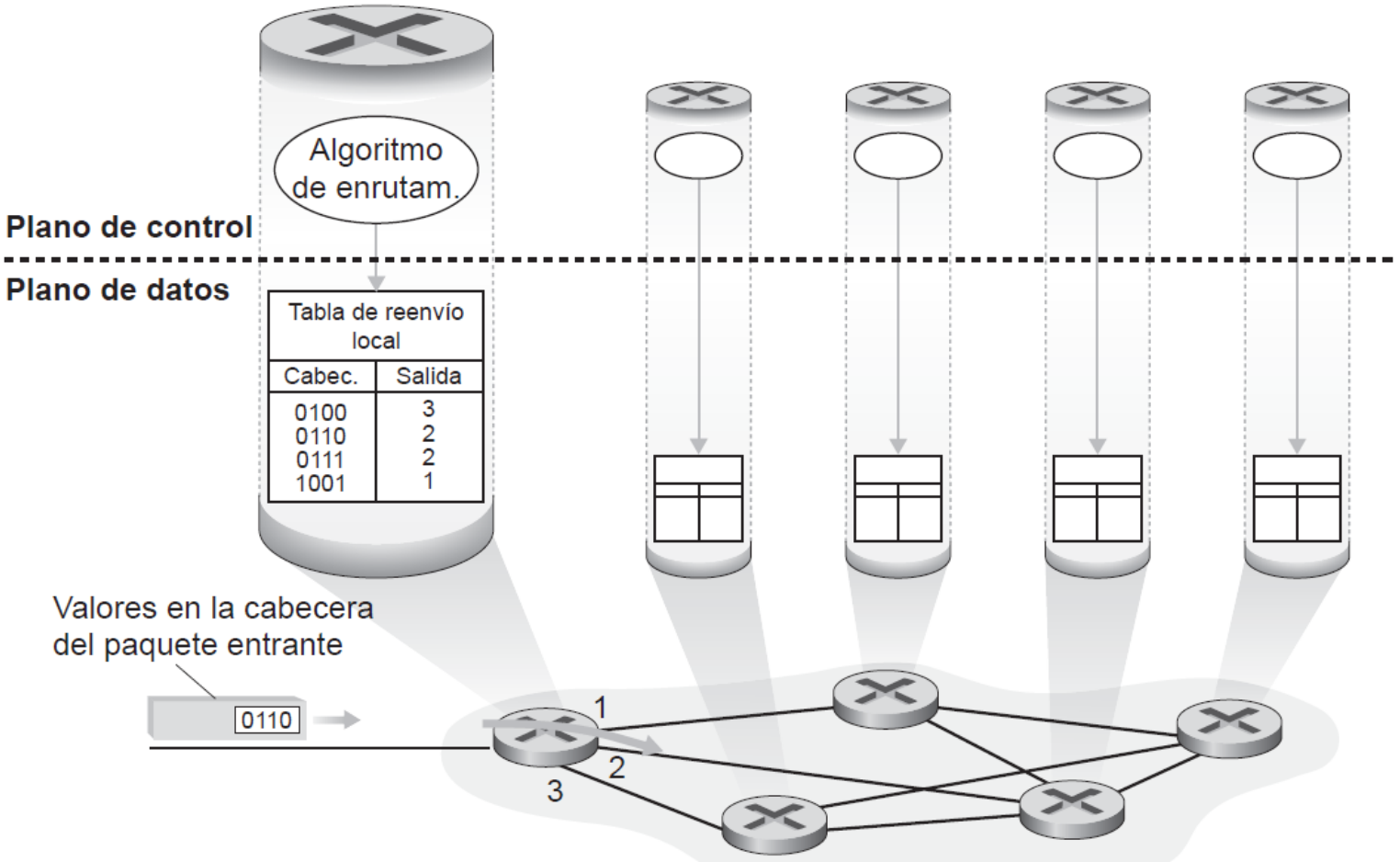
- Reenvío (forwarding): mover paquetes de la entrada a la salida apropiada del router
- Enrutamiento (routing): determinar la ruta de origen a destino para los paquetes

Podemos dividir la capa de red en dos planos:

- Plano de datos (data plane): se encarga del reenvío y lo hace localmente para cada router
- Plano de control (control plane): se encarga de la lógica de enrutamiento
  - Tradicionalmente implementado en los routers con algoritmos de enrutamiento
  - Ahora con SDN implementado en servidores remotos

Del enfoque tradicional a las SDN

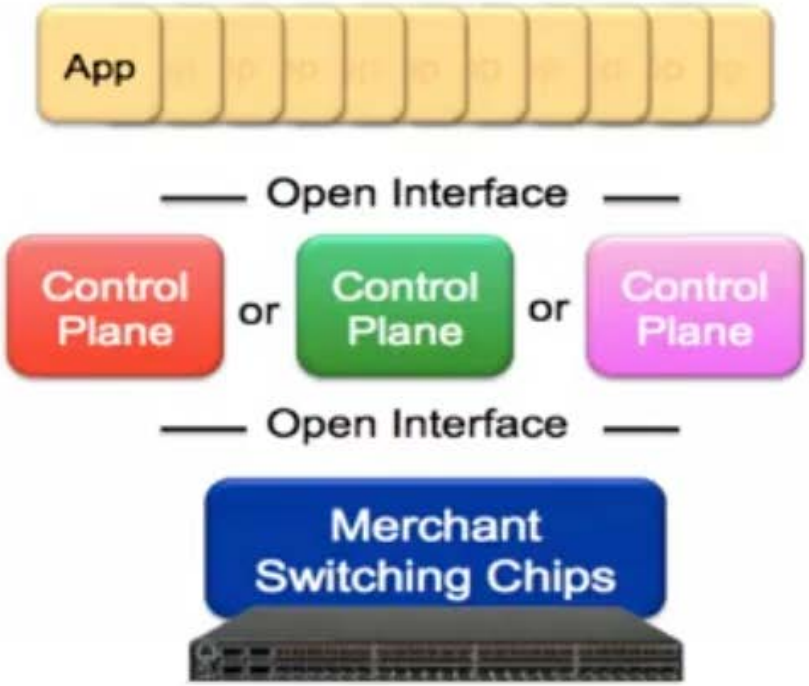
- Separación en la capa de red de los planos de datos y control



Planos de control y de datos

- Plano de control:
  - El cerebro de la red, que controla el comportamiento de la red
  - Puede funcionar separado de los dispositivos de red
  - Es la lógica que determina cómo se reenvía el tráfico
- Plano de datos:
  - Hardware programable controlado por el plano de control

2. Redes definidas por software (SDN)



Vertically integrated
Closed, proprietary
Slow innovation

Horizontal
Open interfaces
Rapid innovation

Fuente: Nick McKeown

Historia de SDN

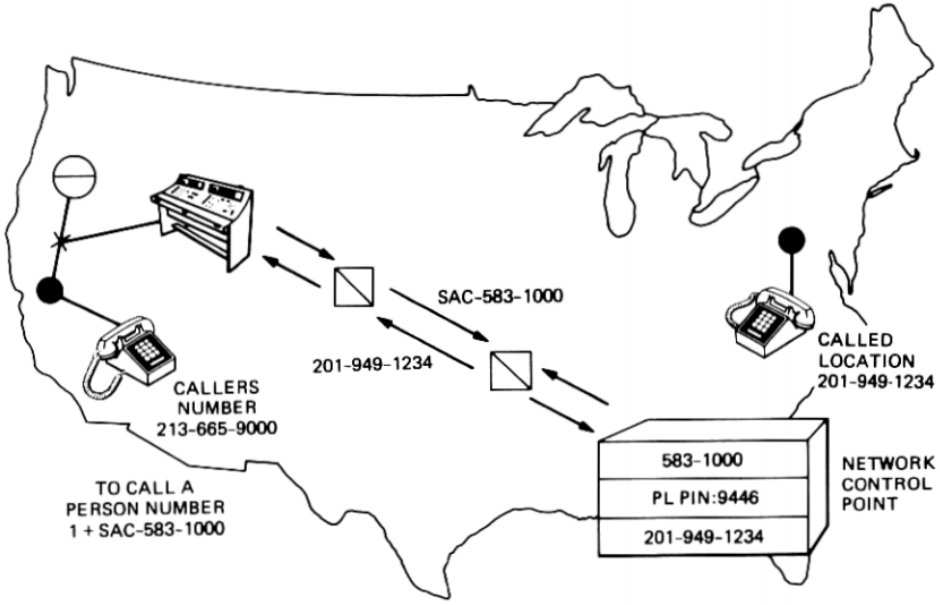
¿Por qué SDN?

- La configuración descentralizada es compleja e impredecible (y más con el incremento de dispositivos conectados y de tráfico generado)
- Solución: Intentar controlar el comportamiento de la red desde un punto central de control
- Ventajas:
  - Más fácil de coordinar comportamientos entre distintos dispositivos de red (ej. políticas del balanceador vs seguridad)
  - Más sencillo evolucionar e innovar sin atarse al hardware
  - Control por software, un programa es más sencillo de entender, incluso podemos aplicar metodologías de programación
  - Más flexibilidad, al ser más sencillo introducir nuevos servicios

Historia de SDN

Universidad Rey Juan Carlos

Un punto único de control centralizado (Network Control Point, NCP) lo introdujo AT&T en su red de telefonía en 1981. Le permitía desplegar nuevos servicios rápidamente y gestionar su red más eficientemente. Por ejemplo, el servicio de Person Locator



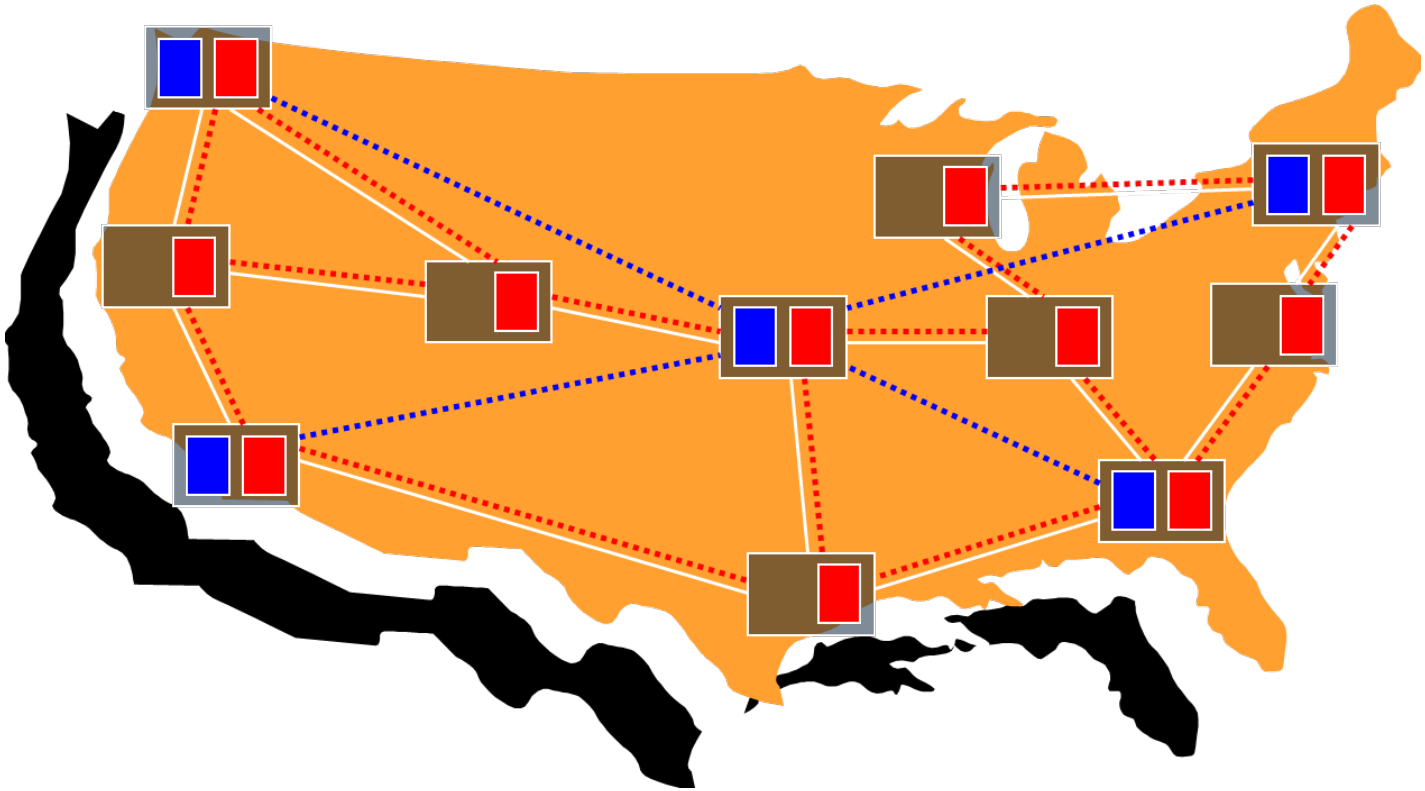
Historia de SDN

En los 90s aparecieron las *active networks*, donde los switches realizaban determinadas acciones sobre los paquetes, más allá de la mera redirección. Funciones de firewall, de proxie, etc.

Dos aproximaciones:

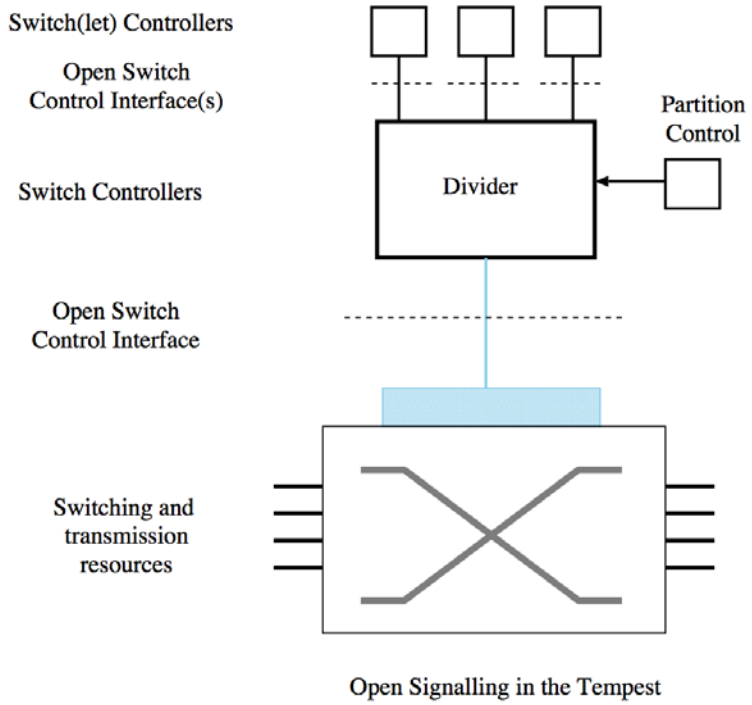
- Cápsulas (código integrado completamente en la cabecera del paquete)
- Switches programables (código en el switch, el paquete indica en su cabecera qué hacer)

También en los 90s se comenzó a virtualizar las redes (algunas incluso antes, como las VLANs), logrando representar distintas topologías lógicas sobre la misma infraestructura física de red.

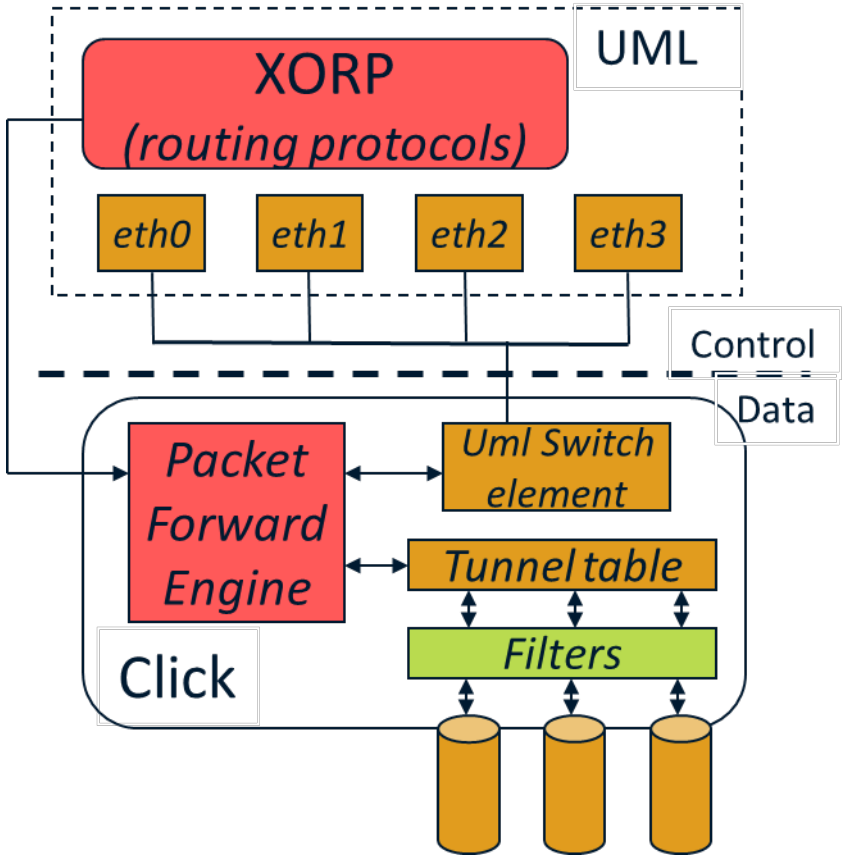


Historia de SDN

Switchlets (1998)



VINI (2006)



CABO (2007): separa infraestructura de servicios

Historia de SDN

FORCES (2003)

Protocolos para el control y reenvío de elementos

Requiere estándares y despliegue de nuevo HW

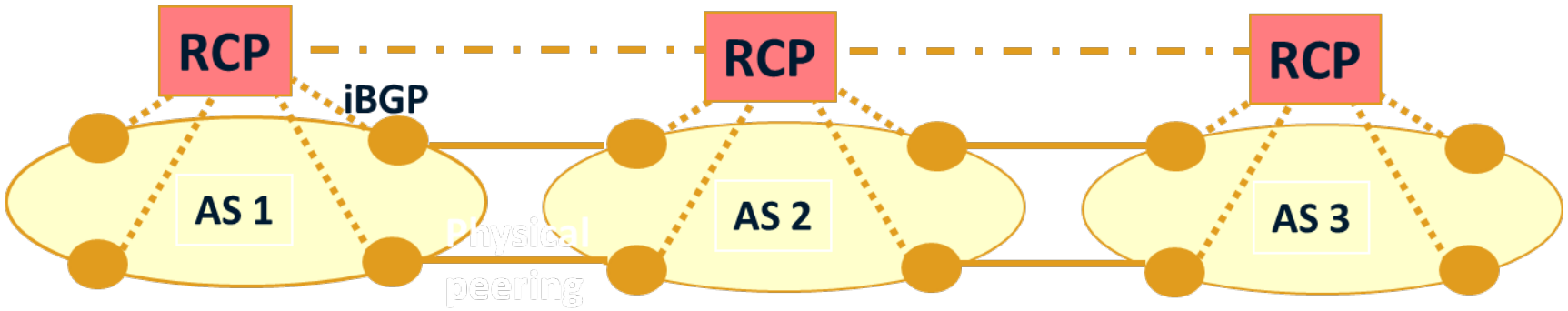
RCP (2004)

Emplea protocolos existentes como BGP, lo que limita mucho sus capacidades

Ethane (2007)

Un controlador de dominio rellena las tablas de reenvío de los dispositivos de red

OpenFlow (2008)



Oportunidades con SDN

Gracias a las ventajas comentadas, con SDN tendré oportunidades en:

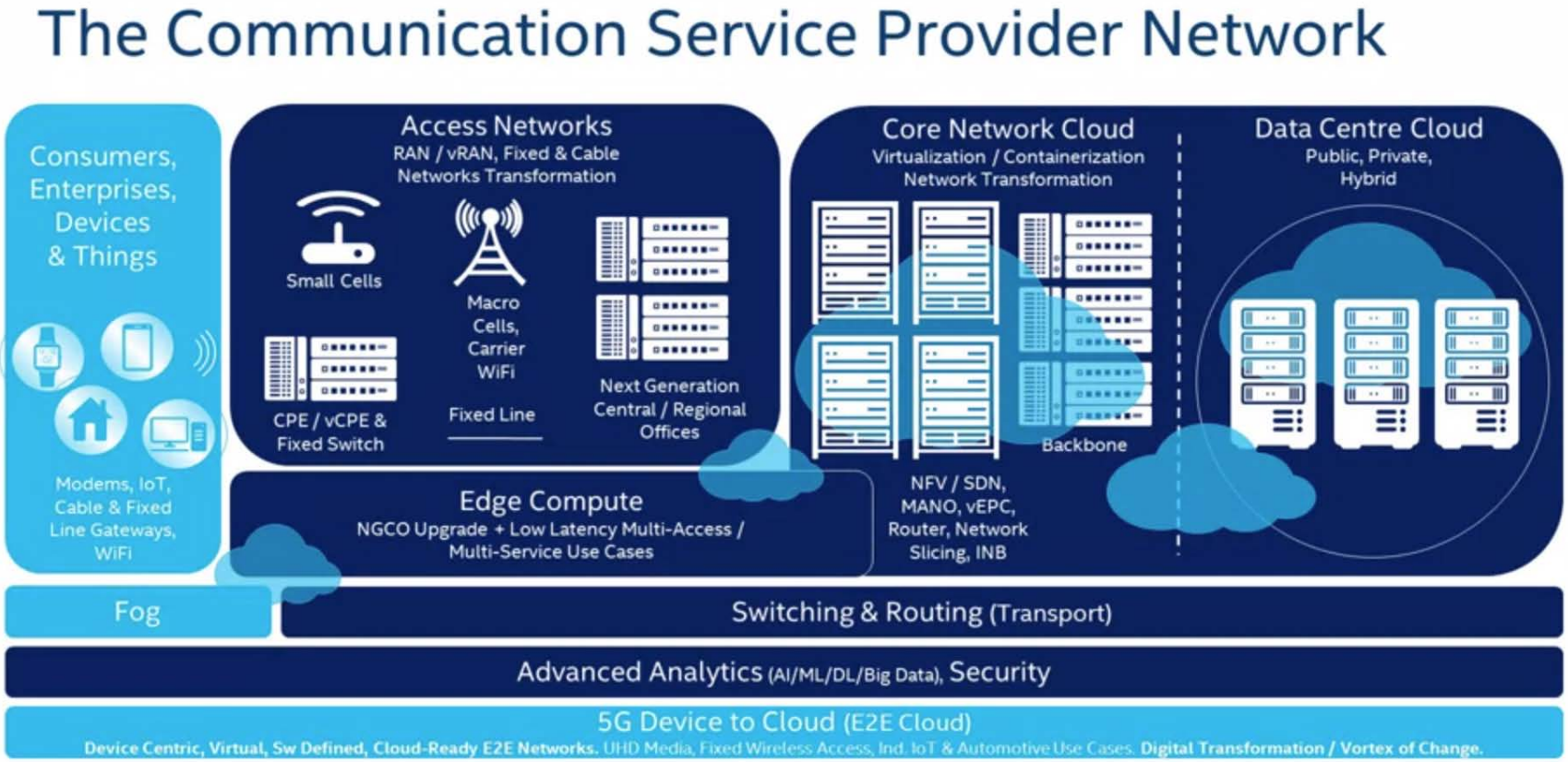
- Nuevos servicios de enrutamiento
  - Elección del Gateway
  - Selección de ruta para mantenimiento
  - Selección de ruta por cliente
- Data centers
  - Menor coste
  - Gestión más sencilla y flexible

Retos con SDN

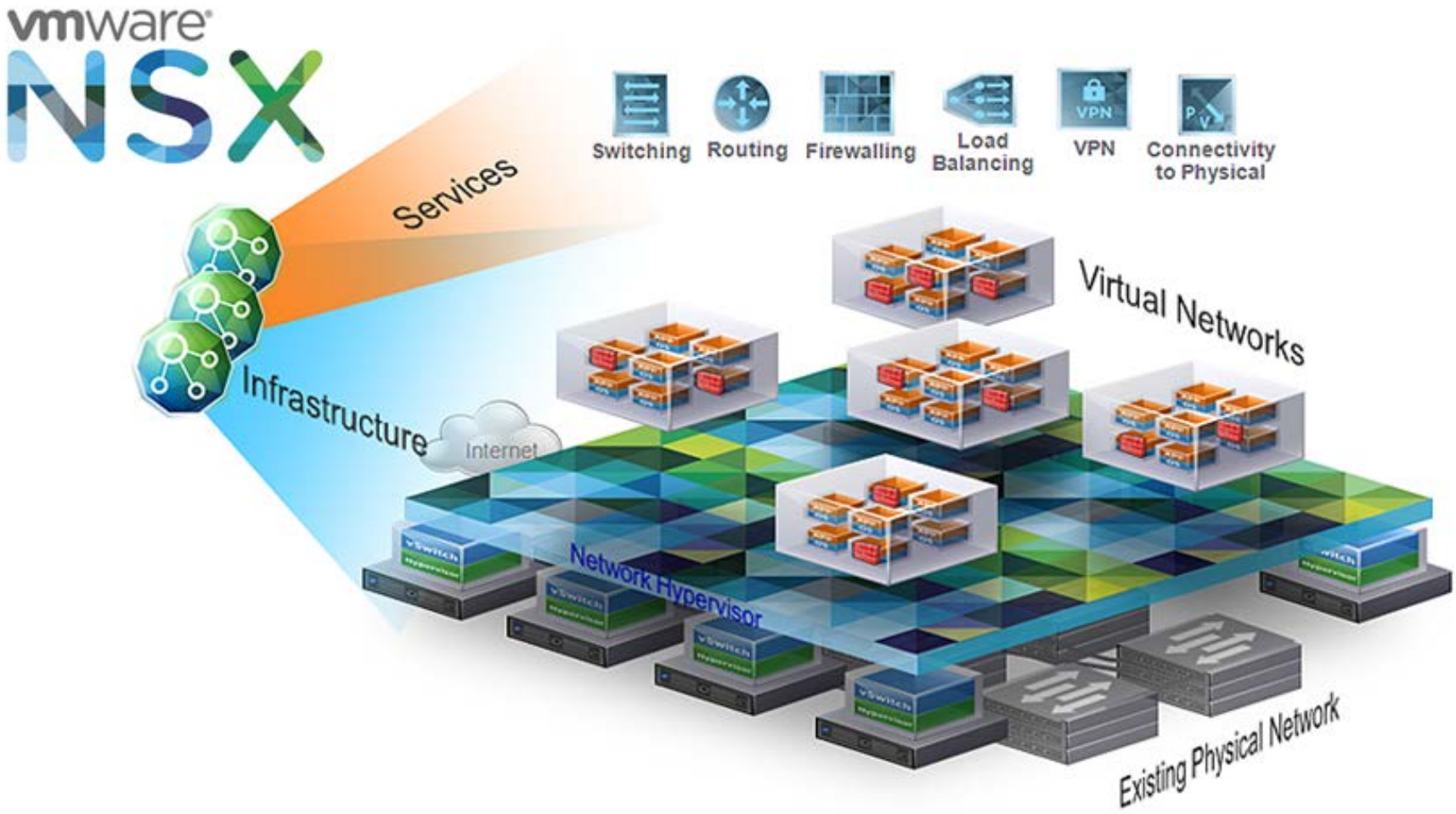
La centralización en SDN conlleva una serie de nuevos retos que afrontar, como:

- Escalabilidad
  - Eliminando redundancia
  - Acelerando búsquedas mediante índices en tablas
  - Particionando o agregando
- Fiabilidad
  - Redundancia
- Consistencia
  - Gestión correcta de réplicas y particiones

Interés de las Telcos en SDN

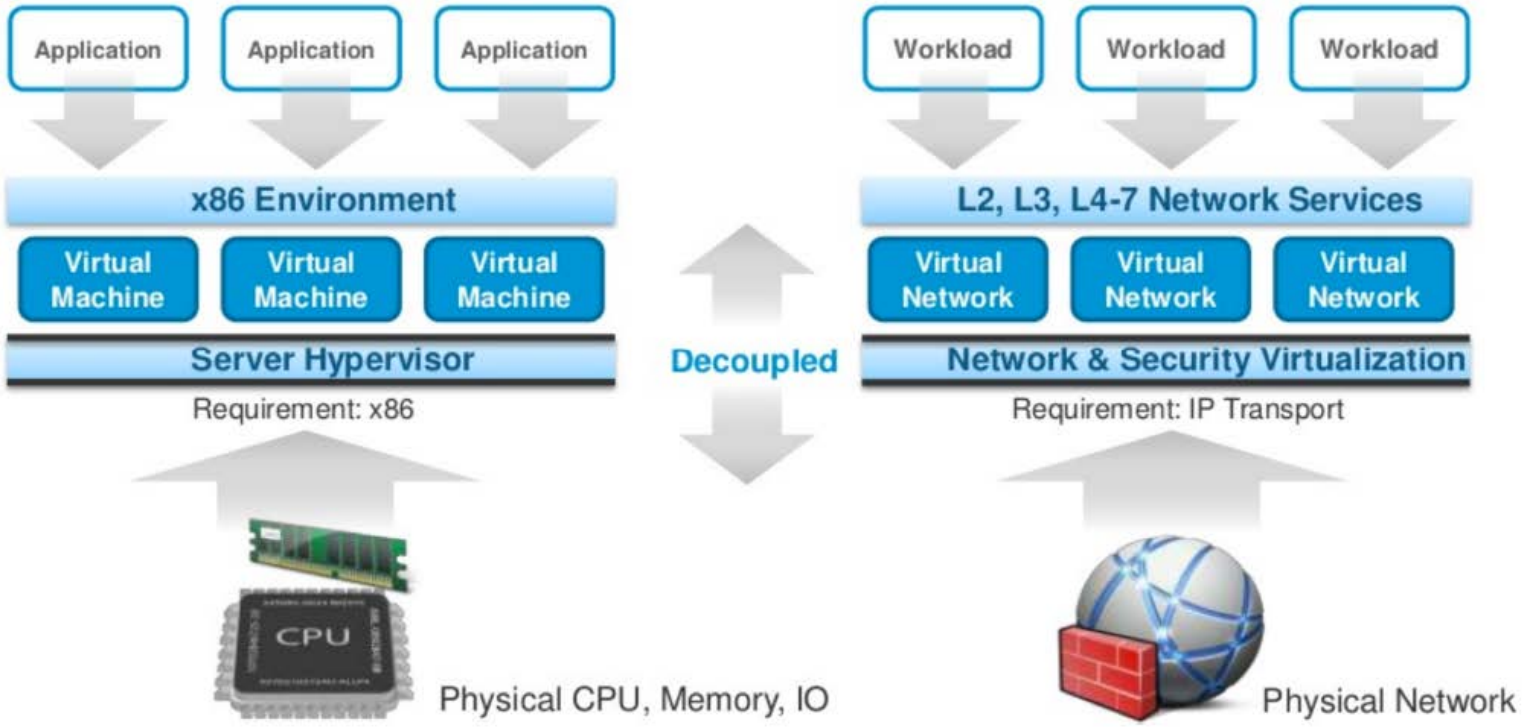


Virtualización de redes



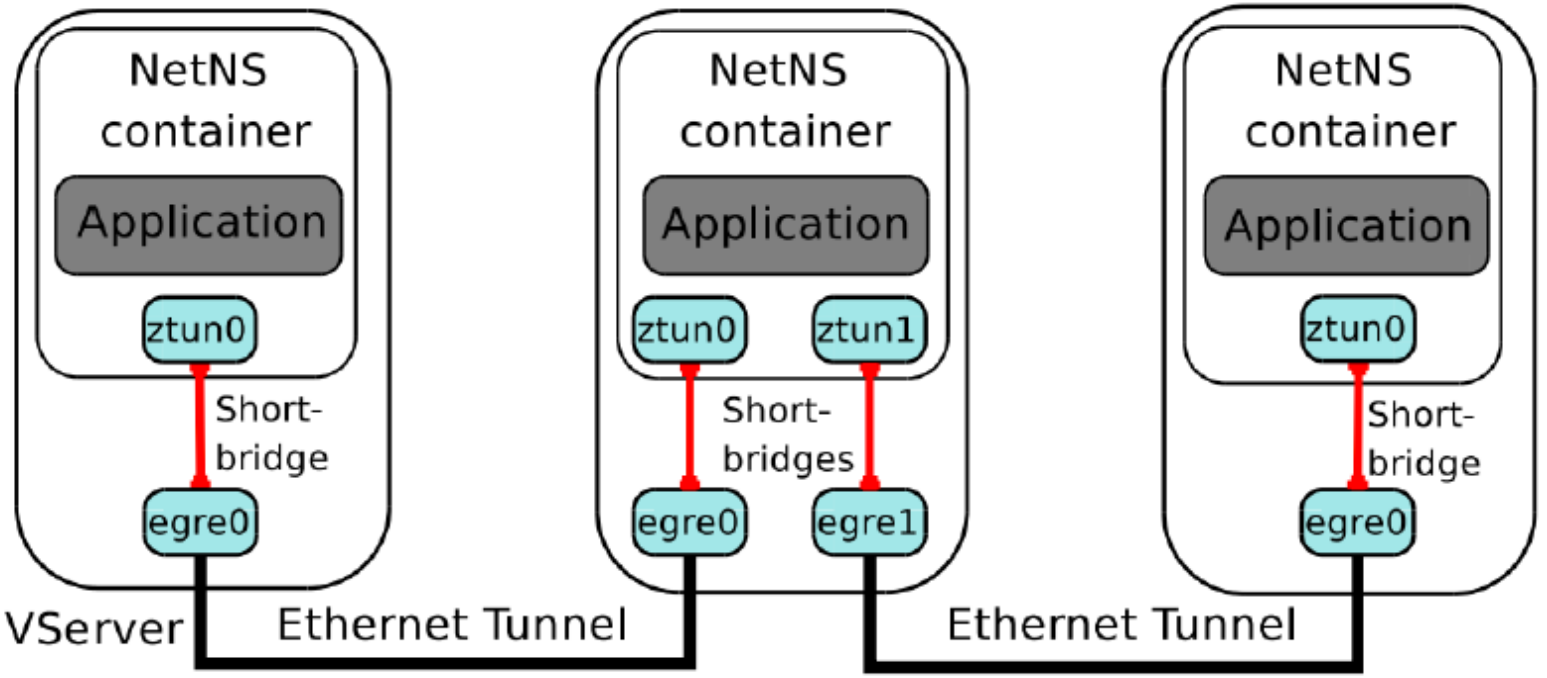
Virtualización de redes

- Abstraerse de la red física, soportando múltiples redes lógicas sobre la misma infraestructura física
- Empleando, por ejemplo, máquinas virtuales para los nodos y túneles para los enlaces



Virtualización de redes

- Abstraerse de la red física, soportando múltiples redes lógicas sobre la misma infraestructura física
- Empleando, por ejemplo, máquinas virtuales para los nodos y túneles para los enlaces



Virtualización de redes

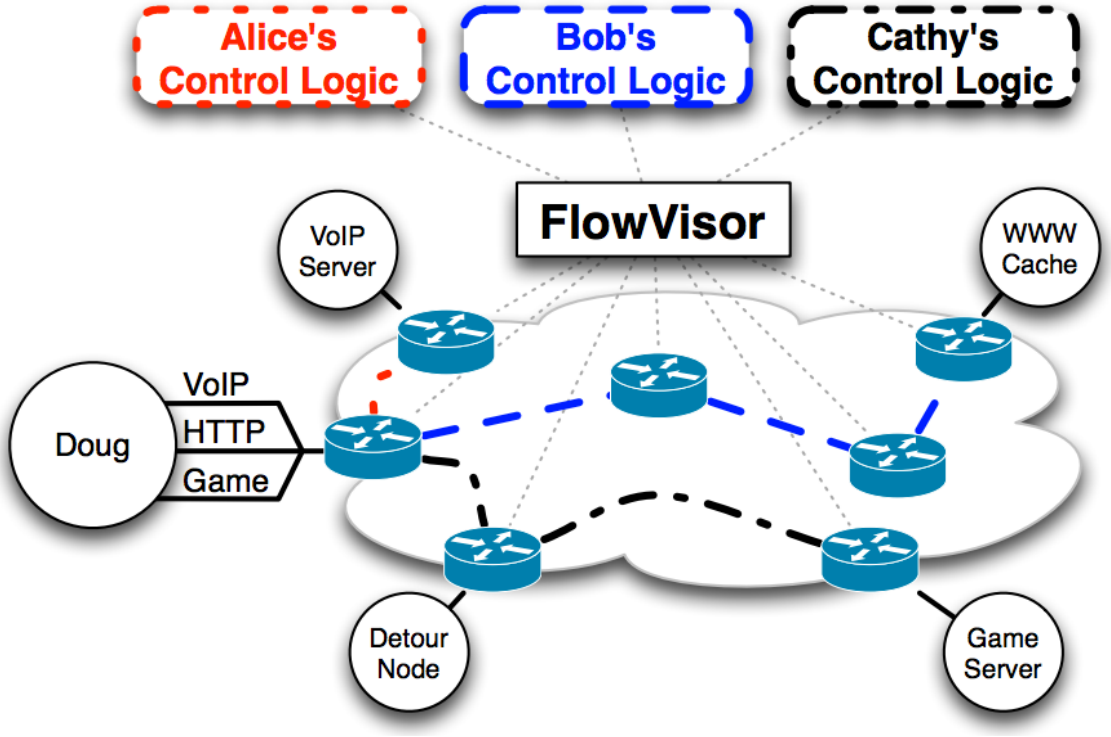
Los objetivos de la virtualización de redes son:

- Flexibilidad: en cuanto a topologías y arquitecturas para el routing y forwarding, y que la configuración sea independiente.
- Gestionable: separando las políticas de los mecanismos.
- Escalabilidad: maximizando el número de redes virtuales que coexisten.
- Seguridad y aislamiento: aislando tanto las redes lógicas como los recursos.
- Programabilidad: routers programables, etc.
- Heterogeneidad: soportando distintas tecnologías.

Virtualización de redes

La virtualización de redes permite:

- Realizar experimentos en redes sobre infraestructura real, en paralelo a la de producción

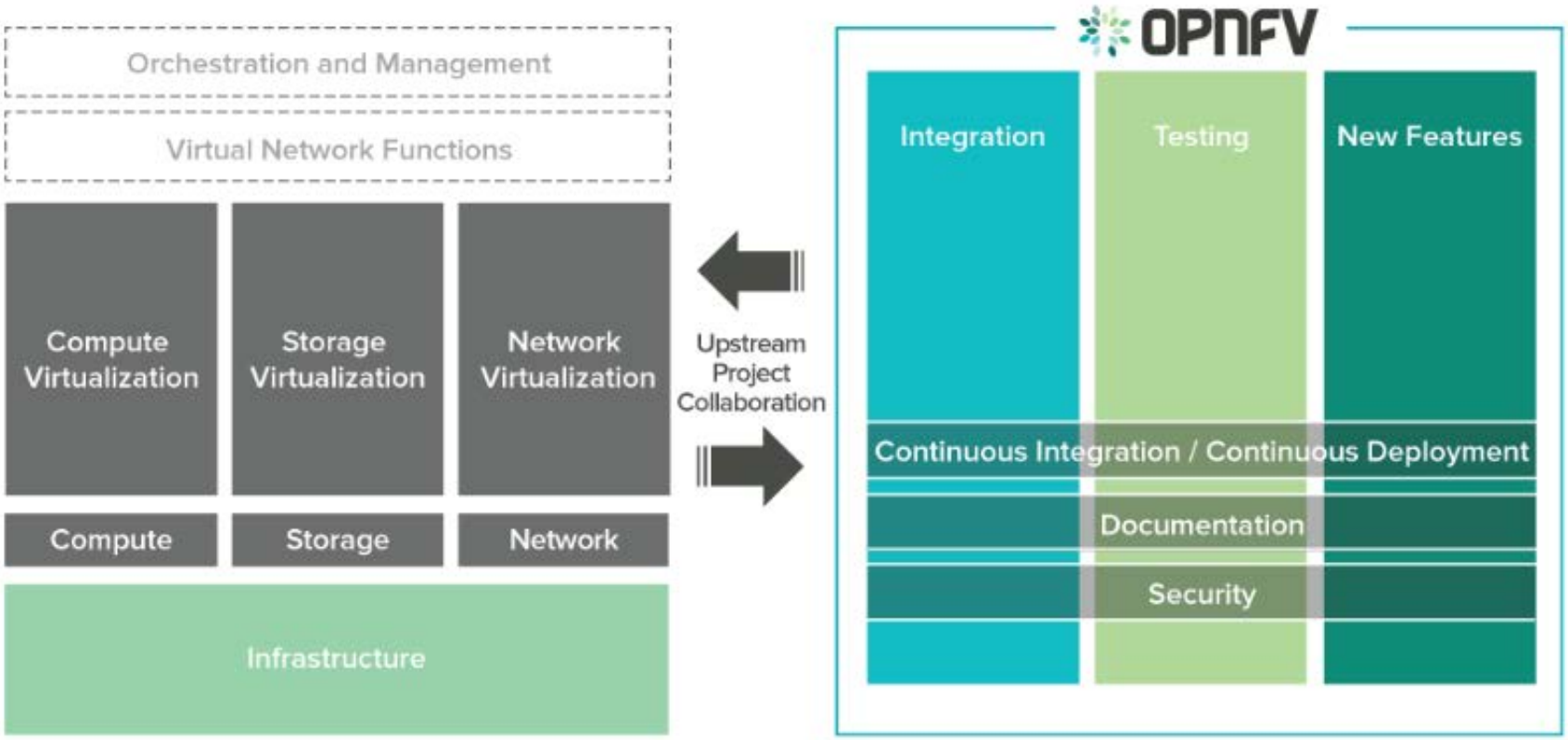


Virtualización de redes

La virtualización de redes permite:

- En data centers grandes con múltiples clientes, crear redes virtuales **aisladas** para cada cliente, sobre la misma infraestructura.
- Crear políticas de seguridad dinámicas, con un gestor centralizado de esas políticas de seguridad y pudiendo aplicarlas a cada red virtual.
- Escalar recursos de forma dinámica mediante la contratación de servicios cloud que extiendan mi red.
- Virtualizar funciones de red (firewall, DPI, balanceadores de carga,…) → NFV

Network Functions Virtualization (NFV)

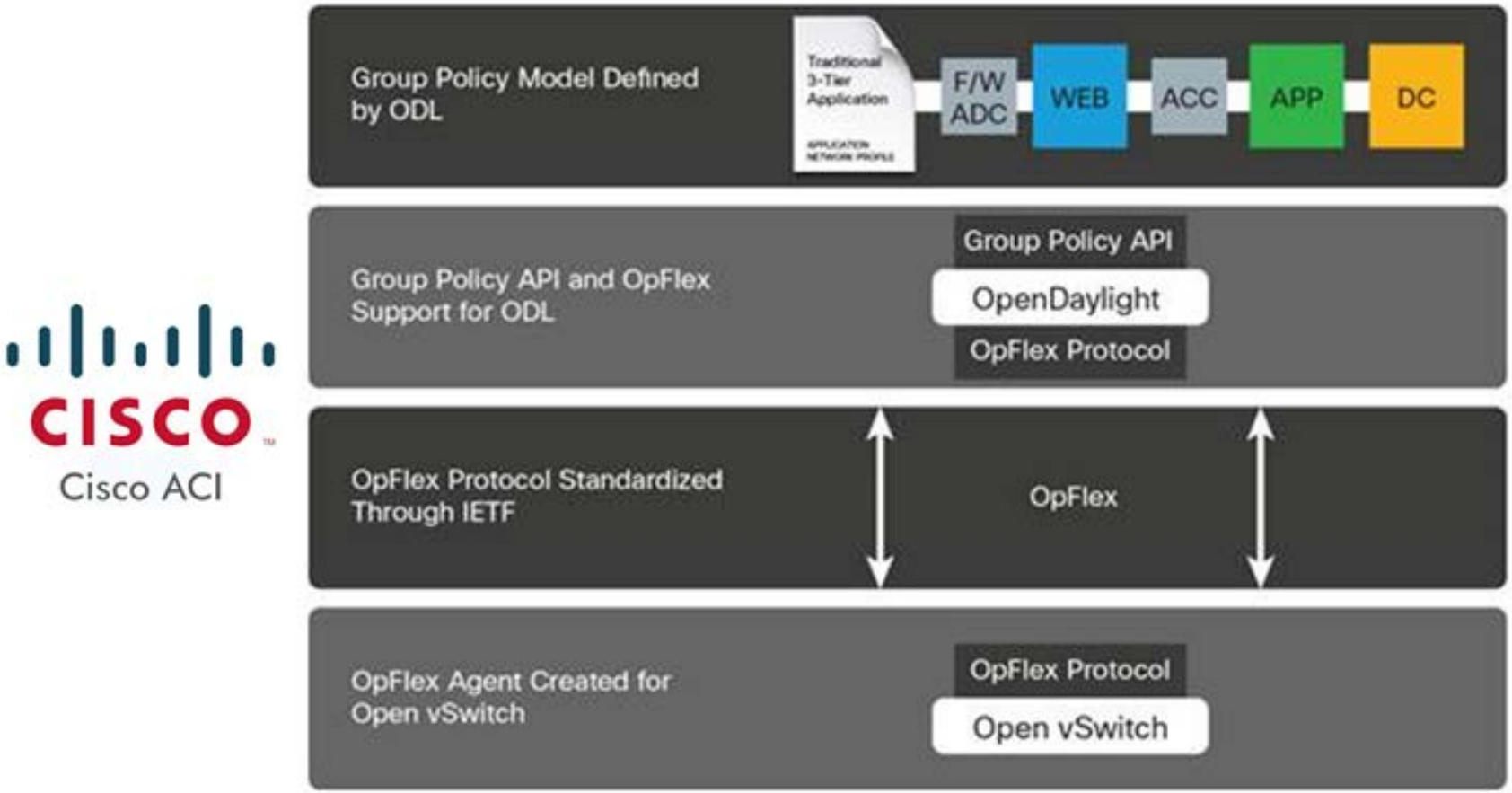


White Boxes

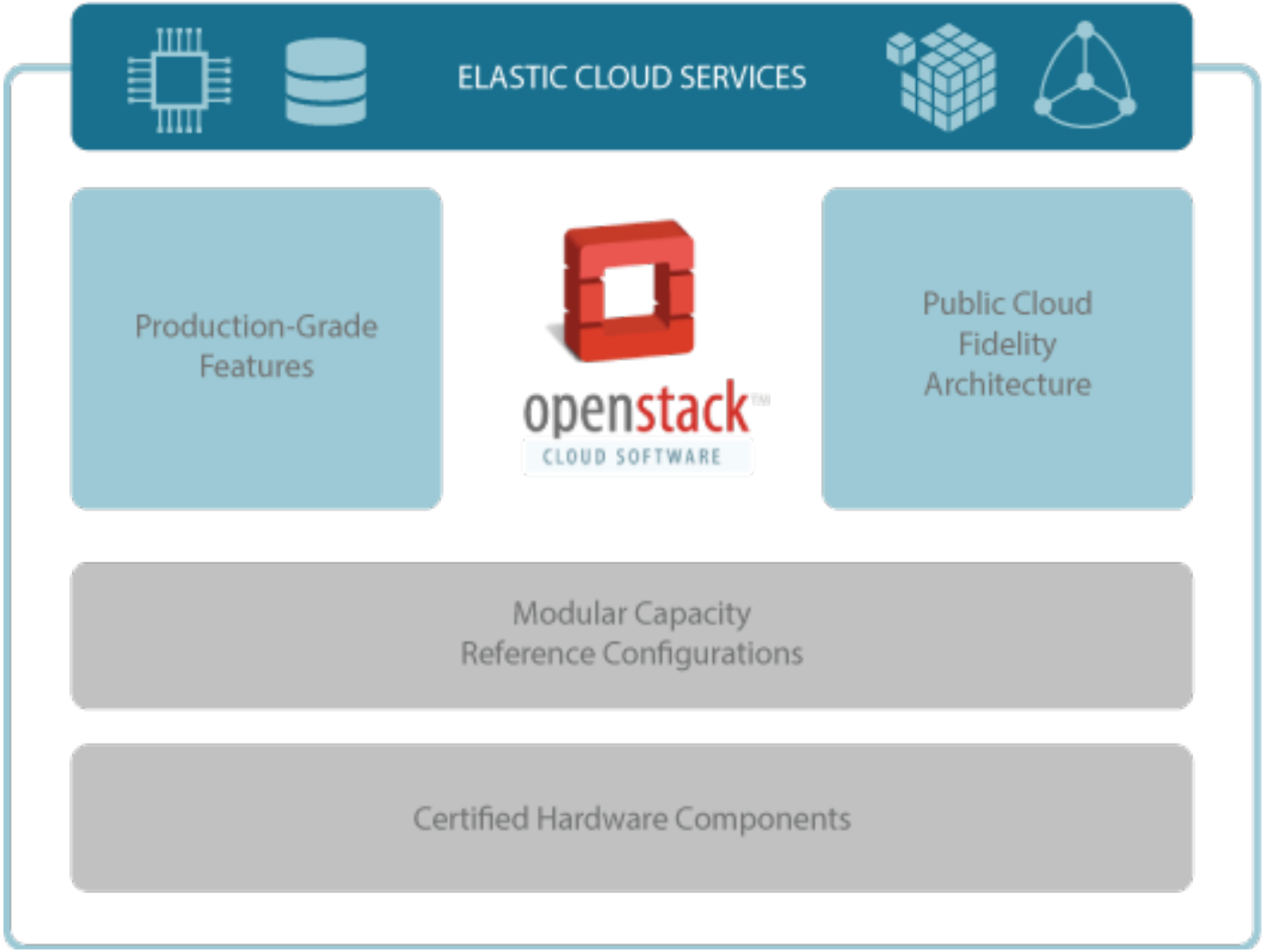
cumulus networks

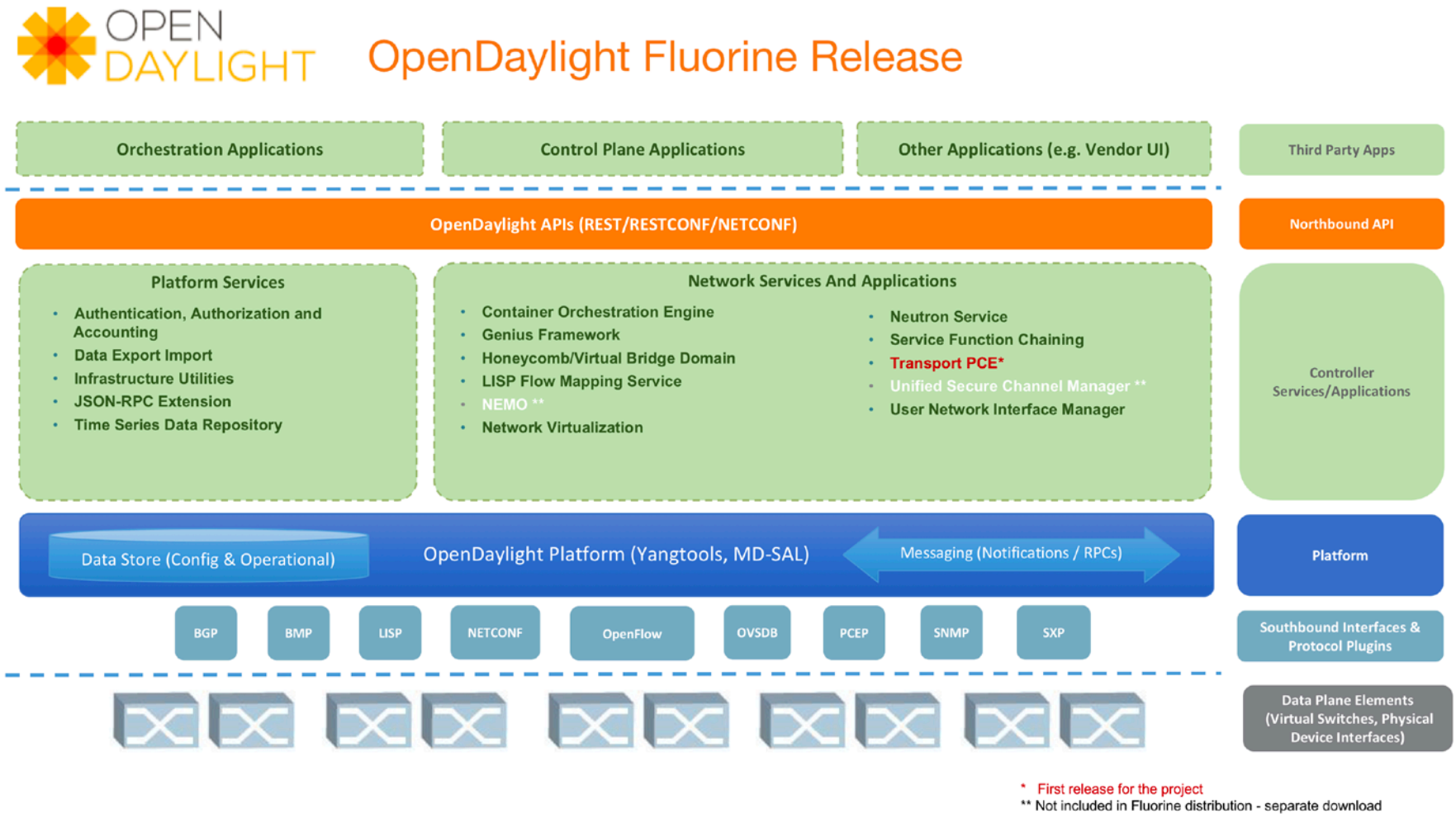
PICA8 WHITE BOX SDN

HW especializado (ASICs)



Cloud Automation




OPEN DAYLIGHT
OpenDaylight Fluorine Release
Orchestration Applications
Control Plane Applications
Other Applications (e.g. Vendor UI)
Third Party Apps
OpenDaylight APIs (REST/RESTCONF/NETCONF)
Northbound API
Platform Services
Authentication, Authorization and Accounting
Data Export Import
Infrastructure Utilities
JSON-RPC Extension
Time Series Data Repository
Network Services And Applications
Container Orchestration Engine
Genius Framework
Honeycomb/Virtual Bridge Domain
LISP Flow Mapping Service
NEMO **
Network Virtualization
Neutron Service
Service Function Chaining
Transport PCE*
Unified Secure Channel Manager **
User Network Interface Manager
Controller Services/Applications
Data Store (Config & Operational)
OpenDaylight Platform (Yangtools, MD-SAL)
Messaging (Notifications / RPCs)
Platform
BGP
BMP
LISP
NETCONF
OpenFlow
OVSDB
PCEP
SNMP
SXP
Southbound Interfaces & Protocol Plugins
Data Plane Elements (Virtual Switches, Physical Device Interfaces)
* First release for the project
** Not included in Fluorine distribution - separate download

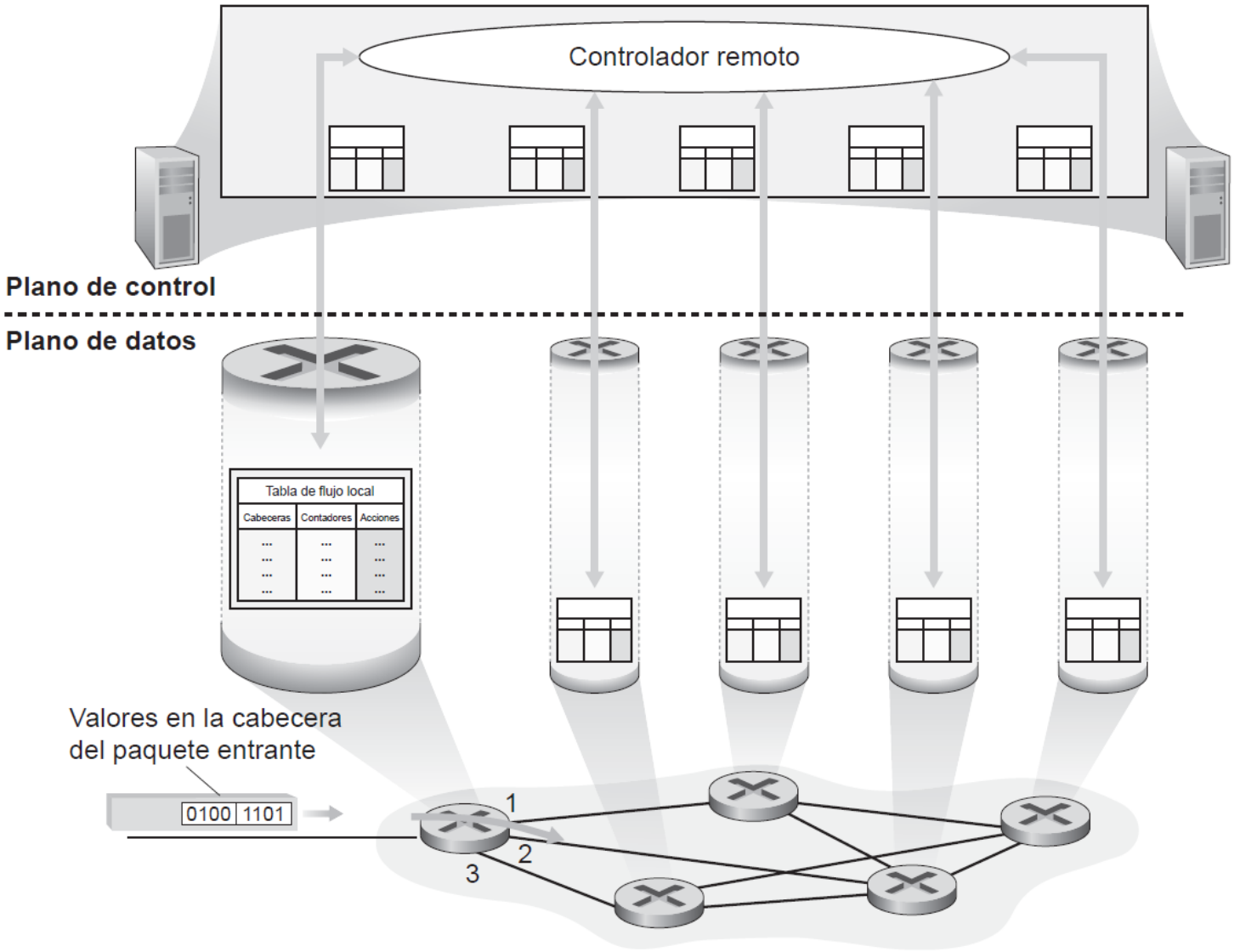
Planos de control y de datos

- Plano de control:
  - El cerebro de la red, que controla el comportamiento de la red
  - Puede funcionar separado de los dispositivos de red
  - Es la lógica que determina cómo se reenvía el tráfico
- Plano de datos:
  - Hardware programable controlado por el plano de control

Plano de datos (Reenvío generalizado)

- Hasta ahora el reenvío era realizado por los routers y estaba basado en la dirección destino del paquete
- Ahora hay muchos dispositivos de red (NAT, firewall, balanceadores de carga, switches…) que toman decisiones en capa 3
- El reenvío tiene 2 pasos:
  - Correspondencia (buscar la IP destino)
  - Acción (enviar el paquete a la salida)
- Ahora la correspondencia puede buscar en más campos
- Ahora la acción puede consistir en enviar el paquete a una o varias salidas, reescribir campos, bloquear paquetes, procesar el paquete y decidir acciones (DPI)…

Reenvío generalizado



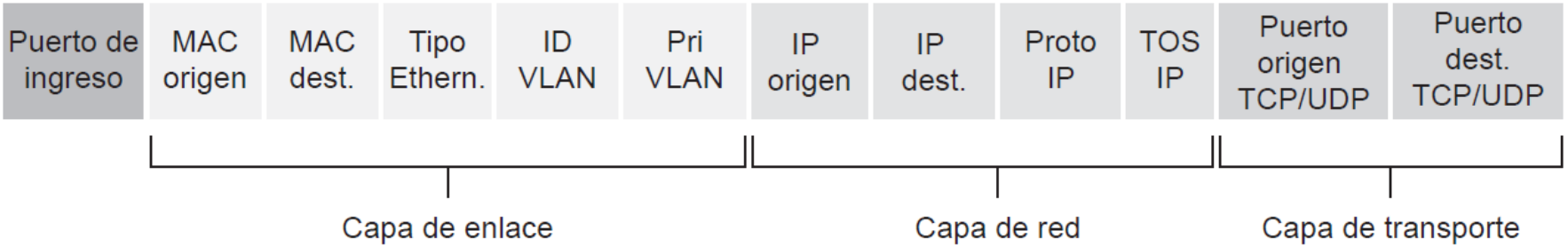
Tablas de flujo

- Cada switch tiene una tabla de flujo local
- Correspondencia+Acción (Match+Action)
- Calculada, instalada y actualizada por un controlador remoto centralizado

Tabla de flujo local		
Cabeceras	Contadores	Acciones
...	...	...
...	...	...
...	...	...
...	...	...

OpenFlow 1.0:

- Cabeceras (donde busco el match)



- Contadores (#paq, #bytes, tiempo desde actualización…)
- Prioridad
- Acciones:
  - Reenviar a un puerto físico de salida concreto, difundir a algunos o todos los puertos, encapsular y enviar a su controlado remoto (que puede instalar nuevas entradas de la tabla y devolver el paquete al dispositivo para que lo reenvíe de acuerdo con el nuevo conjunto de reglas)
  - Eliminar el paquete
  - Modificar campos del paquete

OpenFlow abstraction

Universidad Rey Juan Carlos

match+action: unifica distintos tipos de dispositivos

- Router
  - *match:* prefijo IP destino más largo
  - *action:* reenvía a un enlace
- Switch
  - *match:* direccion MAC destino
  - *action:* reenvío o inundación
- Firewall
  - *match*: direcciones IP y números de puerto TCP/UDP
  - *action:* permitir o denegar
- NAT
  - *match:* dirección IP y puerto
  - *action:* reescribir dirección y puerto

Ejemplos de OpenFlow

Reenvío basado en destino

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

Los datagramas IP con destino la direccion IP 51.6.0.8 se deben reenviar al puerto de salida 6 del router

Firewall

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

No reenviar (bloquea) ningún datagrama destinado al Puerto TCP 22

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

No reenviar (bloquea) ningún datagrama enviado por el host 128.119.1.1

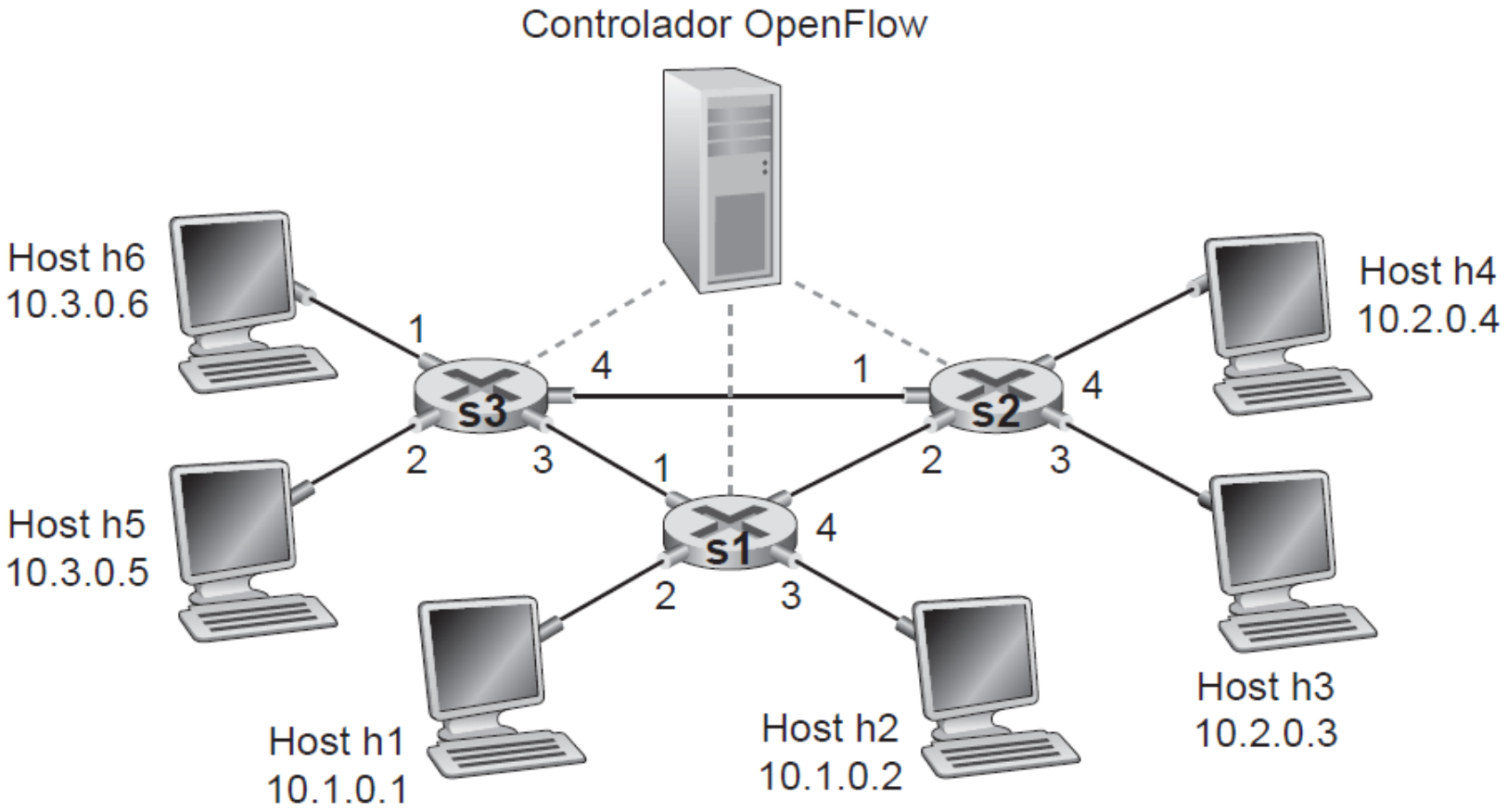
Ejemplos de OpenFlow

Reenvío basado en destino (switch capa 2)

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
*	22:A7:23: 11:E1:02	*	*	*	*	*	*	*	*	port3

Las tramas de capa 2 desde la dirección MAC 22:A7:23:11:E1:02 se deben reenviar al puerto de salida 6

Reenvío basado en destino y origen



Ejemplos de OpenFlow

Reenvío basado en destino y origen

Los paquetes de h5 o h6 destinados a h3 o h4 deben reenviarse de s3 a s1 y luego de s1 a s2 (evitando el enlace entre s3 y s2)

Tabla de flujo de s1 (Ejemplo 1)

Correspondencia	Acción
Puerto de ingreso = 1 ; Origen IP = 10.3.*.* ; Destino IP = 10.2.*.*	Reenvío(4)

Tabla de flujo de s3 (Ejemplo 1)

Correspondencia	Acción
Origen IP = 10.3.*.* ; Destino IP = 10.2.*.*	Reenvío(3)

Tabla de flujo de s2 (Ejemplo 1)

Correspondencia	Acción
Puerto de ingreso = 2 ; Destino IP = 10.2.0.3	Reenvío(3)
Puerto de ingreso = 2 ; Destino IP = 10.2.0.4	Reenvío(4)

Ejemplos de OpenFlow

Balanceo de carga

Los datagramas de h3 destinados a 10.1.*.* deben reenviarse a través del enlace directo entre s2 y s1, mientras que los datagramas de h4 destinados a 10.1.*.* deben reenviarse a través del enlace entre s2 y s3 (y luego de s3 a s1)

Tabla de flujo de s2 (Ejemplo 2)

Correspondencia	Acción
Puerto de ingreso = 3; Destino IP = 10.1.*.*	Reenvío(2)
Puerto de ingreso = 4; Destino IP = 10.1.*.*	Reenvío(1)

¿Tablas s1 y s3?

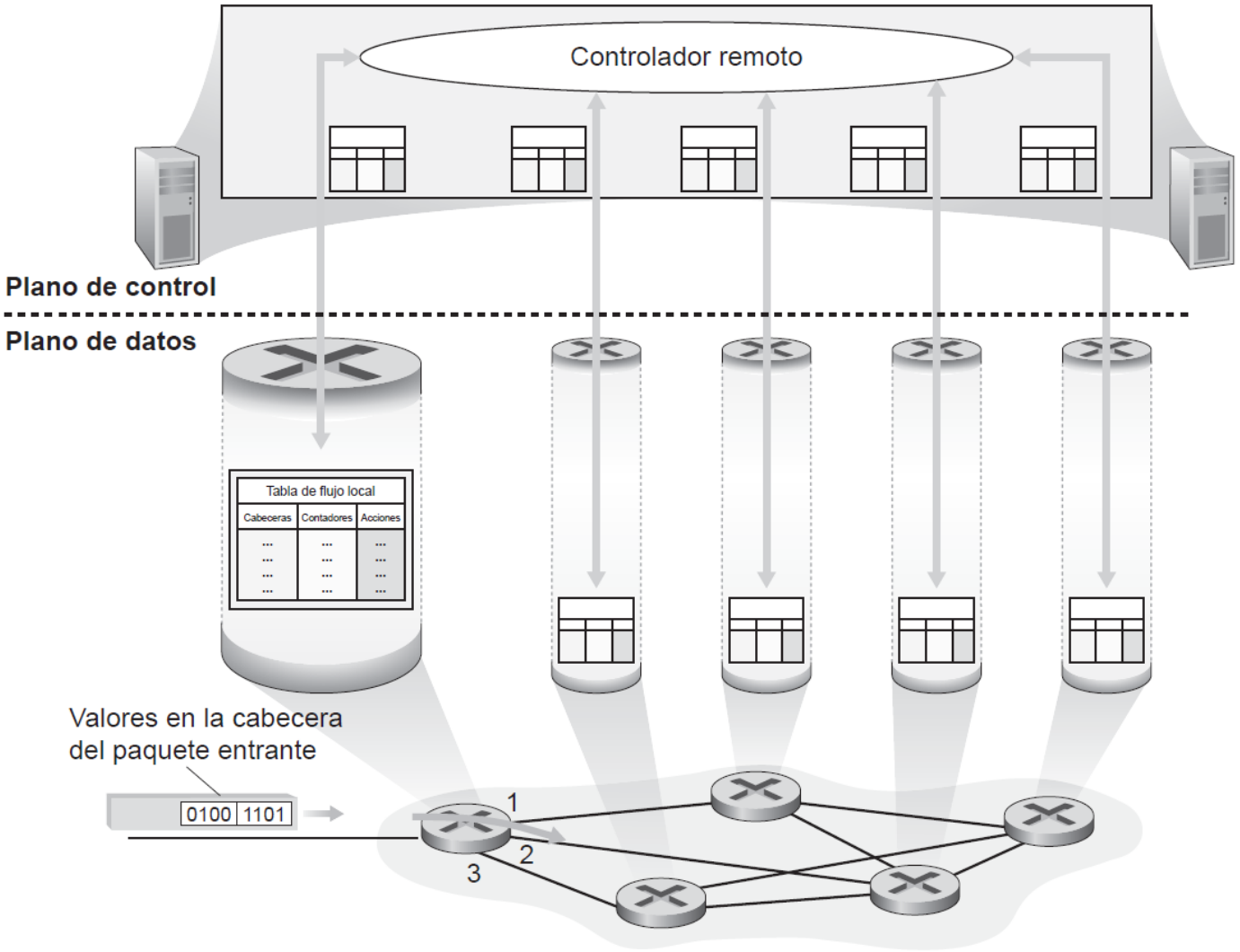
Firewall

s2 solo quiere recibir (a través de cualquiera de sus interfaces) el tráfico enviado por los hosts conectados a s3

Tabla de flujo de s2 (Ejemplo 3)

Correspondencia	Acción
Origen IP = 10.3.*.* Destino IP = 10.2.0.3	Reenvío(3)
Origen IP = 10.3.*.* Destino IP = 10.2.0.4	Reenvío(4)

Plano de control

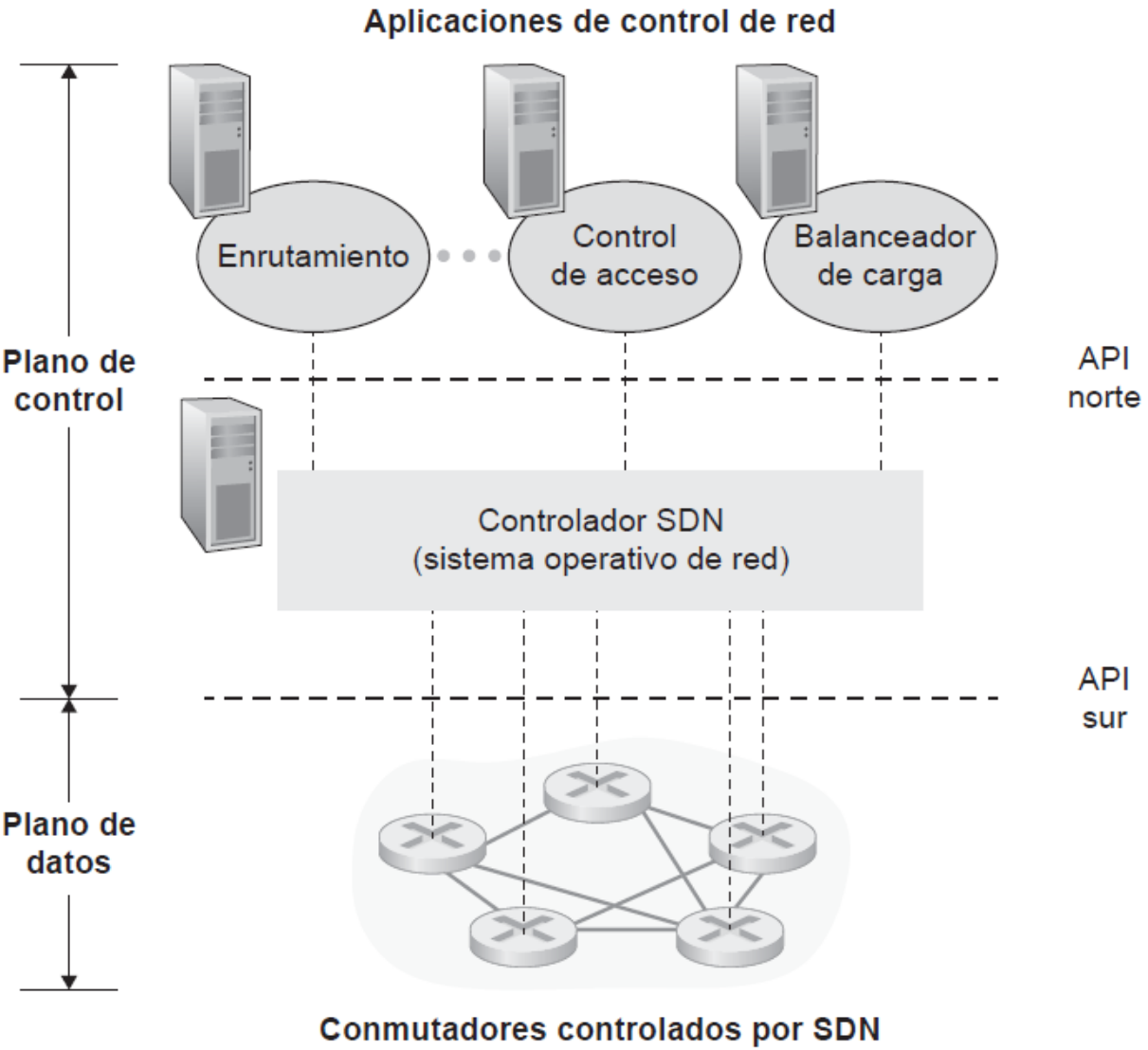


Plano de control

Recordamos algunas de las características de la arquitectura SDN:

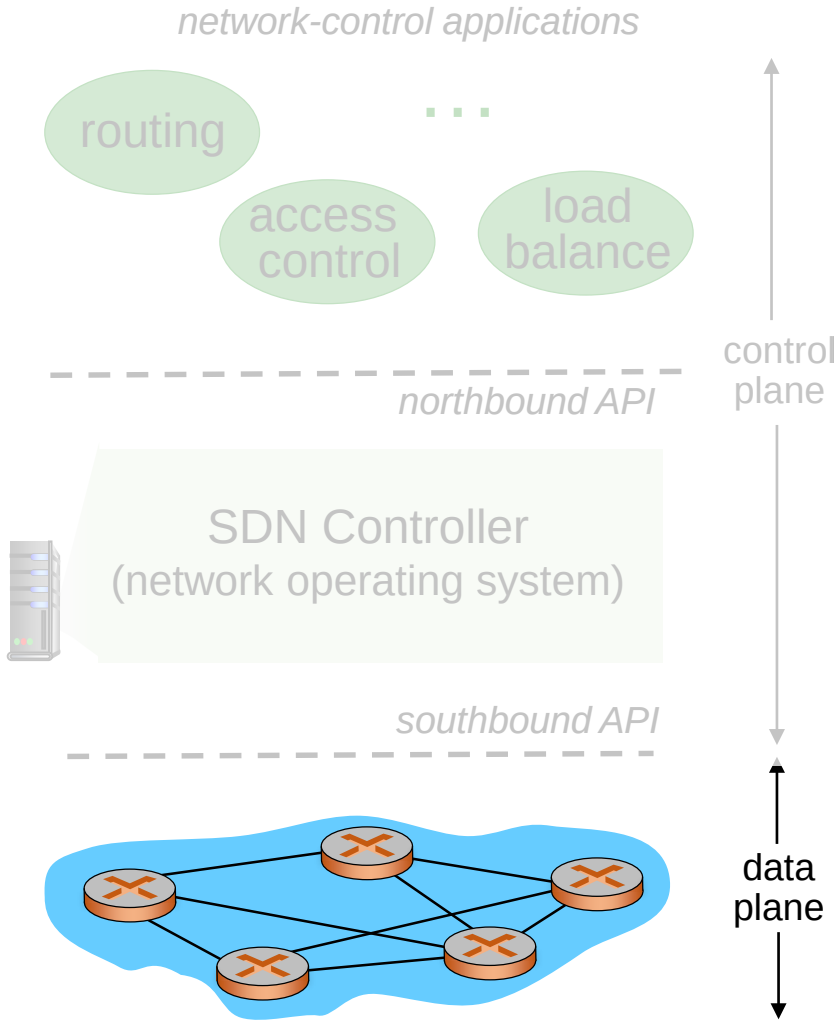
- Reenvío basado en el flujo
- Separación del plano de datos y el plano de control
- Funciones de control de la red
- Una red programable

Plano de control



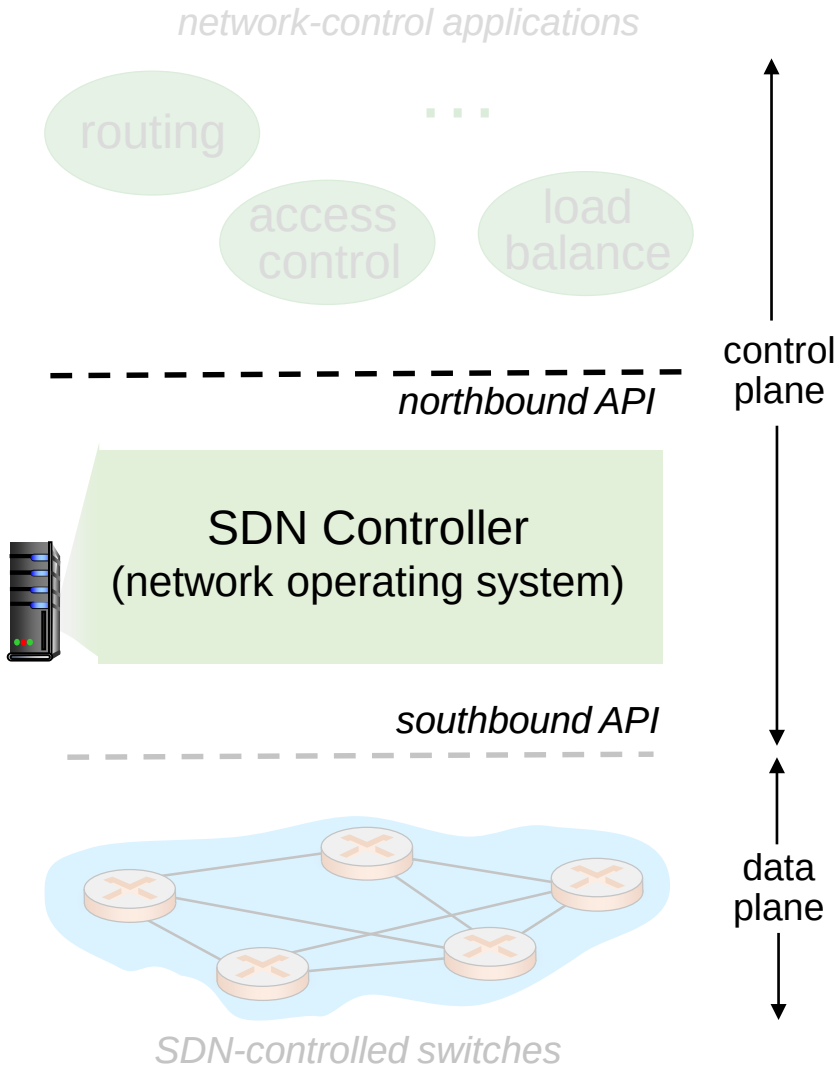
Switches del plano de datos

- Son rápidos y simples, e implementan el reenvío generalizado a nivel HW
- Cada switch tiene su tabla de flujo calculada e instalada por el controlador
- Dispone de API para controlar el switch mediante la table (por ejemplo, OpenFlow)
- Protocolo de comunicación con el controlador (por ejemplo, OpenFlow)



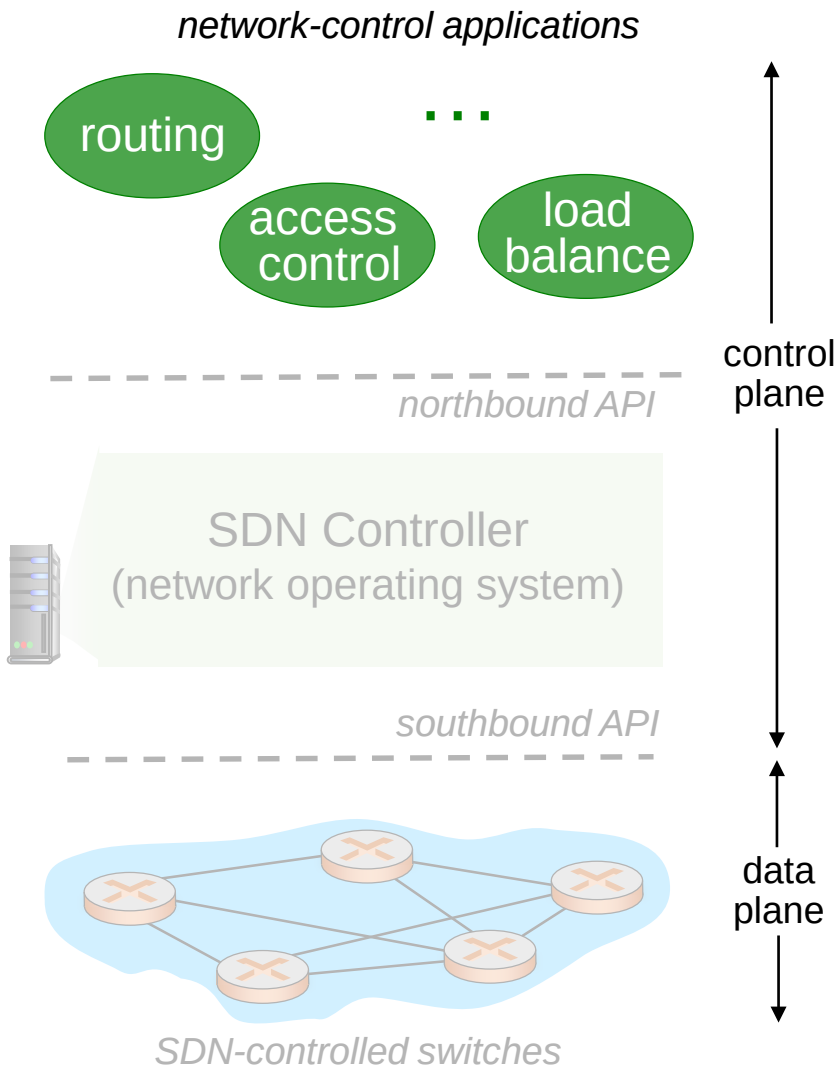
Controlador SDN (Network OS)

- Mantener la información de estado de la red
- Interaccionar con las aplicaciones de control de red de “arriba” mediante la API NorthBound
- Interaccionar con los switches de “abajo” mediante la API SouthBound
- Se implementa como sistema distribuido para mejorar el rendimiento, la escalabilidad, la tolerancia a fallos y la robustez de la solución

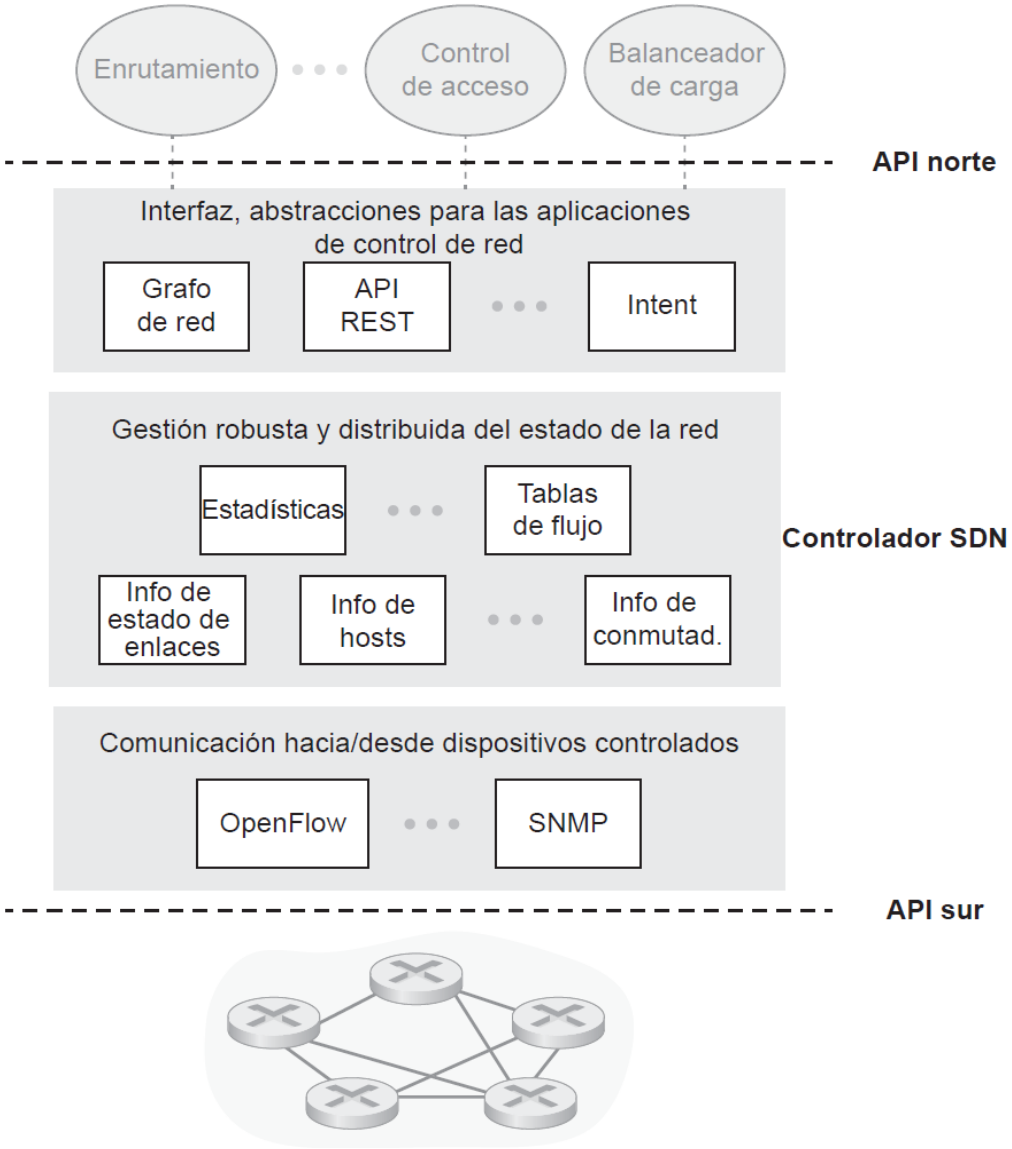


Aplicaciones de control de red

- Son el “cerebro” del control: implementan las funciones de control usando servicios de bajo nivel. La API la proporciona el controlador SDN.
- *desacoplado:* puede ser proporcionado por un tercero, distinto del proveedor del routing o del controlador SDN

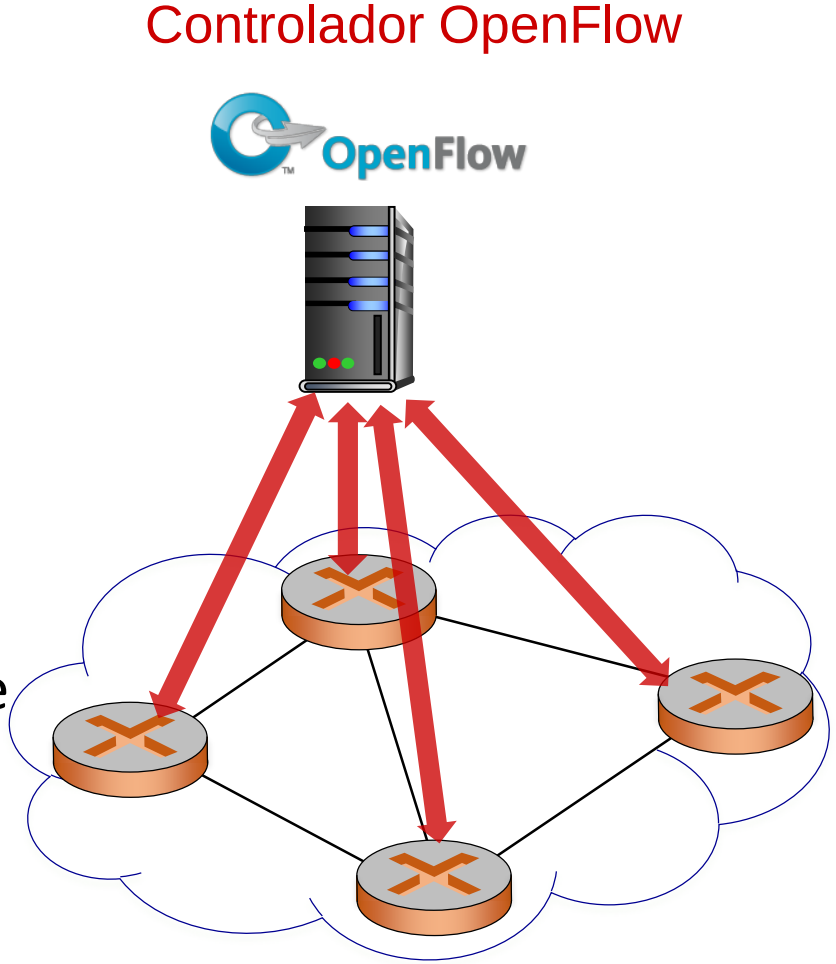


Componentes de un controlador SDN



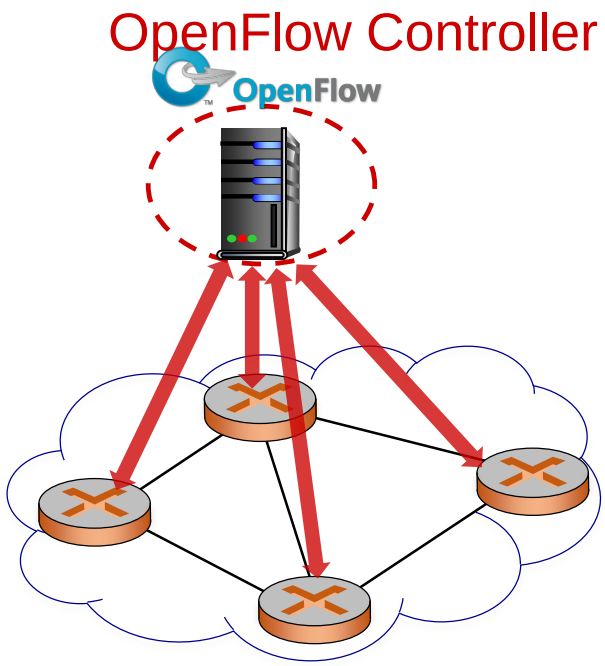
Protocolo OpenFlow

- Opera entre el controlador y el switch
- Emplea TCP (puerto 6653) para intercambiar mensajes (el cifrado es opcional)
- Tres clases de mensajes OpenFlow:
  - Del controlador al switch
  - Asíncronos (del switch al controlador para notificar de una llegada de un paquete, de un cambio de estado o error)
  - Simétricos (varios como el Echo, Hello, etc.)



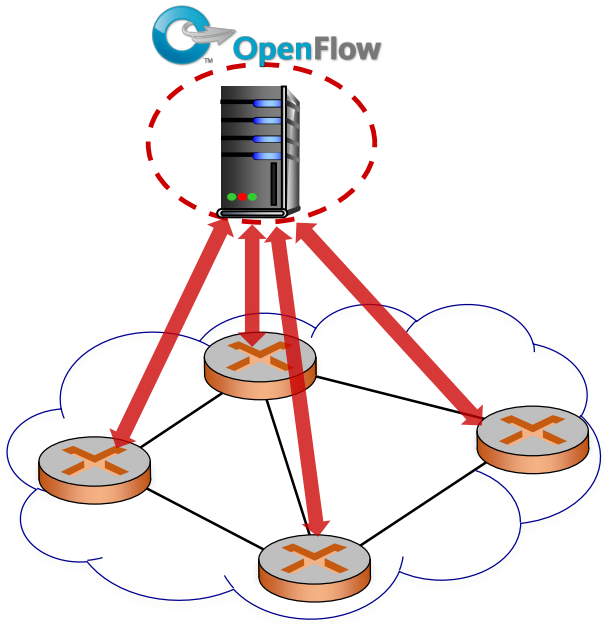
Mensajes OpenFlow del controlador al switch

- *features:* controlador solicita las características del switch y este responde
- *configure:* controlador solicita o fija los parámetros de configuración del switch
- *modify-state:* añade, borra o modifica entradas en las tablas de flujo de OpenFlow
- *packet-out:* controlador envía este paquete por un puerto específico del switch



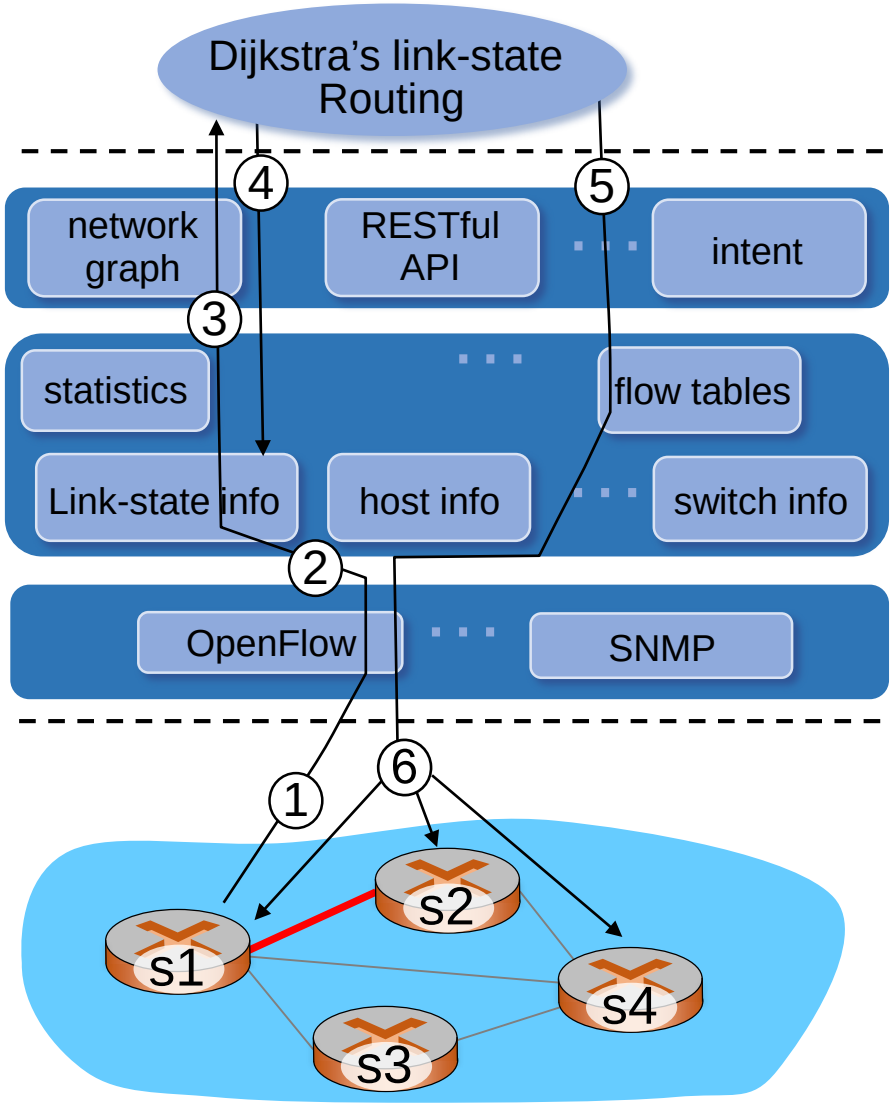
Mensajes OpenFlow del switch al controlador

- *packet-in:* pasa el paquete y su control al controlador
- *flow-removed:* informa al controlador de un borrado de una entrada de la tabla del switch
- *port-status:* informa al controlador de un cambio en un puerto



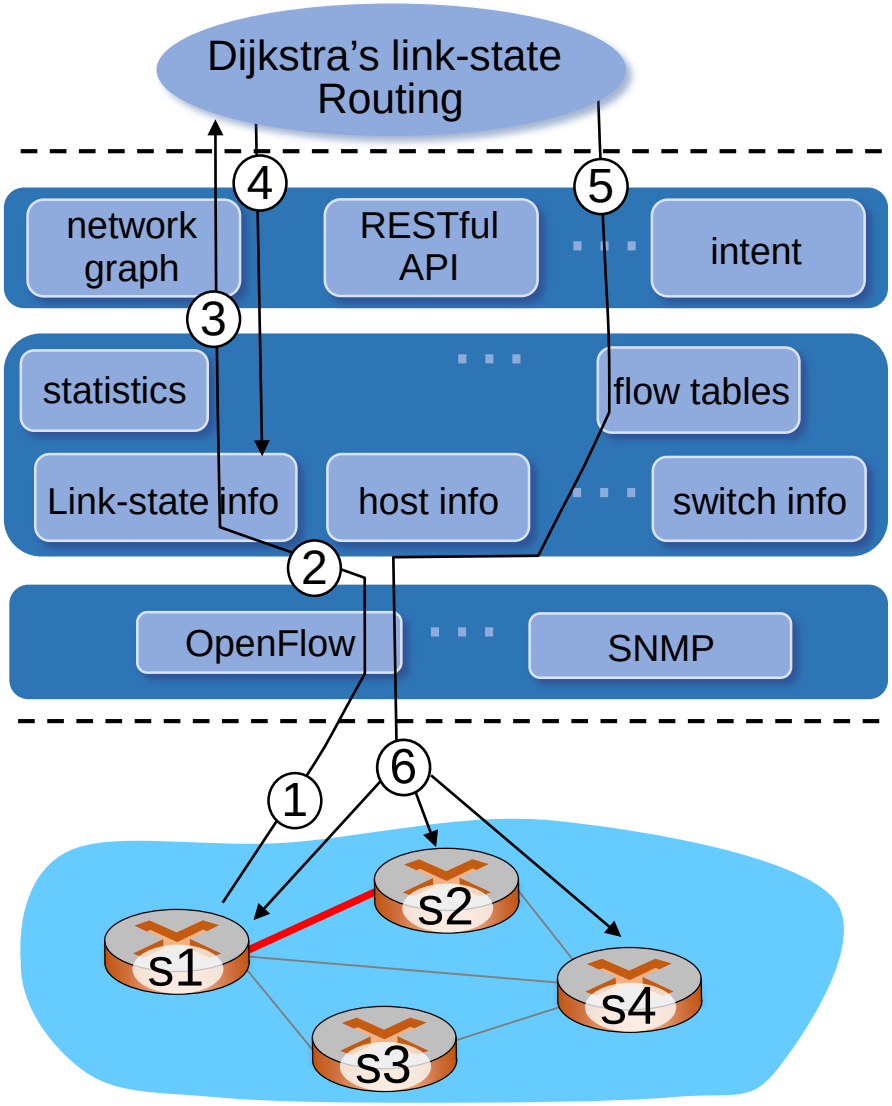
Afortunadamente, los operadores de red no "programan" los switches creando y enviando mensajes OpenFlow directamente. En su lugar, emplean una abstracción de alto nivel en el controlador

Ejemplo de interacción entre planos



1. S1, detecta el fallo en el enlace con S2 y envía el mensaje de port-status al controlador
2. El controlador recibe el mensaje y actualiza el estado del enlace
3. El algoritmo de enrutamiento Dijkstra se ha registrado previamente para ser avisado cuando haya un cambio en el estado de algún enlace. Así que es llamado
4. El algorimto de enrutamiento Dijkstra accede a la información del grafo de red, a la del estado de los enlaces en el controlador y calcula nuevas rutas

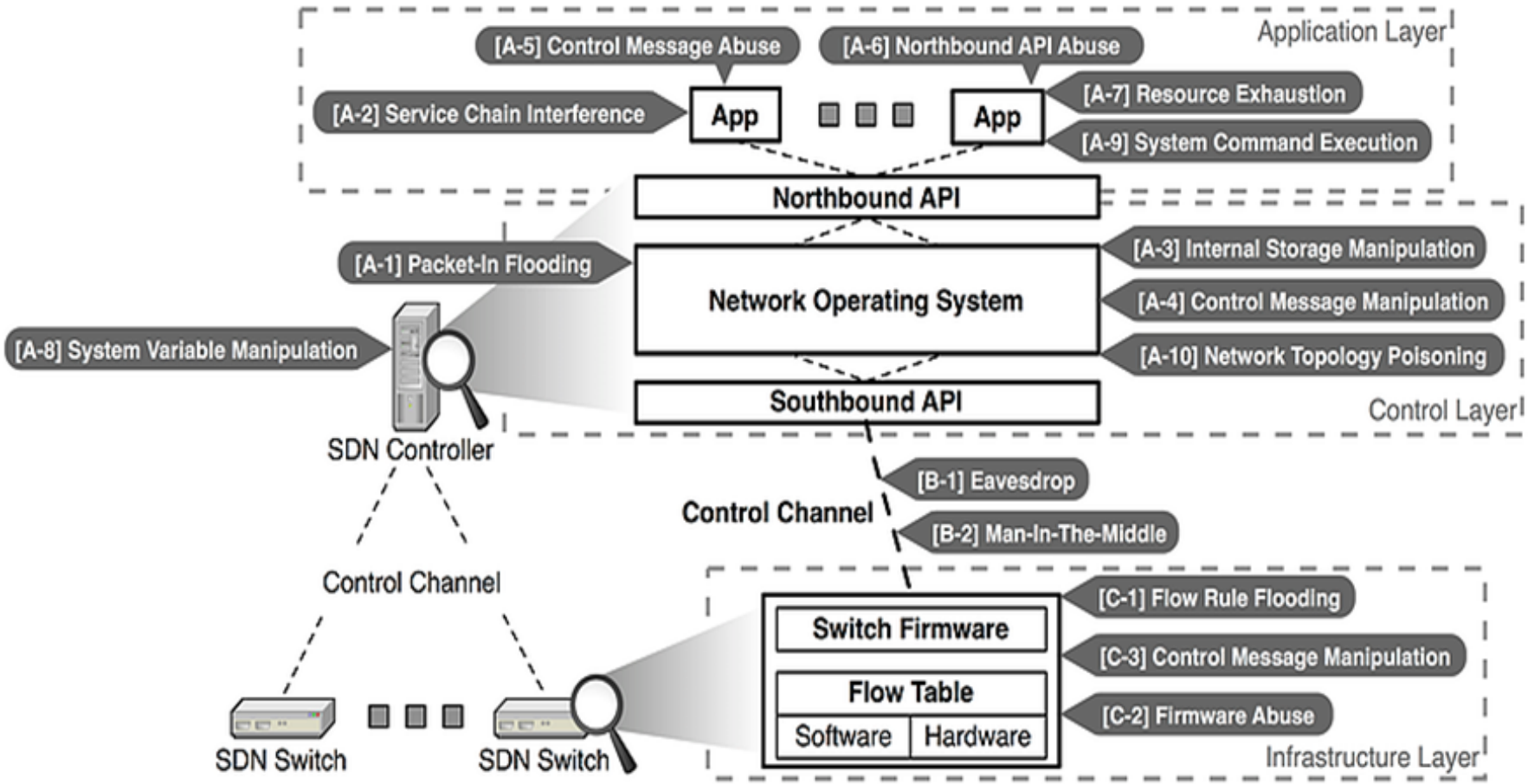
Ejemplo de interacción entre planos



⑤ La aplicación de enrutamiento por estado de enlaces interacciona con el componente encargado de calcular la tabla de flujo, quien realiza los cálculos pertinentes

⑥ El controlador emplea OpenFlow para instalar las nuevas tablas en los switches que hay que actualizar

Ataques y vulnerabilidades en SDN

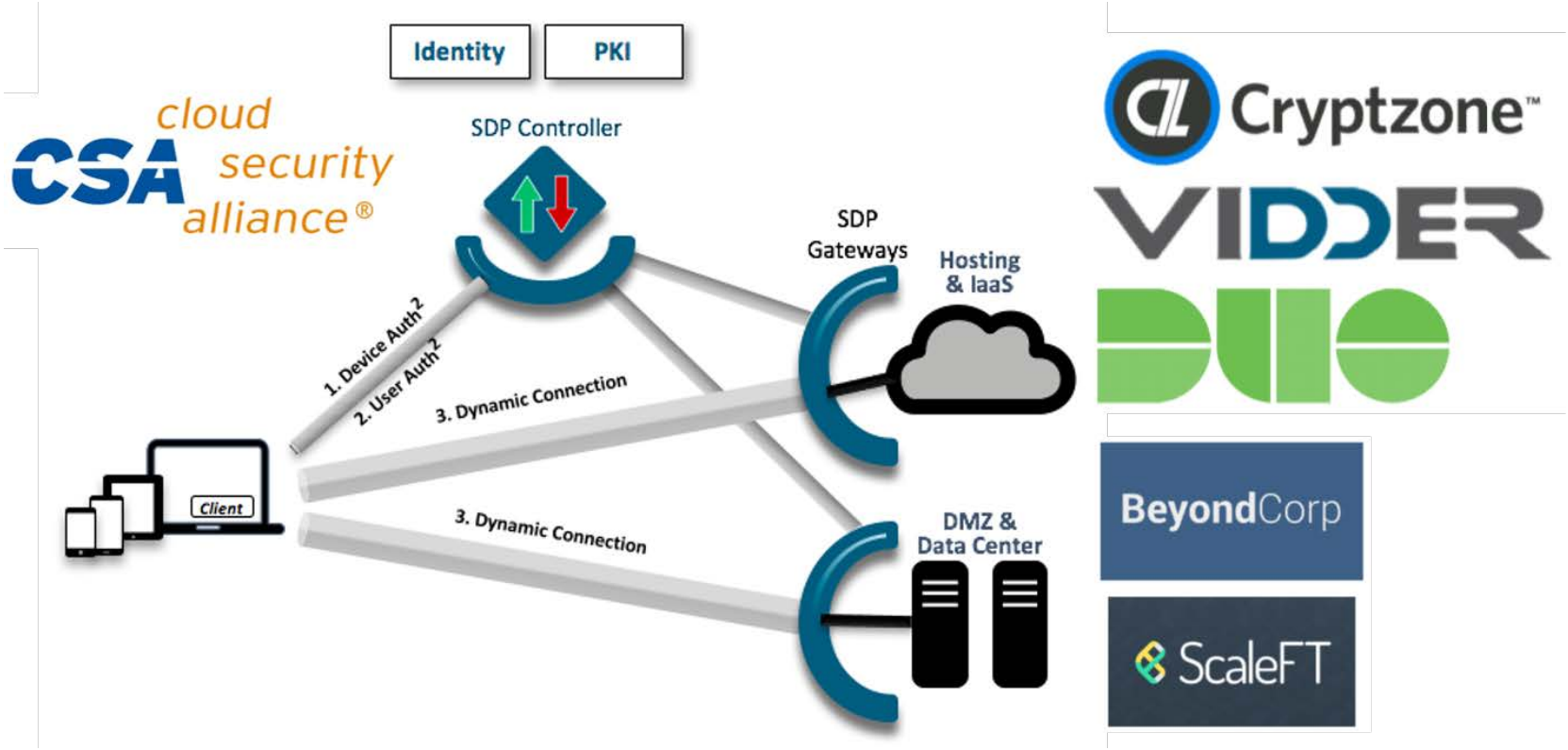


Contramedidas en SDN

- Emplear TLS 1.3 (o UDP/DTLS) para cifrar el tráfico entre los dispositivos de red y los controladores y autenticar ambos extremos de la comunicación mediante certificados.
- Disponer de una arquitectura de alta disponibilidad de los controladores.
- Prevenir el acceso no autorizado a la red de control.
- Separar el tráfico northbound del resto de tráfico.
- Asegurar los dispositivos de red y los sistemas donde corren los controladores.
- Monitorizar los controladores de cerca para detectar actividades sospechosas.
- Emplear prácticas de programación segura para las aplicaciones SDN que van a comunicarse por el Northbound API solicitando recursos SDN.
- Validar los flows en las tablas de los dispositivos de red frente a las políticas que hemos fijado en los controladores.
- Emplear los protocolos de Interconexión de Centros de Datos (Data Center Interconnect, o DCI) para autenticar los extremos de los túneles y securizar el tráfico en el túnel.

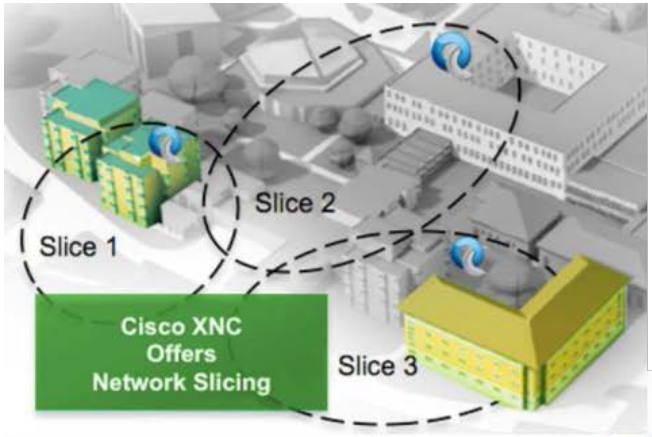
Ejemplos de seguridad con SDN

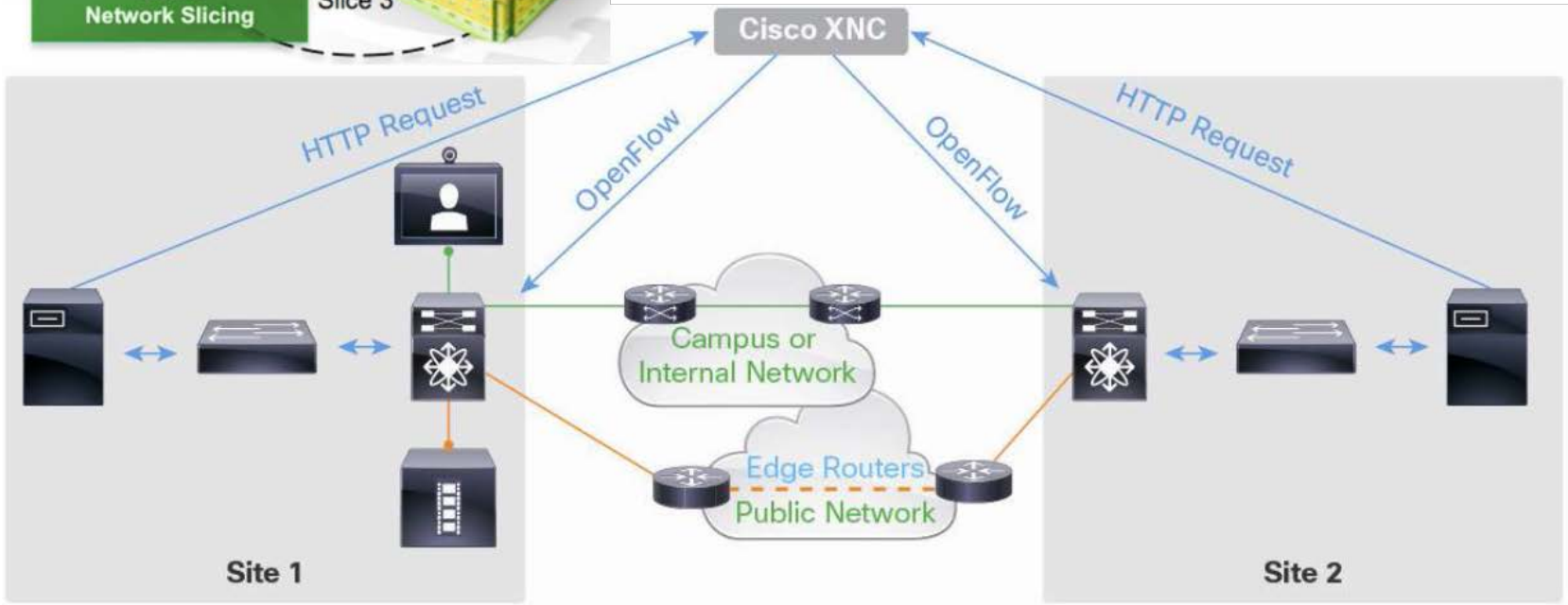
Filtrado de tráfico con SDN – Software-Defined Perimeter



Ejemplos de seguridad con SDN

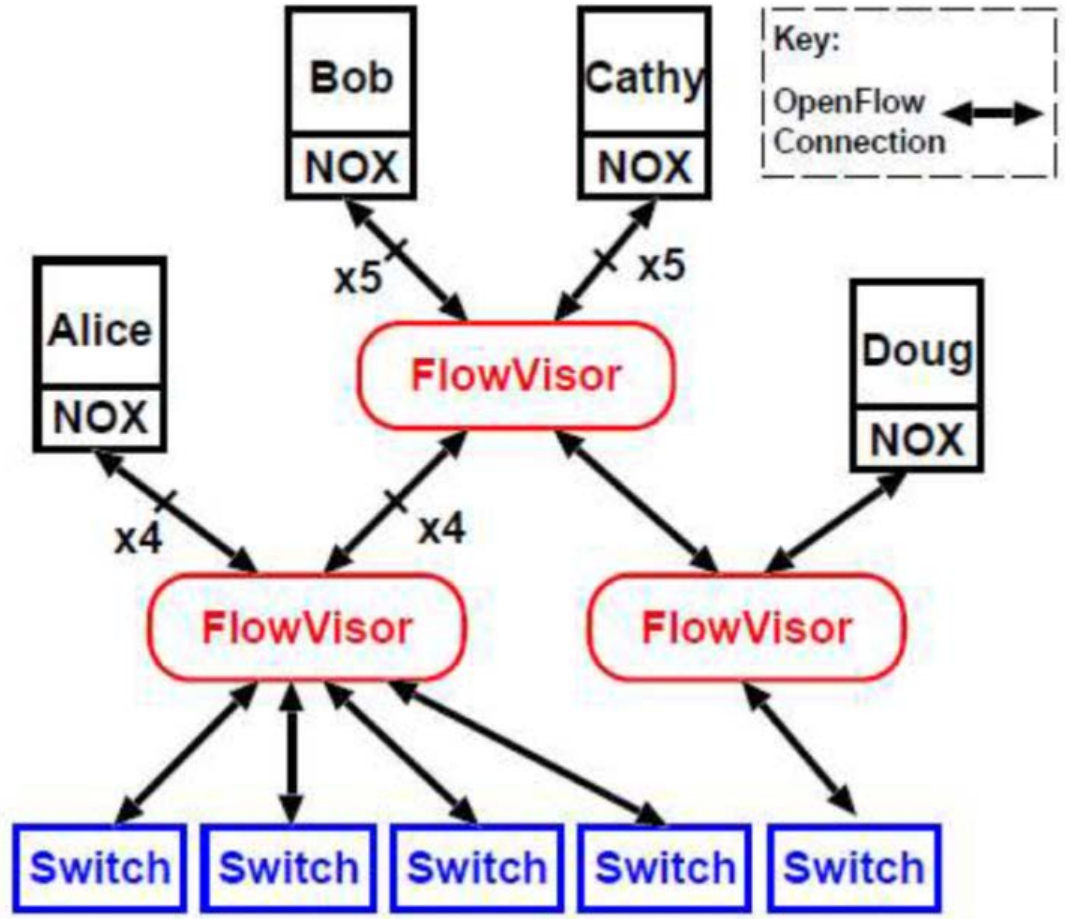
Segmentación de red





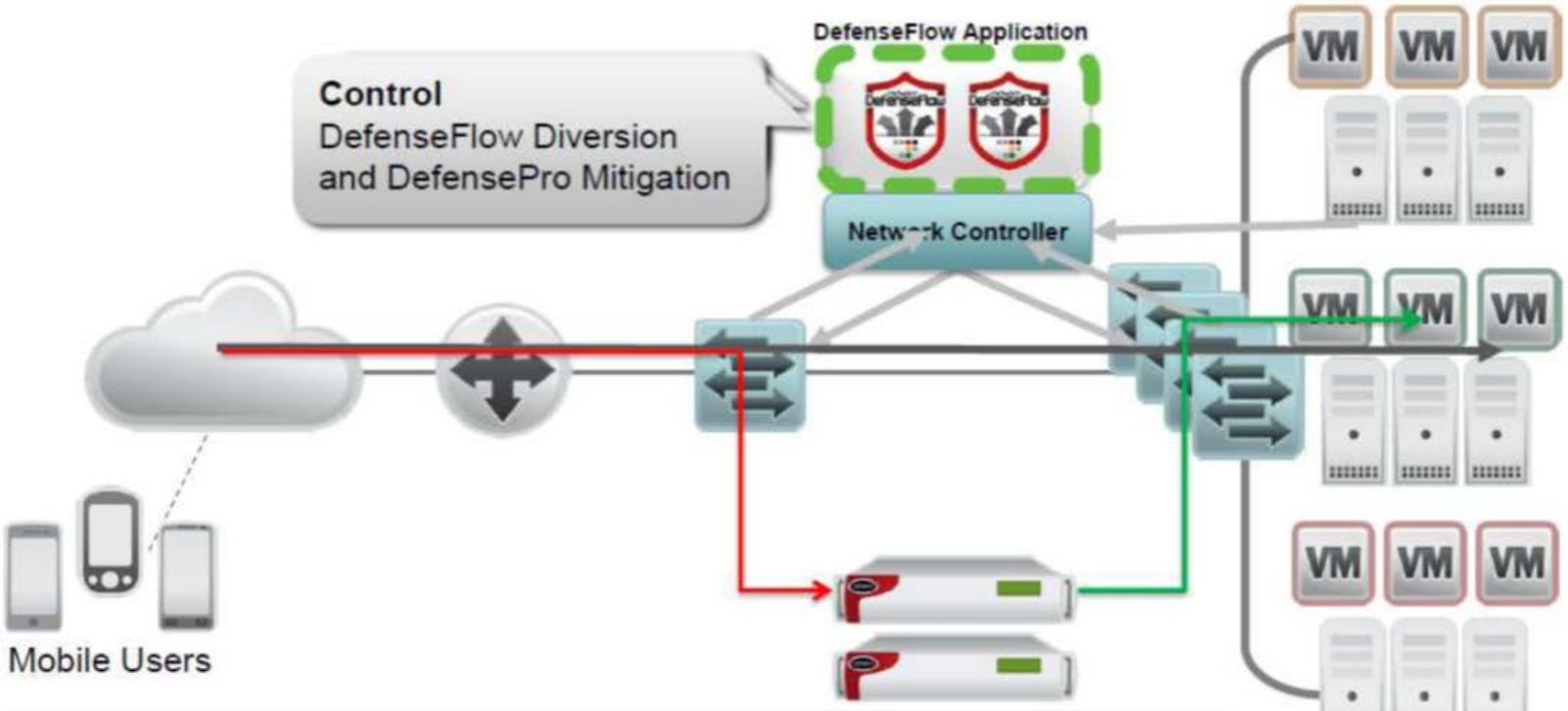
Ejemplos de seguridad con SDN

Segmentación de red



Ejemplos de seguridad con SDN

Mitigación de ataques DDoS

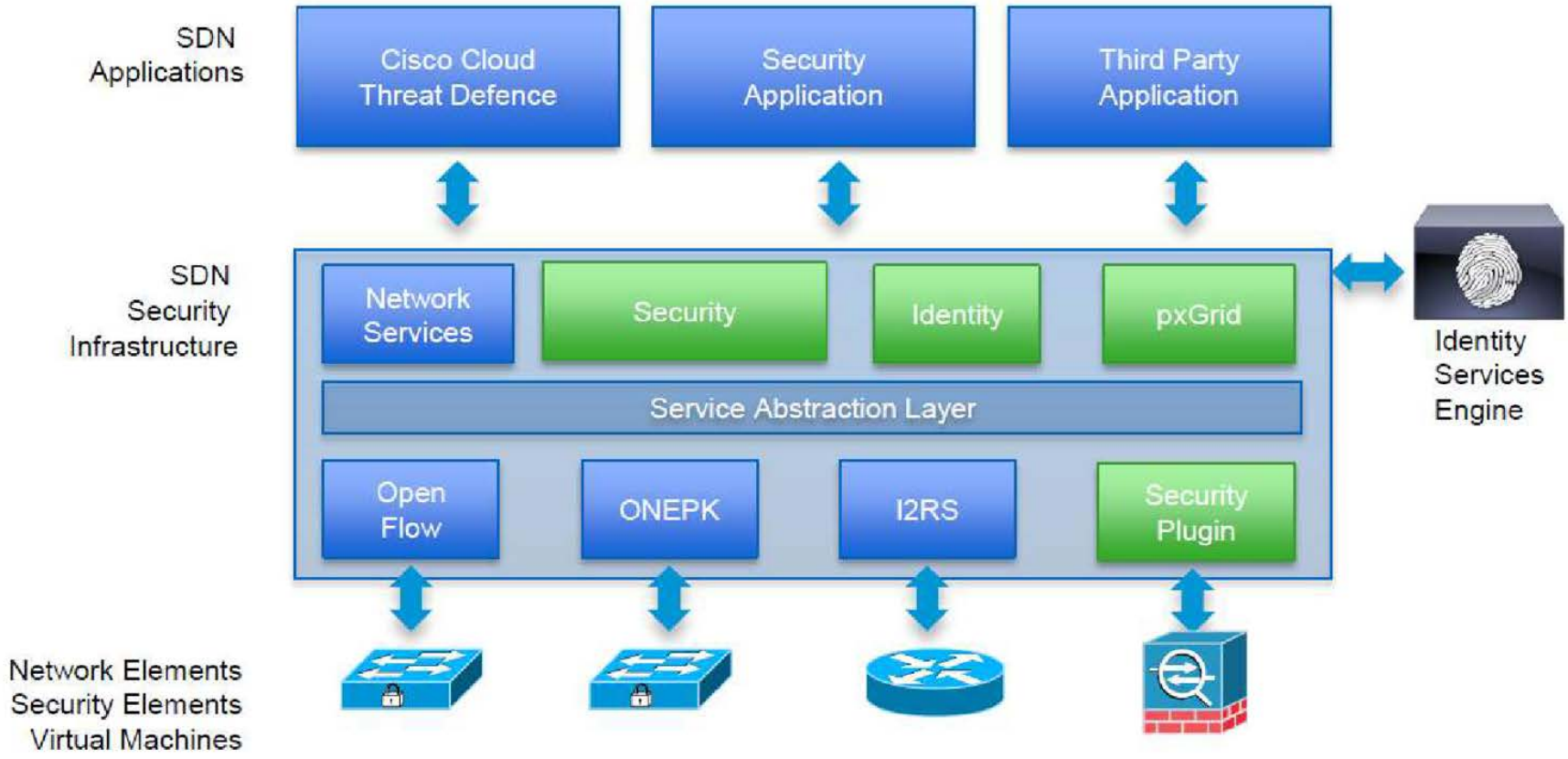


A completely new solution architecture:

- From point security solution to network-wide solution enabled by SDN
- Dynamic, programmable, scalable, easy-to-operate security network service
- Best possible design:
  - Always out of path except for under attack
  - Unprecedented attack detection span

Ejemplos de seguridad con SDN

Network Access Control (NAC)



Monitorización de tráfico con SDN