upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

Redes de Nueva Generación
Área de Ingeniería Telemática

Cloud

Cloud Computing

- *On-demand self-service*
  - El usuario puede crear nuevas instancias de servidores, almacenamiento o red por su cuenta
- *Universal network access*
  - Acceso mediante tecnologías estándar desde cualquier plataforma
- *Resource pooling*
  - Recursos compartidos entre diferentes *tenants*
- *Rapid elasticity*
  - *Provisioning* rápido o automático para un rápido *scale-out* y *scale-in;* parecen recursos ilimitados para el usuario
- *Pay per use*

Public cloud

- Disponible para el público general o una gran industria
- Propiedad de una organización que vende estos servicios
- Ofrecidos típicamente a través de la Internet

Community cloud

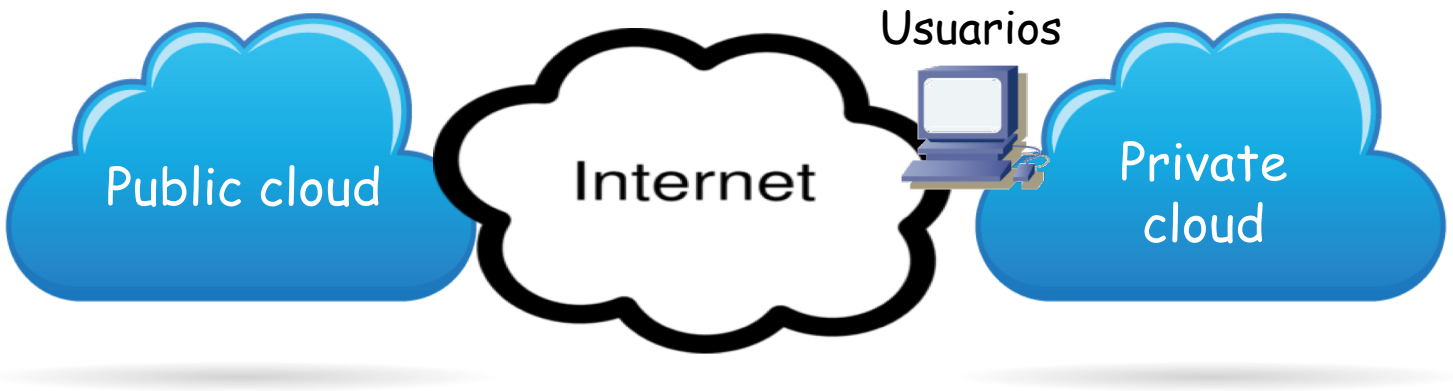
- Compartida por varias organizaciones
- Tienen características similares (misión, requerimientos de seguridad, políticas, cumplimiento necesario de regulación, etc)
- Gestionada por las organizaciones o por un tercero
- En sus propios edificios o en otros (*on-premises vs off-premises)*

Private cloud

- Empleada por una única organización
- Puede ser gestionada por la misma organización o por otra (servicio externalizado)
- Puede encontrarse en sus edificios o en otros

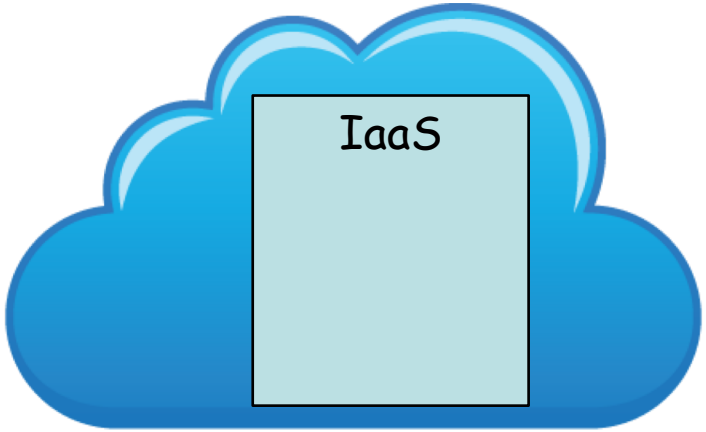
Hybrid cloud

- Utilización de infraestructura de al menos dos de los tipos anteriores para las mismas aplicaciones
- Por ejemplo:
  - Una empresa tiene su *private cloud* con recursos limitados
  - Cuando alcanza los límites, las peticiones en exceso se redirigen a una *public cloud*



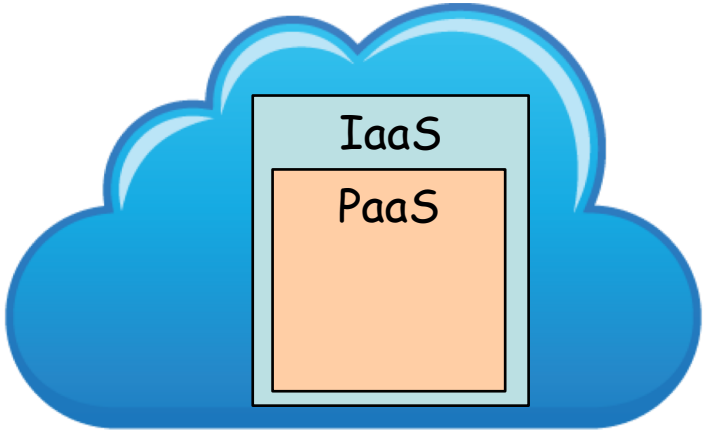
IaaS

- El cliente puede instanciar recursos de servidor, almacenamiento y/o red
- Tiene acceso a los servidores (virtuales) para poder emplear el sistema operativo que quiera
- Tiene control sobre esos sistemas operativos para instalar el software que necesite
- Ejemplos: Amazon EC2, CenturyLink Cloud, Microsoft Azure, Terremark vCloud Express, Arsys Cloud, Fujitsu IaaS Trusted Public S5, Google Compute Engine



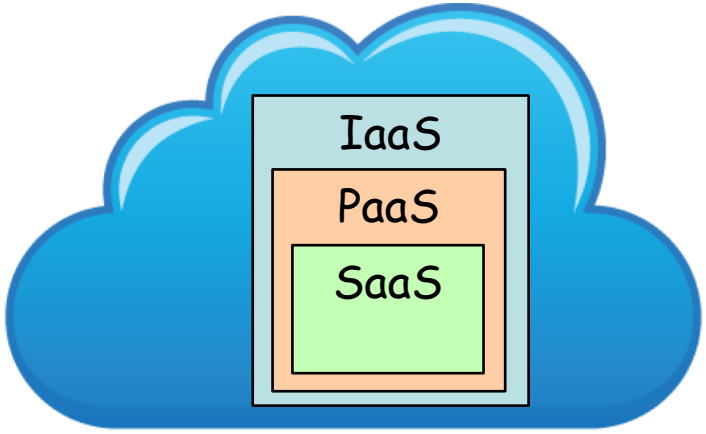
PaaS

- El cliente puede desplegar sus aplicaciones sobre la infraestructura
- Deben estar creadas empleando los lenguajes de programación y utilidades soportadas por ella
- No tiene control sobre la infraestructura
- Tiene control sobre las aplicaciones y tal vez su entorno de *hosting*
- Ejemplos: Google App Engine, AWS Elastic Beanstalk, OpenShift, Salesforce, Heroku



SaaS

- El cliente emplea las aplicaciones ofrecidas que están “en la nube” en lugar de instalarlas en sus equipos *on premises*
- Son accesibles desde diversos tipos de dispositivos
- No tiene control sobre la infraestructura, ni sistemas operativos, ni almacenamiento ni instalación de aplicaciones
- Ejemplos: Google Apps, Office 365, SharePoint Online, Cisco WebEx



upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

Redes de Nueva Generación
Área de Ingeniería Telemática

SDN

Internet

- Simple
- Control distribuido
- Ha permitido su gran crecimiento
- Muy bueno para los fabricantes de routers
- Sin embargo, ese crecimiento lleva a ser *commodity* (mercancía)

Equipos hardware

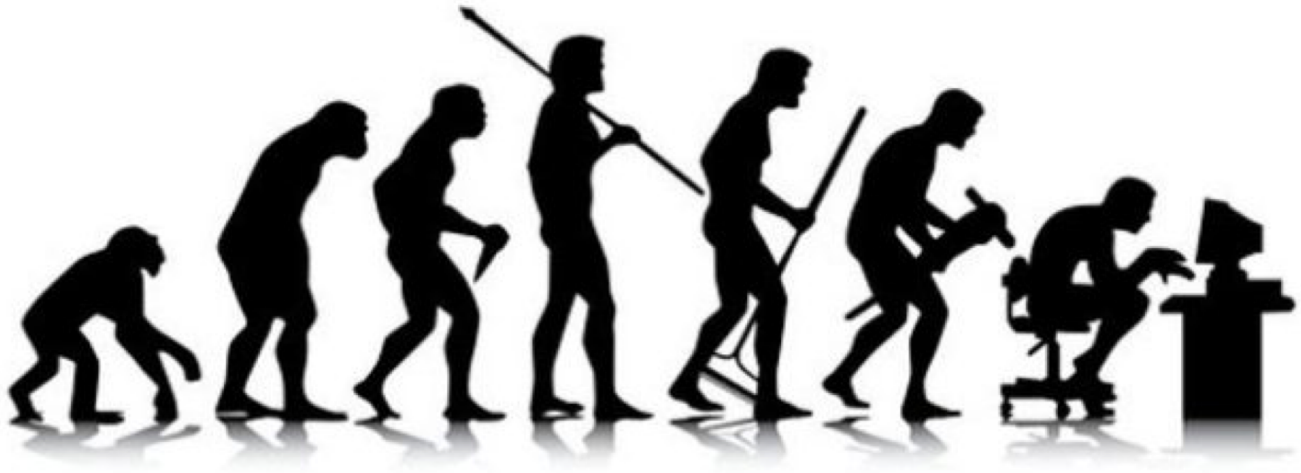
- La funcionalidad de red la dan equipos dedicados: switches, routers, ADCs, etc
- Dependientes de la implementación de funcionalidades en ASICs
- Eso hace su evolución muy lenta
- Y los hace propietarios respecto al desarrollador del ASIC
- Hoy en día estos ASICs y la conmutación en general son "*commodity*"
- El problema está en implementar nuevas funcionalidades
- La implementación de nuevos servicios es muy lenta en ASICs
- Es un entorno mucho menos flexible que el entorno software
- Por ejemplo si tenemos que esperar a que el fabricante del equipo corrija un bug o implemente una funcionalidad

¿SDN?

- *Software Defined Networking*
- ¿Qué es?
- El “hype” tiene poco tiempo (<10 años)
- Aunque se basa en ideas que tienen bastante más edad
- Hay confusión en los últimos años en lo que significa SDN
- ¿SDN = Un API estándar para configurar switches?
- ¿SDN = Separación del plano de datos y de control?
- ¿SDN = Plano de control centralizado?
- ¿SDN = OpenFlow?
- El movimiento actual sí empezó con OpenFlow pero hoy en día consideramos que no son lo mismo

Evolución

- El entorno informático evoluciona rápidamente
  - Tenemos virtualización en el sistema operativo y en el almacenamiento
  - Es hoy en día sencillo de gestionar y provisionar (mediante software)
  - Nuevos lenguajes de programación, nuevos sistemas operativos, todo ello simplifica el desarrollo de nuevos servicios
- Las redes no
  - Seguimos empleando el CLI a cada equipo; gestión primitiva
  - Cambiar los protocolos es lento pues requiere la colaboración de los fabricantes de los equipos
  - Y muchas veces cambiar el hardware

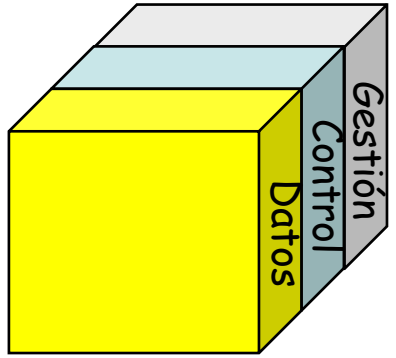


Evolución

- Las redes eran simples
- Ethernet, IP, son tecnologías muy simples
- Pero les hemos añadido nuevas necesidades
- En la parte del control de las mismas
- Y se nos han descontrolado
- ACLs, VLANs, Traffic engineering, Firewalls, VPNs, ADCs, DPI, TRILL, SPB, VXLAN, etc
- Las hacemos funcionar, porque tenemos en la cabeza cómo funcionan todos esos protocolos
- ¿Cómo simplificarlo?

Planos

Planes



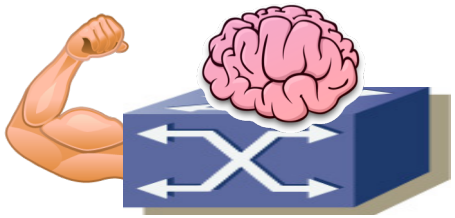
- *Data plane*
  - Paquetes de usuarios
  - Reenvío, fragmentación, replicación para multicast, etc
  - Simple y rápido (*fast path*)
- *Control plane*
  - Actividades necesarias para que funcione el plano de datos
  - Crear tablas de rutas, configurar políticas, anunciar servicios, etc
  - Complejo pero no necesita ser rápido (*slow path*)
- *Management plane*
  - Control opcional
  - Gestión de fallos, configuración, accounting, performance, seguridad, monitorización, provisionamiento, etc
  - Complejo pero no necesita ser rápido

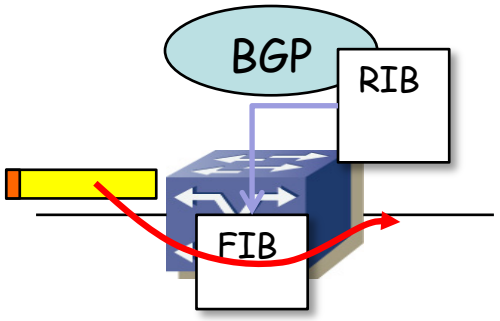
Complejidad y abstracciones

- La forma que tenemos de simplificar el problema es la modularidad
- Dividir el problema en problemas/módulos más pequeños
- Crear interfaces entre esos módulos

Control vs Data Plane

- *Data plane*
  - Conmutación, reenvío layer 2, reenvío IP (el “músculo”)
  - Aquí tenemos las abstracciones de las capas
- *Control plane*
  - Señalización y control, routing protocols (aprendizaje, el “cerebro”)
  - Los datos empleados para conocer la topología
  - Aquí no tenemos una forma abstracta de resolver el problema
- (…)





Control plane

- ¿Qué hacemos en el plano de control?
- Calcular la configuración de los dispositivos
  - Tablas de rutas
  - Listas de filtrado
  - Etc
- Tenemos que desplegar esta configuración a través de una red sin garantías
- Tenemos que desplegarla empleando el protocolo que tenemos

Control plane

- ¿Cómo hacemos evolucionar hoy en día el plano de control?
- Nos inventamos un nuevo protocolo desde cero (y esperamos a que se implemente, compremos hardware que lo soporte, etc)
- O reconfiguramos algún mecanismo existente (por ejemplo para hacer ingeniería de tráfico)
- O hacemos configuración manual (ACLs, middleboxes, routers domésticos, etc)

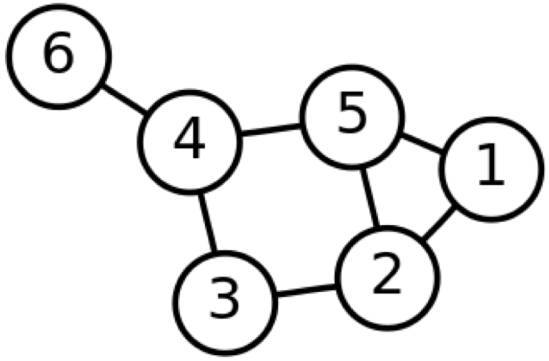
Analogía informática

- Calcular la configuración de los dispositivos
- Desplegarla a través de una red sin garantías
- Y empleando el protocolo que tenemos
- Especificar cómo es y dónde está cada bit del programa en memoria
- Gestionar los recursos hardware y todos los problemas que surjan para emplear la CPU, la memoria, etc
- Y lo tiene que hacer en Cobol

- ¿En realidad cómo lo hace un programador?
  - No se preocupa de cuáles son los bits resultado del programa ni de dónde está cada uno
  - El sistema operativo se encarga de esa gestión y comunicación
  - Desarrolla un lenguaje de programación de más alto nivel
- Es decir, resuelve el problema en cuestión, no todos los añadidos colaterales para hacerlo funcionar

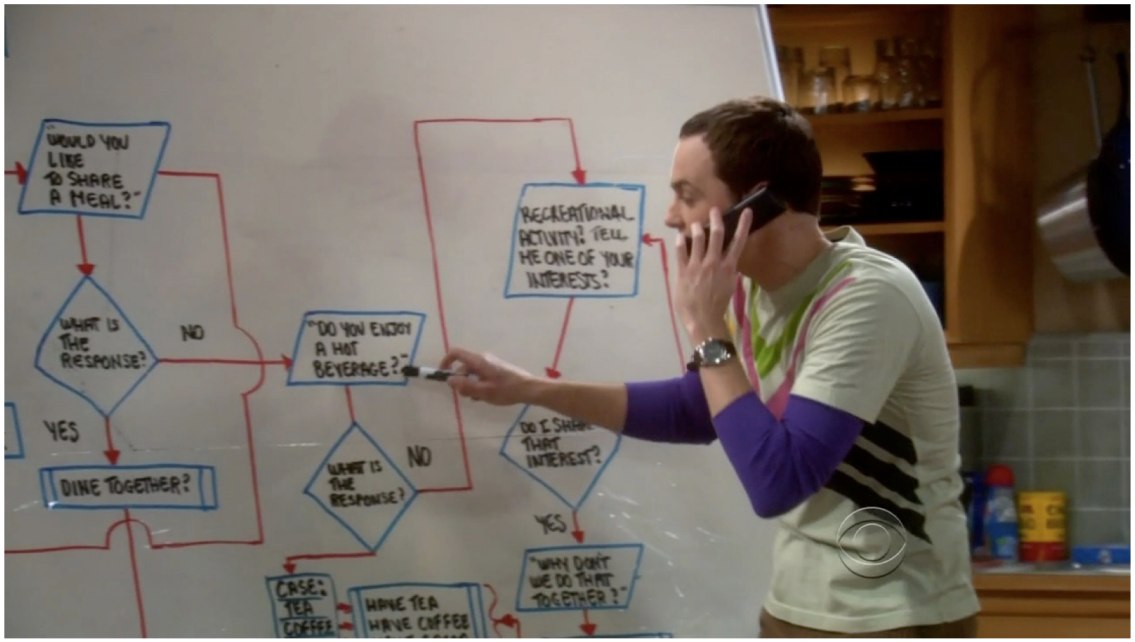
Abstracciones

- Estado distribuido
  - Tenemos que distribuir el estado (la configuración)
  - Esto ya está resuelto, no queremos volver a resolverlo
  - Para un problema queremos poder suponer que lo resolvemos de forma centralizada
  - No preocuparnos de cómo luego se distribuya



Abstracciones

- Especificación
  - El programa de control lo que quiere es especificar algo
  - Su problema no debería ser implementarlo
  - Necesitamos una abstracción de la red (virtualización)



Abstracciones

- Forwarding
  - Modelo para el comportamiento del plano de datos
  - Independiente del equipo, fabricante, etc

- Ejemplo:
  - De unas 240 páginas que tiene la RFC de OSPFv2 solo unas 20 son sobre el cálculo del camino (Dijkstra)
  - El resto es principalmente la distribución del estado

SDN

- Pretende ofrecer estas abstracciones
- No el “cómo”
- No es una solución a un problema, no es un nuevo protocolo o una nueva tecnología
- Es una nueva arquitectura
- Es un facilitador de nueva innovación
- *“Software Defined Networking”*
- Networking quiere evolucionar hacia el software
- El software definirá las redes, cómo se comportan, etc
- La red pasa a ser infraestructura abstracta como para las redes lo son los enlaces
- Que podremos administrar y controlar automática y dinámicamente
- Hablamos de esto solo desde hace menos de 5-10 años
- Todas las organizaciones están hablando ahora de ello: IETF, ITU, ONF, ETSI, etc.

upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

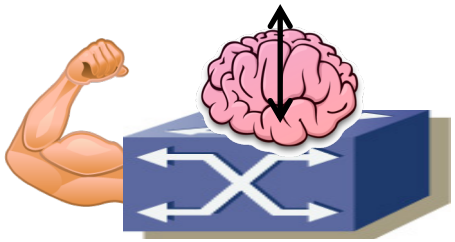
Redes de Nueva Generación
Área de Ingeniería Telemática

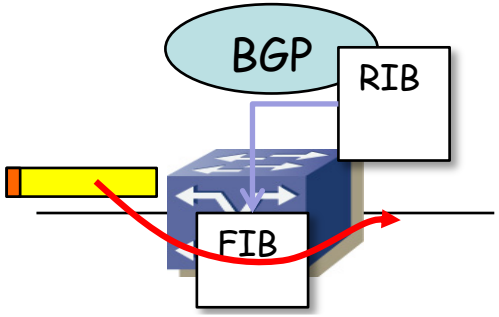
Elementos en SDN

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

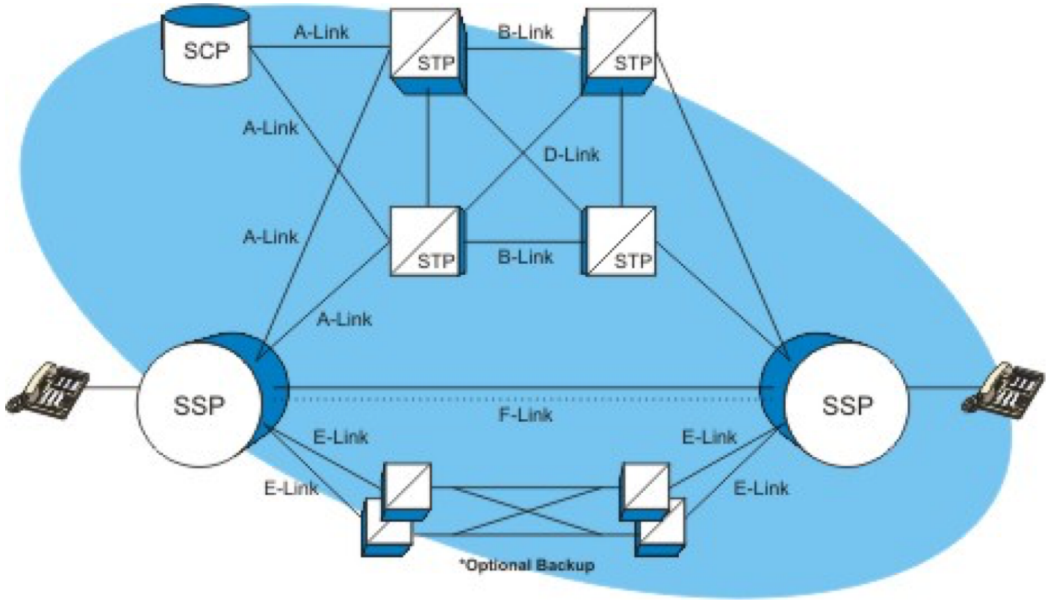
¿ OpenFlow ?





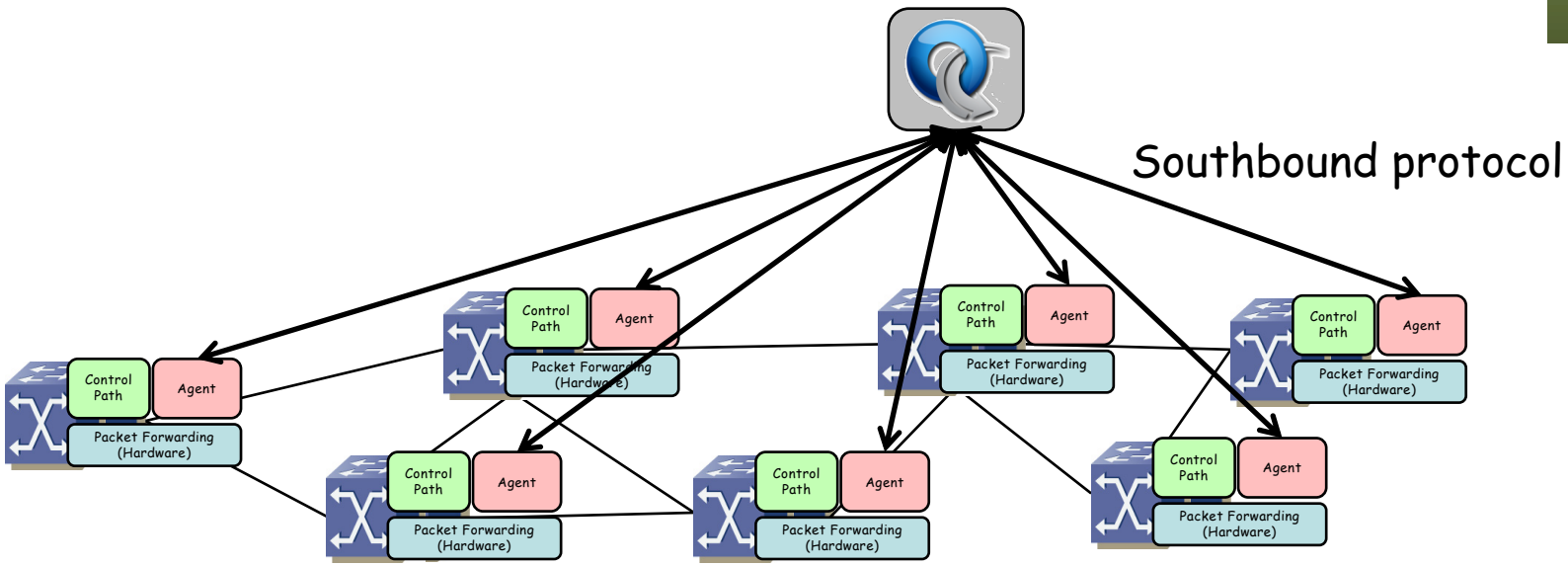
Separación control-datos

- 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



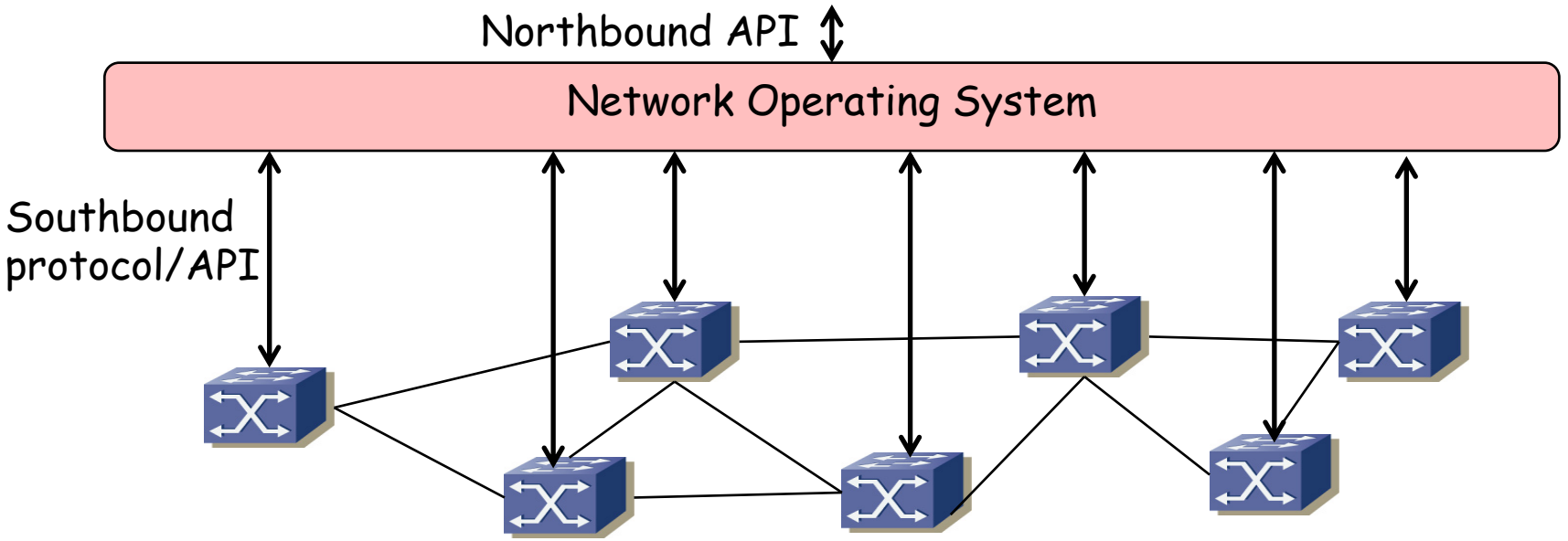
Controlador

- 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



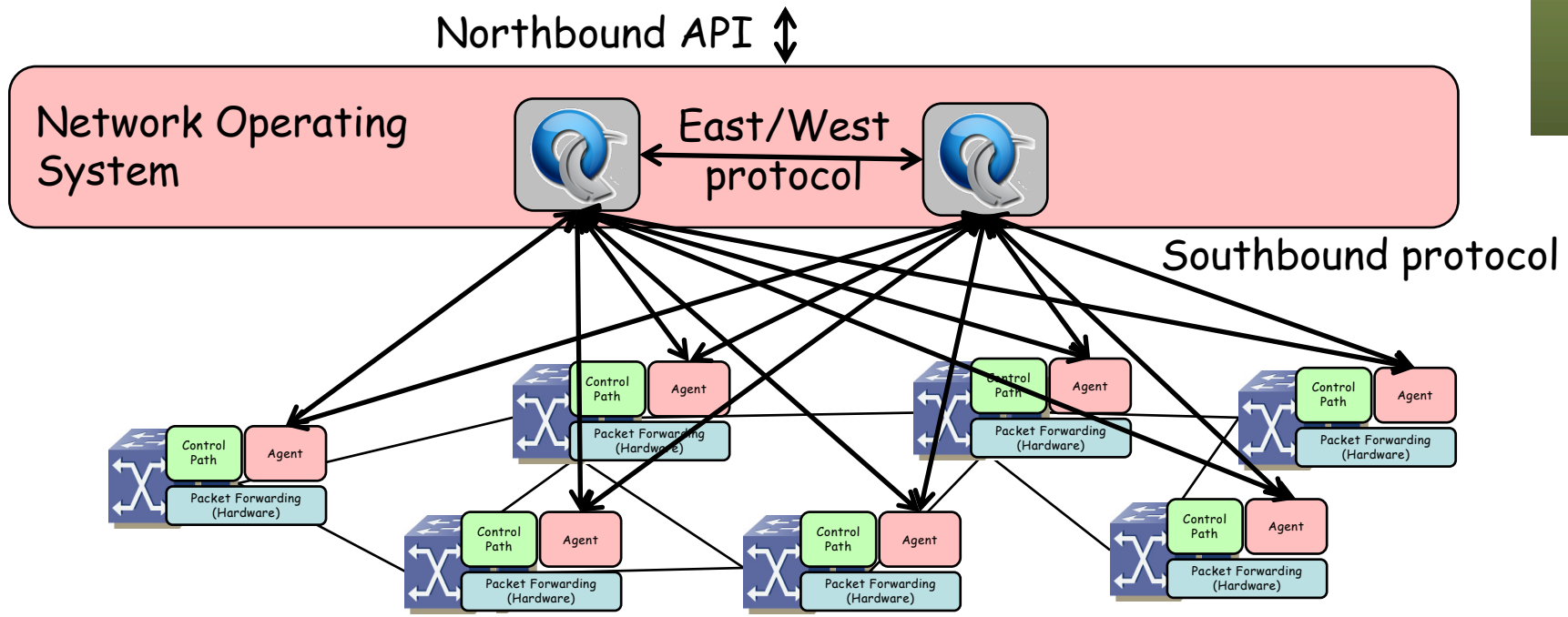
Network Operating System

- 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*



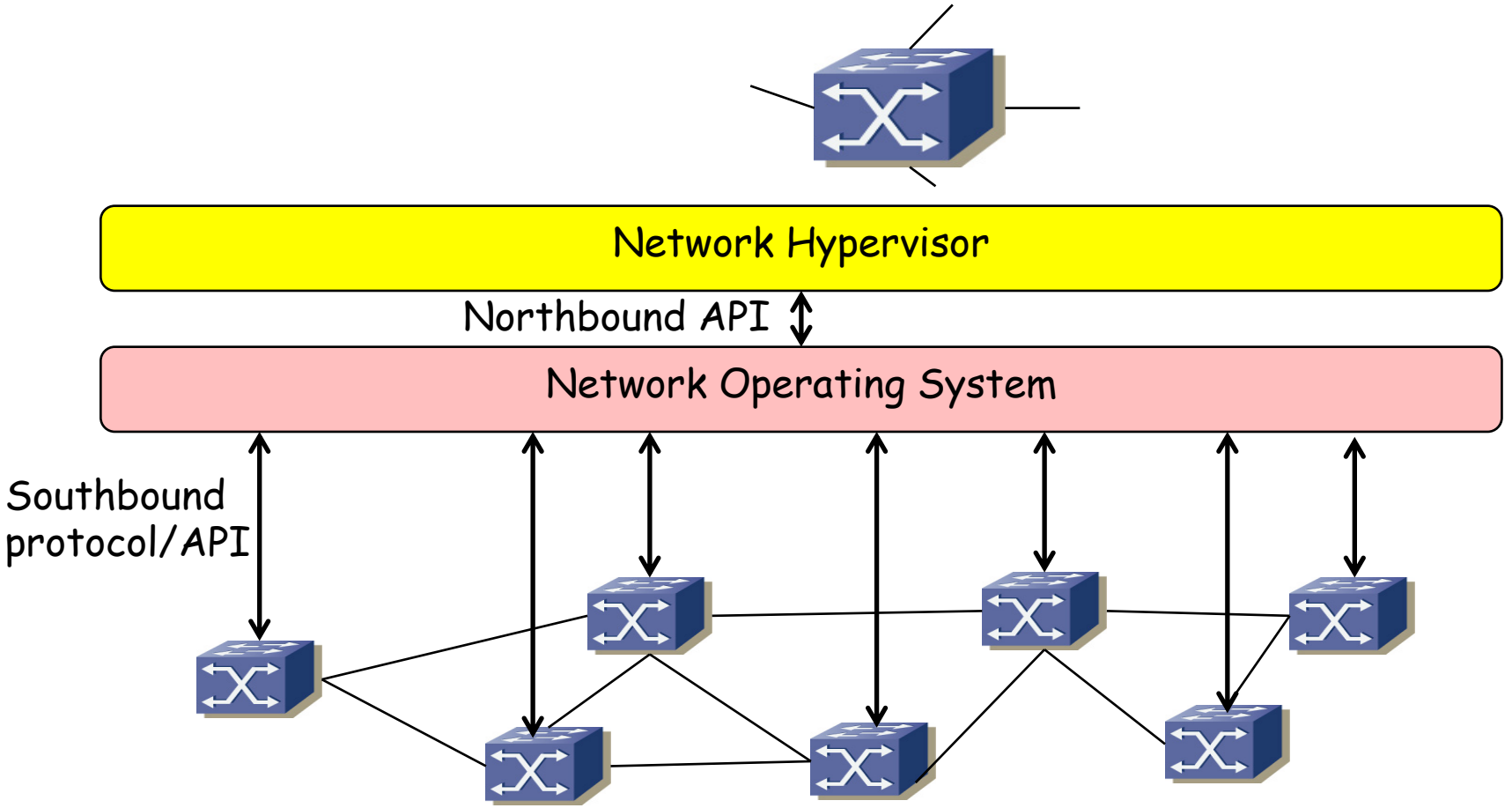
upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

Redes de Nueva Generación
Área de Ingeniería Telemática

Network Virtualization

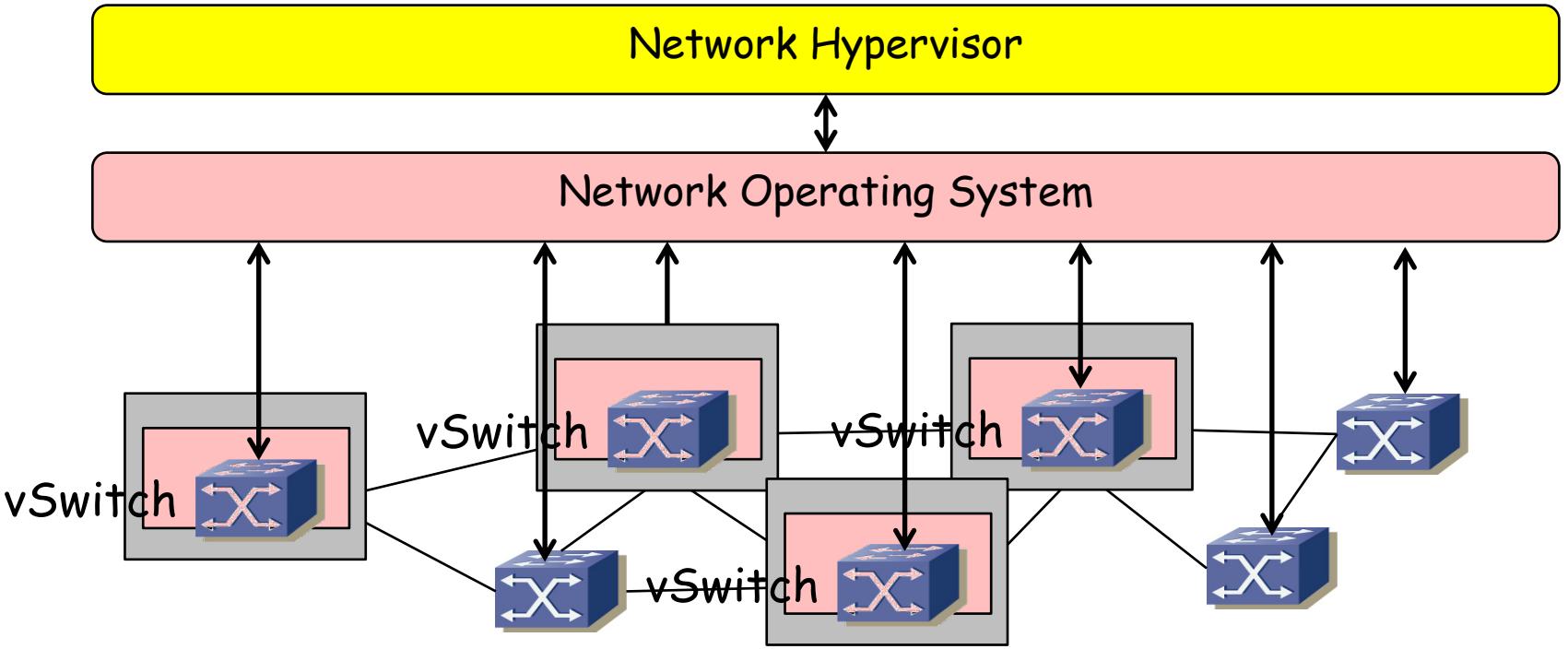
Network Virtualization

- 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?
- 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?

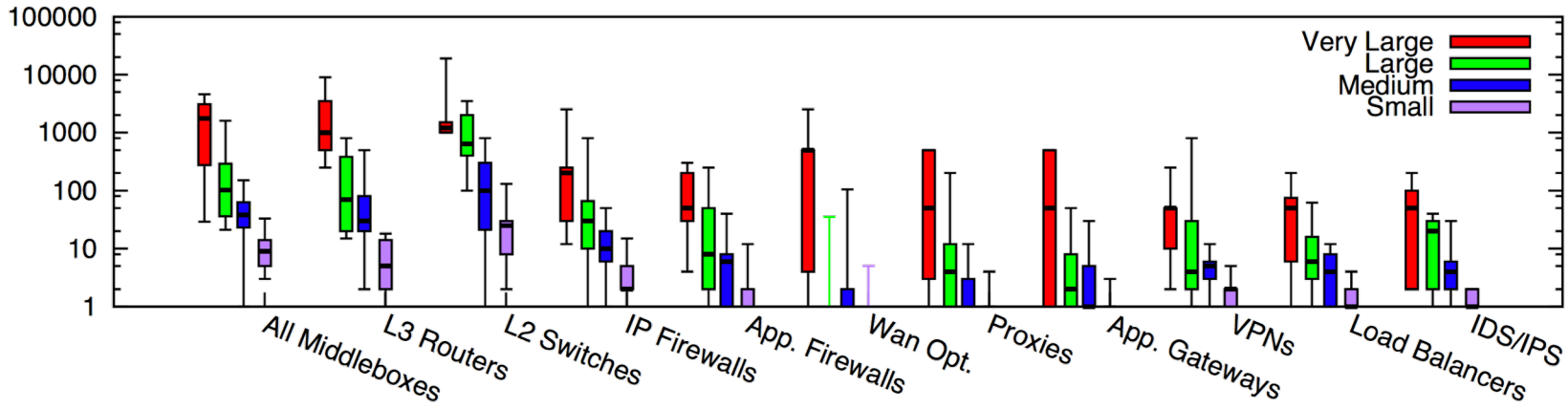


Figure 1: Box plot of middlebox deployments for small (fewer than 1k hosts), medium (1k-10k hosts), large (10k-100k hosts), and very large (more than 100k hosts) enterprise networks. Y-axis is in log scale.

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