Presentación

Area de Ingeniería Telemática
http://www.tlm.unavarra.es

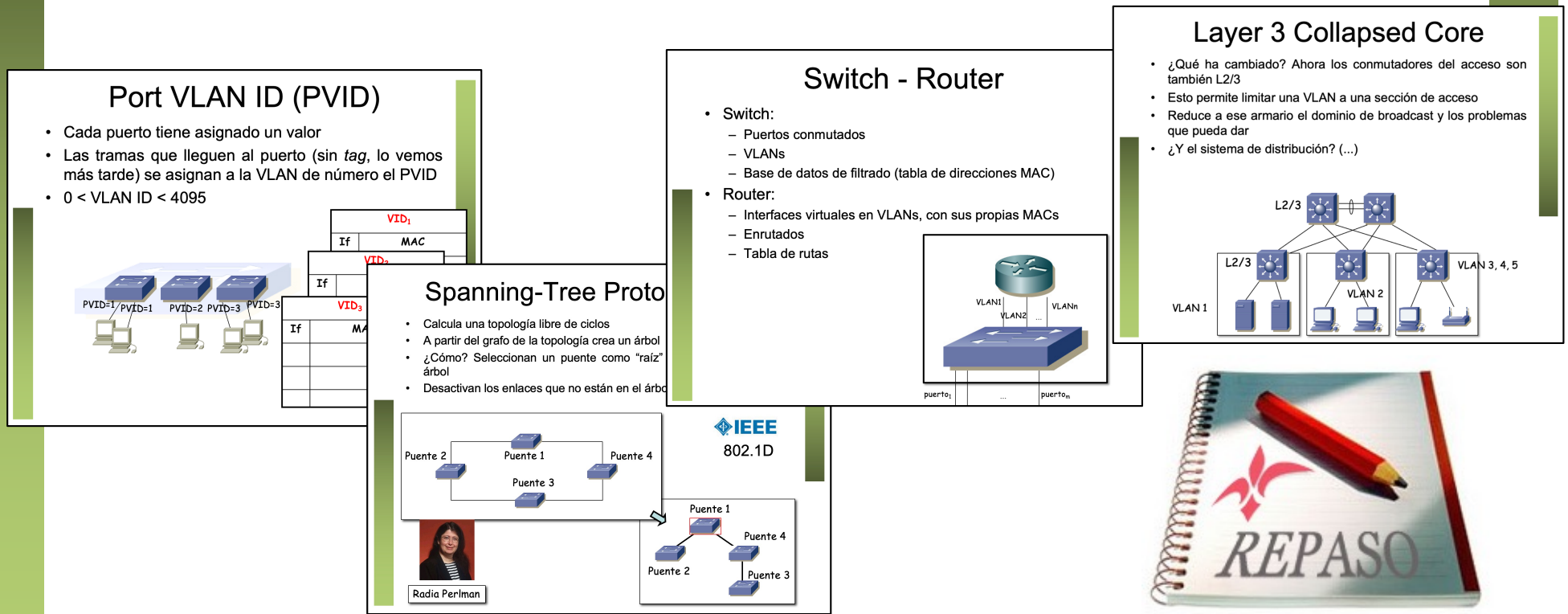
Máster en Ingeniería de Telecomunicación

Redes de Nueva Generación

- Vamos a hablar sobre Centros de Datos (Datacenters)
  - Sobre la arquitectura de sus redes
  - Sobre las nuevas tendencias tecnológicas para formarlas
  - Sobre almacenamiento y almacenamiento en red
  - Sobre virtualización (de servidor, de almacenamiento y de red)
  - Sobre “la nube”, cloud computing y redes definidas por software
- Sobre redes de acceso, MAN y WAN
  - Tecnologías
  - Ingeniería de tráfico
- Y sobre análisis y dimensionamiento, problemas de rendimiento que surgen, arquitectura de conmutadores, costes, etc.

Conceptos previos

- Sobre redes
  - Conmutación Ethernet, tecnologías Ethernet, VLANs, STP/RSTP/MSTP, conmutadores capa 2/3, VRRP, firewalls, routing IP, NAT
  - Diseño de LANs (con QoS, routing, gestión y seguridad)
  - Tecnologías de acceso: xDSL, FTTH, cable
  - Tecnologías WAN: ATM, PDH, SDH, MPLS



Conceptos previos

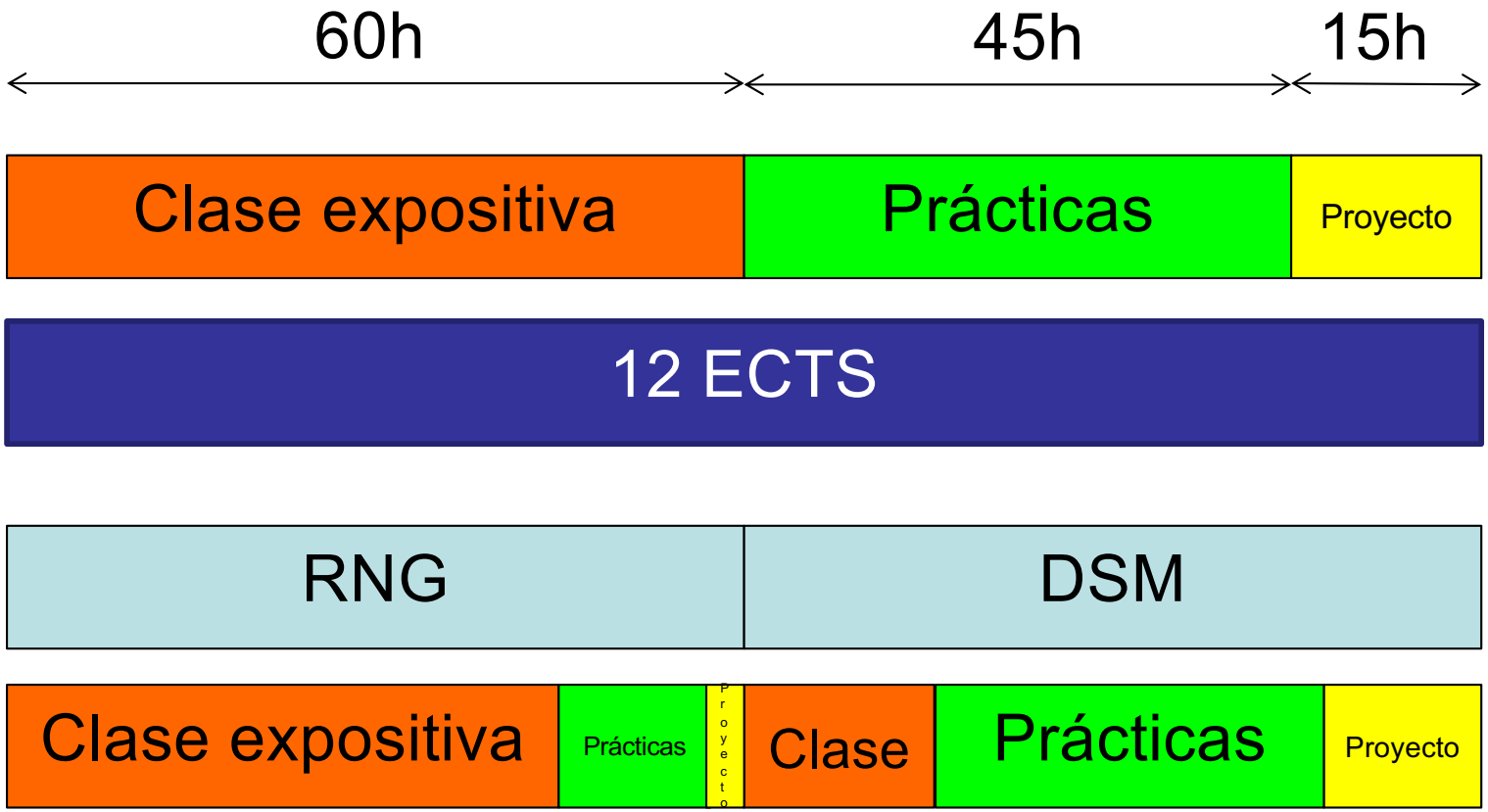
- Sobre servicios
  - Arquitecturas de servicio en capas (*tiers*)
  - Protocolos de transporte (TCP) y aplicación (DNS, HTTP, etc)
  - Rendimiento de protocolos y aplicaciones
  - CDNs
  - VoIP

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

Cuestiones administrativas

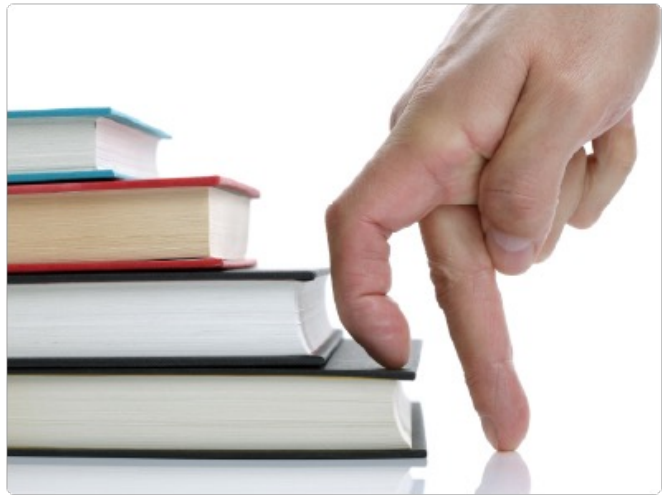
Esta materia en la UPNA

- "Diseño e implantación de servicios en redes de comunicaciones"
  - "Redes de Nueva Generación"
  - "Despliegue de Servicios Multimedia"



Temario

0. Introducción
1. Tecnologías para el centro de datos
2. Interconexión de redes
3. Modelado y dimensionamiento de redes y servicios



Actividades de laboratorio

- Virtualización de servidor y equipos de red
- Networking con contenedores
- Introducción a OpenFlow
- Análisis y presentación de casos de planificación, diseño e implementación de infraestructuras para redes multiservicio

Horarios

- Miércoles de 15:00 a 17:00
- Viernes de 15:00 a 17:00
- Comenzaremos con teoría en los dos días
- Periodo lectivo (clases) hasta el 24 de mayo
- Cuando hayamos visto suficiente teoría se plantearán las actividades de laboratorio en el mismo horario
- Recuperación de las primeras semanas previsto para las últimas 2 semanas en horario de 720512

	LUNES	MARTES	MIÉRCOLES	JUEVES	VIERNES
15:00-17:00	SISTEMAS INTEGRADOS Y EMBEBIDOS (720510)	DESPLIEGUE DE SERVICIOS MULTIMEDIA (720511)	REDES DE NUEVA GENERACIÓN (720508)	DESPLIEGUE DE SERVICIOS MULTIMEDIA (720511)	REDES DE NUEVA GENERACIÓN (720508)
17:00 19:00	SISTEMAS INTEGRADOS Y EMBEBIDOS (720510)	SISTEMAS INTEGRADOS Y EMBEBIDOS (720510)	DIRECCIÓN Y GESTIÓN DE PROYECTOS TIC II (720512)	DIRECCIÓN Y GESTIÓN DE PROYECTOS TIC II (720512)	SISTEMAS DE COMUNICACIONES AVANZADAS (720509)
19:00-21:00	SISTEMAS DE COMUNICACIONES AVANZADAS (720509)	SISTEMAS INTEGRADOS Y EMBEBIDOS (720510)	DIRECCIÓN Y GESTIÓN DE PROYECTOS TIC II (720512)	DIRECCIÓN Y GESTIÓN DE PROYECTOS TIC II (720512)	SISTEMAS DE COMUNICACIONES AVANZADAS (720509)

Planificación tentativa

Días	Actividades	Actividades	
21 feb y 23 feb	Presentación	Cambios en 802.3 y 802.11	
28 feb y 1 mar	Arquitectura del servicio. DCs	Diseño DCs. NICs	
6 mar y 8 mar	Virtualización	Virtualización. Almacenamiento	
13 mar y 15 mar	SAN y NAS	Virtualización	
20 mar y 22 mar	Prácticas	Prácticas	
27 mar y 29 mar	Contenedores	FESTIVO	
10 abr y 12 abr	VXLAN	Prácticas	
17 abr y 19 abr	Prácticas	FESTIVO	
24 abr y 26 abr	Balanceadores y servicios de red	Prácticas	
1 may y 3 may	FESTIVO	Prácticas	
8 may y 10 may	Prácticas	Consolidación I/O, limites Arq. DCs	
15, 16 y 17 may	Conmutadores. TRILL y SPB	SDN y NFV	Interconexión DCs. MPLS y BGP
22, 23 y 24 may	L3VPNs. L2VPNs	HTTP2+, MPTCP	Tráfico. Fin
7 jun	Examen		
14 jun	Recuperación		

Evaluación

- Examen: 6 ptos
  - Sobre todo el temario
  - Nota mínima del 50% para sumar el resto
  - Sin apuntes
- Actividades de laboratorio: 4 ptos
  - En cuanto demos suficiente teoría
  - Puntos de control y documentos

Resultados de aprendizaje	Actividad de evaluación	Peso (%)	Carácter recuperable	Nota mínima requerida
R1, R2, R3, R4	Prueba escrita que recoja los conceptos adquiridos (examen sin apuntes)	60	Sí	30% de la nota total de la asignatura
R3, R4	Actividades de laboratorio de resolución de problemas prácticos y comprensión de conceptos	30	No	
R2, R3, R4	Revisión de una propuesta tecnológica novedosa (informe individual)	10	No	

http://www.unavarra.es/ficha-asignaturaDOA?languageId=100000&codPlan=7205&codAsig=720508&anio=2023

802: Cambios físicos recientes

802.3: Nuevas versiones

- **802.3bz (2016)**
  - 2.5GBase-T y 5GBase-T empleando cat. 5e y 6 (gran planta instalada)
  - Útil para salidas de Access Points
- **802.3cb-2018**
  - 2.5Gb/s y 5Gb/s en backplane
- **25GBase-CR, 25GBase-SR**
  - Sobre twinaxial cable (5m) o fibra multimodo (802.3by-2016)
  - Más eficiente que 40G (40G es 4x10G, 25G es 1 *lane*)
  - Mejor utilización de capacidad de conmutación de ASICs (2015 ya a 3.2Tbps)
- **25GBase-LR, 25GBase-ER (802.3cc-2017)**
  - 10 ó 40Km sobre fibra óptica monomodo
- **25GBase-T, 40GBase-T (802.3bq-2016)**
  - 30 metros sobre par trenzado categoría 8
- **25GBase-BR10, 25GBase-BR20, 25GBase-BR40 (802.3cp-2021)**
  - Single strand single-mode fiber (10Km, 20Km, 40Km)
- **50GBase-BR10, 50GBase-BR20, 50GBase-BR40**
  - Single strand single-mode fiber (10Km, 20Km, 40Km)
  - 802.3cp-2021 (también 10 Gb/2 y 25 Gb/s)
- **50GBASE-CR, 50GBASE-FR, 50GBASE-KR, 50GBASE-LR, 50GBASE-SR**
  - 802.3cd-2018
- **100GEth**
  - 1ª generación 10x10Gbps, 2ª 4x25Gbps
  - Reduciendo tamaño y consumo
  - Eso permite mayor densidad de puertos
- **802.3cd-2018**
  - 50Gb/s, 100Gb/s y 200Gb/s Ethernet sobre twinaxial, fibra o backplane
- **802.3cu-2021**
  - 100GBase-FR1, 100GBase-LR1, 400GBase-FR4, 400GBase-LR4-6
  - 10Km, SMF (100Gb/s per wavelength)
- **802.3bs-2017**
  - 200Gb/s y 400Gb/s
  - 400GBase-SR8, 400GBase-DR4, 400GBase-XDR4, 400GBase-FR4, etc
- **802.3cm-2020**
  - 400GBASE-SR4.2 y 400GBASE-SR8
  - 2 pares y 8 pares de fibra multimodo respectivamente, 100m
- **802.3cn-2019**
  - 50 Gb/s, 200 Gb/s y 400 Gb/s fibra monomodo
- **802.3ck-2022**
  - Eléctricos 100, 200 y 400 Gb/s con señal 100Gb/s
- **802.3db-2022**
  - 100, 200 y 400 Gb/s sobre f.o. MM usando señal a 100 Gb/s sobre cada wavelength

802.3: Otras modificaciones

- **802.3br-2016**
  - *"Specification and Management Parameters for Interspersing Express Traffic"*
  - Latencia para iniciar la transmisión de un paquete *express* debe ser < 2 veces el tiempo del paquete mínimo + IPG (añade soporte para tráfico preemptivo)
  - *eMAC = express MAC; pMAC = preemptable MAC*
- **802.3bw-2015**
  - 100Base-T1 Ethernet sobre un solo par trenzado balanceado (15m)
- **802.3bp-2016**
  - 1000Base-T1 Ethernet sobre un solo par trenzado (15m)
- **802.3cg-2019**
  - 10Base-T1L (1Km), 10Base-T1S (15m), single balanced pair of conductors
- **802.3ch-2020**
  - 2.5GBase-T1, 5GBase-T1 y 10GBase-T1 sobre par balanceado (Automotive Electrical Ethernet) (15m)
- **802.3cz-2023**
  - 2.5, 5, 10, 25 y 50 Gb/s sobre fibra óptica de vidrio (GBase-AU)
- **802.3bn-2016**
  - 10GPass-XR, EPON sobre coaxial
- **802.3ca-2020**
  - EPON compatible con 10Gb/s EPON, 20 km, Split ratio 1:32
  - 25/10 Gb/s, 25/25 Gb/s, 50/10 Gb/s, 50/25 Gb/s y 50/50 Gb/s
- **802.3cs-2022**
  - Super-PON. Hasta 50 km y 1024 ONUs
  - 10/10 Gb/s y 10/2.5Gb/s

Wi-Fi 5 y 6

802.11ac (wave 1)

- 1.3 Gbps (PHY rate), 845 Mbps (MAC throughput)
- SU-MIMO, 3 streams espaciales
- Canal de 20 MHz, 40 MHz ó 80 MHz; aprox. 25, 12 ó 6 canales

802.11ac (wave 2)

- 3.47 (PHY), 2.26 Gbps (MAC), MU-MIMO (solo downlink)
- Canal hasta 160 MHz; hasta 4 streams espaciales
- AP 4x4 puede enviar a 4 clientes 1x1 simult.
- Evita que desaprovechen el tiempo de uso del medio
- Requiere *explicit transmit beamforming*

802.11ax (mayo 2021)

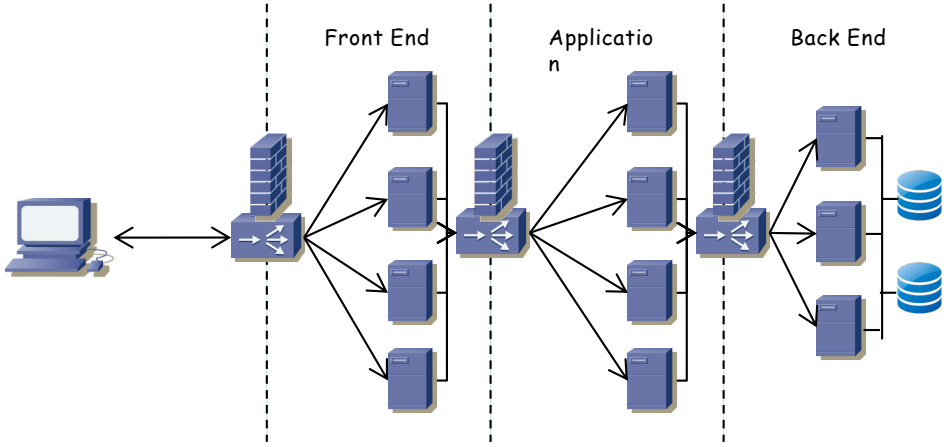
- 2.4 GHz y 5 GHz; hasta 1024-QAM
- Wave 1 puede soportar 8 streams. Beamforming
- OFDMA o MU-MIMO hasta 8 clientes, en wave 2 también en uplink
- Wi-Fi 6E
  - 6 GHz sin licencia (1.2GHz de BW); hasta 7 canales de 160 MHz no interferentes
  - Aprox. 2Gb/s con latencia sub-ms hasta 3m, 1.4 Gb/s a 7m
  - MU-MIMO uplink; nuevo MAC

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

Arquitectura del servicio

Arquitectura del servicio

- Antes de las LANs y la arquitectura PC el *mainframe* era accedido desde terminales
- Los terminales eran *thin clients*, el trabajo pesado lo hacía el *mainframe*
- Está más orientado al trabajo en bloques (*batch*)
- Cliente-servidor; servidores de menor capacidad que *Mainframe*
- Clientes de mayor capacidad que terminales “tontos” (*thick client*)
- Servidores como hardware independiente, distribuidos por la red de la empresa
- Interfaces de aplicaciones propietarios hasta llegar la web
- Se migra de una arquitectura básica c/s a una basada en web
- Se sigue lo que se conoce como el modelo *n-Tier*
- La aplicación puede estar en un servidor, pero seguramente con usuarios remotos
- ¿Cómo escalar?
  - Verticalmente (***scale-up***)
  - Horizontalmente (***scale-out***) aumentando el número de servidores y repartiendo el trabajo entre ellos
  - Pueden ser de capacidad moderada y bajo coste
  - Ahora se requiere una forma de repartir el trabajo entre ellos (más complejo)



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

Enterprise network

Computer room

- Salas o edificios; centralizan recursos de cálculo y almacenamiento críticos para la empresa
- Contienen: mainframes, servidores web, servidores de aplicaciones, servidores de documentos y de impresión, sistemas de almacenamiento, equipos de infraestructura de red, etc
- Centraliza recursos de refrigeración y alimentación ininterrumpida
- Contienen información crítica para los procesos de la empresa
- Muchas empresas hoy en día cerrarían en unas semanas si perdieran los datos que tienen
- Contienen los recursos centralizados de almacenamiento y cómputo
- Se habla de los "recursos de TI" (Tecnologías de la Información) o en inglés "IT" (*Information Technologies)*
- Hoy en día esto es tanto informática como comunicaciones

Mainframe:

- IBM: *"a large computer, in particular one to which other computers can be connected so that they can share facilities the mainframe provides".* "*Big iron*"
- Probablemente el ejemplo clásico es el IBM System/360 ('60s)
- Computación centralizada, segura, robusta, predecible
- Capacidad para dividirlo en sistemas más pequeños (LPARs o Logical Partitions), cada uno es un mainframe separado
- Se viene "profetizando" su desaparición desde los '90s
- En esa época surge la alternativa distribuida: Servidores independientes distribuidos, hardware de menor coste
- La Internet surge enfocada a ese entorno distribuido
- Orientado también al *e-business*
- Su propio OS (z/OS, z/VM) pero también otros con mucha virtualización (Linux)
- Lenguajes populares (C/C++, Java) y otros no tanto (PL/I, CLIST, REXX)
- Ejemplo (z16): Hasta 200 procesadores (x8 cores @5.2GHz superescalares. 7nm), hasta 10TB de RAM por armario (x4 armarios), hasta 320 puertos I/O, CPU con aceleración por hardware para encriptación, AI, compresión, etc. *7 nines availability*

Entorno distribuido

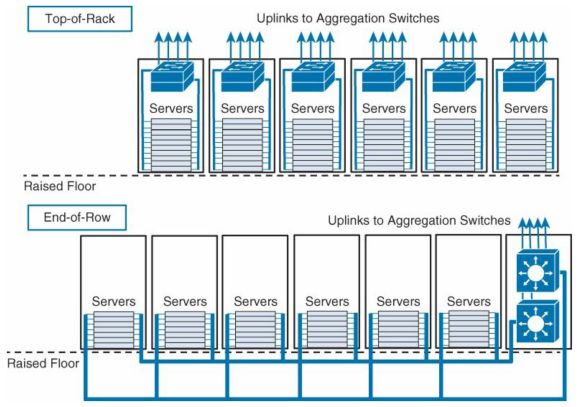
- La Internet surge como una interconexión de redes
- Servicios como la web son las *killer-apps* que la han llevado a su despliegue
- Esos servidores no eran *mainframes* sino hardware más genérico, arquitectura PC
- Ordenadores de sobremesa, torre, enracables…
- Servidor autónomo tiene problemas de escalabilidad al aumentar las peticiones de servicio
- Llega un momento que no puedes “poner uno más potente”
- Lo que se hace es poner varios servidores que se reparten el trabajo ¿Varios?¿Cuántos?
- Depende del número de usuarios/peticiones, pueden ser decenas… o decenas de miles
- Hemos vuelto a centralizar esos recursos de computación distribuidos
- Grandes data centers (decenas o centenares de miles de servidores)
- En ellos conviven los mainframes con grandes cantidades de servidores
- Existen varias soluciones de interconexión (Infiniband, Myrinet) pero nos centraremos en el entorno Ethernet

DCs

- Centralizamos el mantenimiento, lo cual permite una operación 24x7 más económica
- Simplifica la seguridad, tanto de red como física
- Más posibilidades de consolidación de servidores
- Consolidación de almacenamiento
- Evidentemente interesa que no se interrumpa
- Con servidores distribuidos empleamos SAIs (Sistemas de Alimentación Ininterrumpida, a.k.a. UPS)
- En DCs podemos:
  - Ante fallos de la red eléctrica tener conexión a dos proveedores (si existe esa disponibilidad)
  - Si hay solo un proveedor, recibir potencia de dos subestaciones diferentes
  - Disponer de un generador (normalmente diesel)
  - Baterías para mantener la alimentación en lo que se hace el cambio

Cableado

- Estándar ANSI/TIA-942-2005
- Dentro del propio armario no es trivial su “encaminamiento”
- Se encaminan a un conmutador en el armario o “Top-of-rack switch” (ToR)
- Lo más común es el suelo técnico pero
  - Si aumenta el peso de los racks pueden hacerlo poco estable
  - Inestable ante terremotos
  - El cableado interrumpe el flujo de aire
  - Difícil acceso al cableado
  - Riesgo de seguridad
  - Muy difícil de limpiar



DCs

Cableado

- Lo más común es el suelo técnico
- Con suelo sólido el cableado se eleva
- No solo está el cableado de datos, también alimentación
- El cableado dentro del armario puede interrumpir el flujo de aire
- Esto resulta en sobrecalentamiento de equipos
- Vigilar por dónde refrigeran los equipos y por dónde está pasando el cableado

Refrigeración

- Requieren un rango de temperatura para operar
- Centralizamos el calor pero también la refrigeración
- Sistemas empleando aire frío o agua
- Pueden sacar provecho del frío exterior
- Se crean pasillos “calientes” y “fríos”
- Es un sistema crítico que se mantiene redundado. Costoso (€€€)

Seguridad física

- Sin ventanas, muros gruesos
- Bunkers a prueba de bombas
- Alambradas
- Cámaras de vigilancia
- Autentificación biométrica
- Autentificación en varias capas de acceso

DCs

Costes

- En el ámbito de la granja de servidores empresarial se gasta en personal de operaciones
- En el caso del DC esos costes son mucho menores
- El DC está automatizado; personal:servidores del orden 1:1000
- Servidores (45%): Scale-out vs scale-up; economía de escala; resistencia ante fallos
- Infraestructura (25%)
  - Necesario para entregar potencia eléctrica y disipar calor
  - Generadores, transformadores, baterías; sistemas de refrigeración
- Power (15%)
  - Power Usage Efficiency (PUE) = Potencia total / Potencia gastada en equipamiento de IT
- Networking (15%)
  - Equipamiento interno en el DC (switches, routers, balanceadores, firewalls, etc)
  - También coste de conectividad WAN con ISP o entre DCs

Albert Greenberg, James Hamilton, David A. Maltx, Parveen Patel, "The Cost of a Cloud: Research Problems in Data Center Networks", ACM SIGCOMM Computer Communication Review, 39:1, Jan. 2009

Tiers

- Diferentes clasificaciones según la organización: Uptime Institute (https://uptimeinstitute.com), ANSI/TIA-942, etc
- Tier I (Basic Site Infrastructure): Sin redundancia
- Tier 2 (Redundant Site Infrastructure Capacity Components): Redundancia parcial: componentes de refrigeración redundados pero un único suministro eléctrico
- Tier 3 (Concurrently Maintainable Site Infrastructure): Suministro eléctrico y refrigeración redundados; redundancia N+1; equipamiento IT con fuentes de alimentación redundantes
- Tier 4 (Fault Tolerant Site Infrastructure): Cada servidor tiene alimentación duplicada, 2 procesadores y puede cambiar discos en caliente

DCs

Estructura de propiedad

- Data center propiedad de la empresa
- Data center propiedad de un proveedor que ofrece *hosting*
- Data center propiedad de un proveedor que vende el uso del equipamiento y se encarga de la gestión
- Data center de un proveedor, utilizados los equipos por un cliente, gestionados por un tercero
- Multi-tenancy
- ”Housing/Colocation”
- “Cloud Connect”
- “IX Peering”

Data centers modulares

- Contiene servidores, alimentación refrigeración, etc
- Unos 4-20 armarios
- Fácil de reubicar (en camión)
- Fácil de ampliar (pones otro armario)

http://www.astmodular.com/solutions/family/modular-data-centers_1

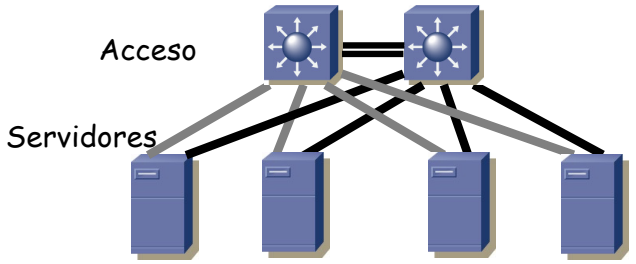
upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa
Redes de Nueva Generación
Área de Ingeniería Telemática

Diseño del data center

Single layer DC

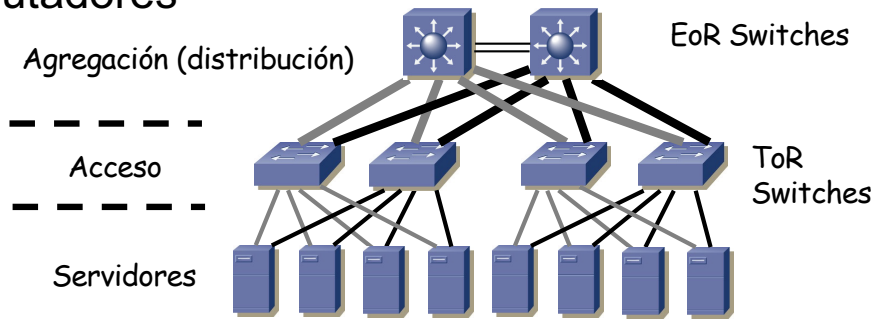
Single layer DC

- ¿Servidores en diferente Tier de aplicación?
  - VLANs, es decir, virtualización en la red
  - Routing entre ellas
  - Posible segundo interfaz en Tier Aplicación
- ¿Redundancia?
  - Requiere segundo interfaz en los hosts
  - ¿Cómo usar el segundo interfaz? Lo vemos más adelante
  - ¿Cómo hacer funcionar la redundancia? ¿STP? ¿ECMP?



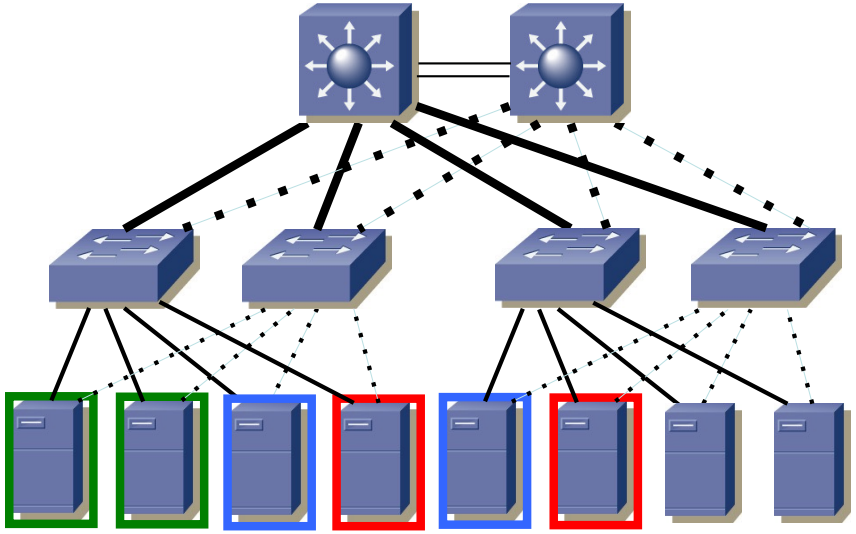
Dual Tier DC

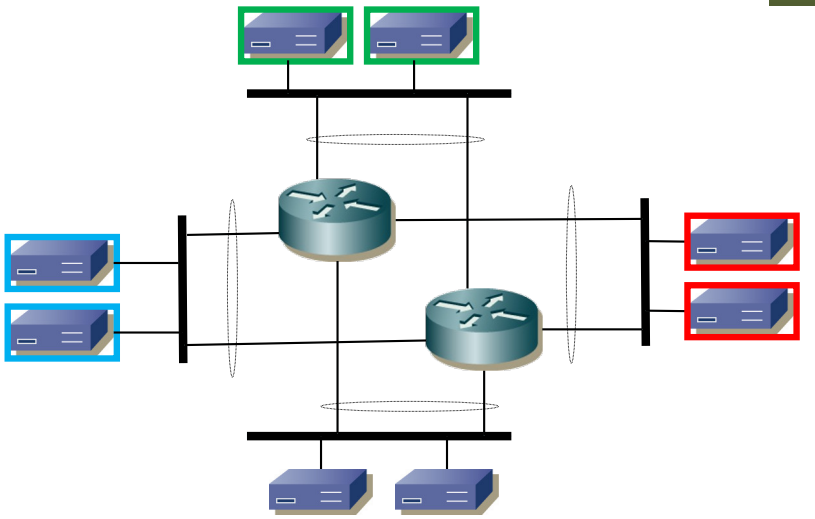
- Los conmutadores de la capa de acceso dan alta densidad de puertos
- Los conmutadores de agregación agregan tráfico hacia y desde el acceso y para conectar con los servicios de red
- No hay puntos únicos de fallo
- Enlaces gigabit o 10G a los servidores
- Enlaces gigabit, 10G o LAGs entre los conmutadores
- Todo full duplex, ¡nada de hubs!



Red para n-tier

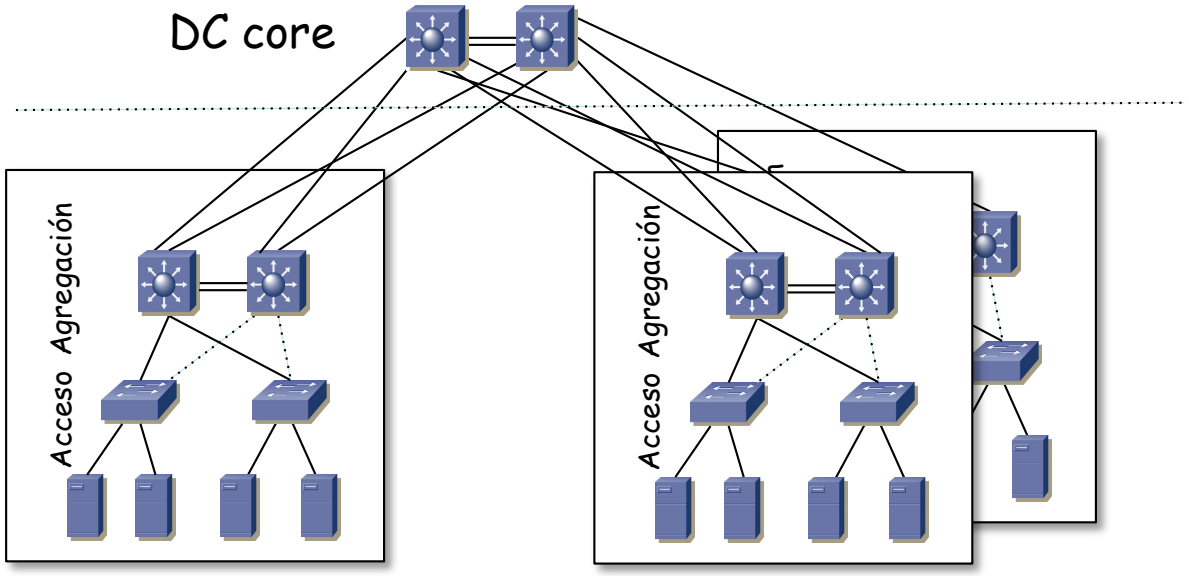
- Lo normal es emplear VLANs
- ¿Qué funcionalidades necesitamos entonces en la capa de agregación?
- Conmutación capa 2 en la VLAN, capa 3 entre ellas
- En la VLAN árbol de expansión
- Se puede emplear un FHRP (VRRP, HSRP,...)
- Posible MLAG con capa de agregación
  - Equipos de capa de agregación se comportan como uno solo
  - No necesita STP, no desactiva puertos
  - Redundancia y mayor capacidad mediante LAGs





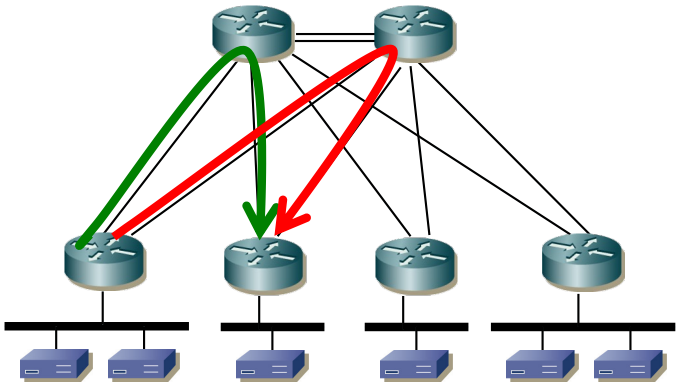
Data center core

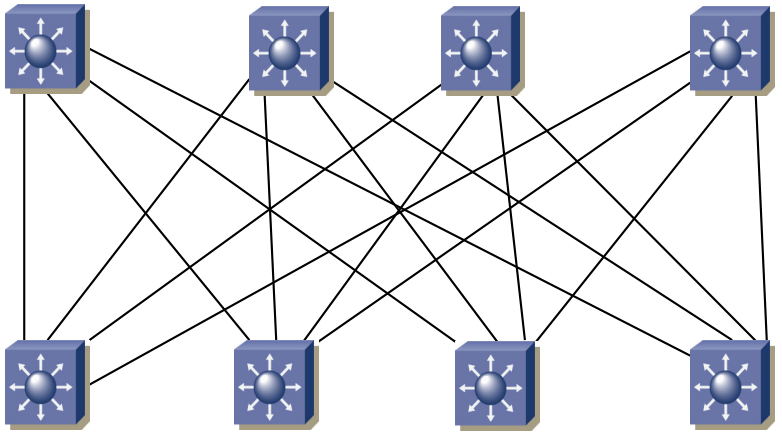
- Puede que no sea suficiente con un par de conmutadores de agregación
- Por ejemplo por limitación en número de puertos
- O por políticas que recomienden la separación de capas de agregación
- En ese caso añadimos una capa de *core* del data center
- Enlaces 10GE o superiores
- Enlaces agregación-core en capa 3
- Se conectará al core de la red Campus
- Pueden ser directamente los equipos del core del campus pero podemos estar limitados por número de puertos



L3 y ECMP

- Escenarios en los que tenemos enlaces capa 3
- Puede ser entre core y agregación o entre agregación y acceso
- Empleamos un protocolo de encaminamiento dinámico
- En muchas ocasiones estos protocolos pueden calcular varias rutas del coste mínimo
- Equipos en la misma VLAN deben estar conectados al mismo conmutador de acceso
- Configuración (subredes IP) más compleja
- Podemos aumentar la capacidad con LAGs (hemos puesto parejas de enlaces pero podrían ser más)
- Aumentar el número de caminos
- En este ejemplo hay 4 caminos de igual coste entre cada pareja de conmutadores
- Si las capas son core y agregación recordemos que por debajo están los conmutadores de acceso
- Si las capas son agregación y acceso recordemos que por debajo están los servidores
- Estamos limitados por el número máximo de puertos disponibles en los equipos



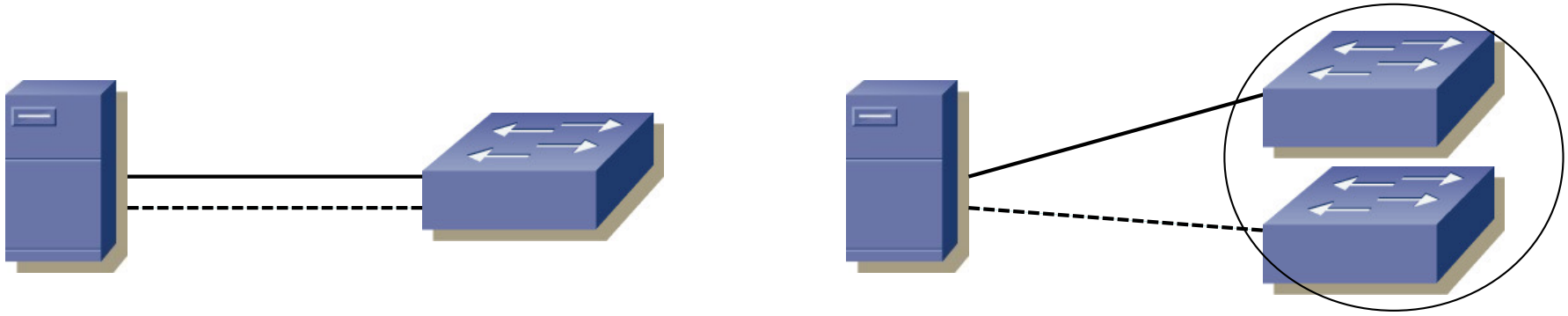


upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa
Redes de Nueva Generación
Área de Ingeniería Telemática

NICs Ethernet para servidor

Server multihoming

- Un segundo enlace, modo activo-pasivo
  - Si falla el primero (la NIC, el conmutador o el cable) se activa el segundo con la misma dirección MAC e IP
  - Se desaprovecha el segundo enlace
- O se usan los dos enlaces para transmitir pero solo se recibe por uno
- Cada interfaz suele enviar con diferente dirección MAC origen para no tener *MAC flapping* en el conmutador
- O se forma un LAG (802.3ad / 802.1AX)
  - Permite usar la capacidad de ambos enlaces
  - Normalmente requiere colaboración por parte del switch
  - Si se quiere redundancia de switch hay que hacer una agregación en la que un extremo son 2 conmutadores



Tareas en la NIC

- Por un enlace 10GE pueden llegar en 1 segundo más de 14 millones de tramas de 64 bytes
- Eso da a la CPU unos 67ns para procesar cada una
- Las CPUs tienen serios problemas para procesar en ese tiempo cabeceras TCP/IP
- Una NIC puede incluir electrónica para llevar a cabo ciertas tareas de TCP/IP descargando a la CPU
- La NIC puede incluir ASICs, Network Processors o un procesador con un sistema operativo de tiempo real
- A 400Gbps una trama cada 1,67ns lo cual está en el rango de los mejores tiempos de acceso a memoria

Checksum offload

- La NIC descarga del cálculo a la CPU; en transmisión y recepción
- Checksum IP (v4 y v6), UDP y TCP

Coalescencia de interrupciones

- Las NICs solían generar una interrupción por paquete
- Alto coste para la CPU
- Por ejemplo los mainframes tienen CPUs dedicadas a atender I/O
- La coalescencia hace que la NIC genere una interrupción para un grupo de paquetes en vez de por cada uno
- También puede hacer *polling* la NIC

(R)DMA

DMA (*Direct Memory Access)*

- Transferencia desde la NIC a memoria sin requerir a la CPU

RDMA (Remote Direct Memory Access)

- Remote Direct Memory Access
- Copias entre RAM de hosts diferentes sin involucrar a la CPU
- Latencia de pocos microsegundos
- iWARP
  - RFCs 5040, 5041, 5044
  - Sobre TCP o SCTP
- RoCE
  - RDMA over Converged Ethernet (DCB, Data Center Bridging)
  - RoCE v1 sobre Ethernet, v2 sobre UDP
  - RoCE v1 mecanismos de control de flujo y congestión de DCB
  - RoCE v2 emplea control de congestión basado en ECN

Jumbo frames

- Tramas Ethernet con MTU superior a 1500byes
- No están estandarizadas, la MTU estándar sigue siendo de 1500bytes
- Motivos para limitarlo
  - NICs tenían memoria limitada
  - Se quería limitar el tiempo que una estación tenía capturado el medio transmitiendo
  - El CRC es menos efectivo cuanto más grande es la trama
- Hoy en día no son problemas reales:
  - Decenas o centenares de Megabytes en la NIC
  - No tenemos medio compartido (ni coaxial ni hubs)
  - El CRC de Ethernet soporta más de 11 Kbytes de trama
- Diversos estándares han ido aumentando el tamaño máximo de la trama (802.1Q, 802.1ad, MPLS, FCoE, etc)
- A estas últimas en ocasiones se las llama “Baby Giant”
- Jumbo frames suelen estar cerca de los 9 Kbytes (que se puedan transportar bloques de datos de 8Kbytes + encapsulados varios)
- ¿Postivo?
  - Cuanto más grandes menor ratio de cabeceras y menos interrupciones
  - Menos carga de procesado de cabeceras en equipos de red y hosts
- ¿Negativo?
  - Todos los equipos del camino deben soportarlas
  - Posibles problemas con implementaciones que esperan 1500 bytes
  - Mayores tramas sufren mayor retardo así que no son adecuadas para todos los servicios
  - Mayores tramas pueden llenar antes los buffers de los conmutadores

LRO

LRO

- *Large Receive Offload, Receive Segment Coalescing*
- La NIC une varios segmentos TCP en uno solo
- Crea unas cabeceras TCP e IP para ese nuevo segmento
- Reduce el número de interrupciones y procesado de cabeceras en el kernel

RSS (*Receive Side Scaling)*

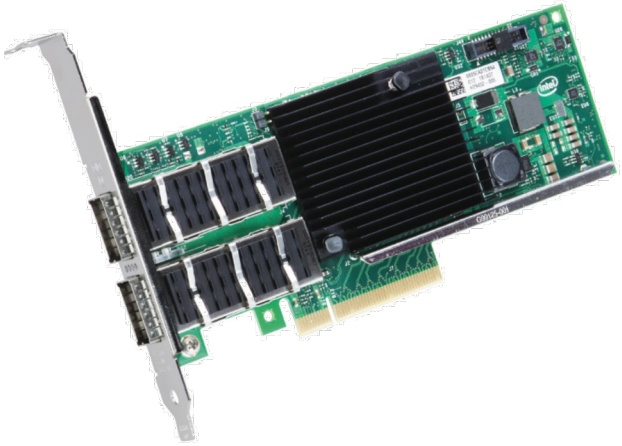
- Multi-CPU o CPU multi-core
- NIC calcula un hash sobre el paquete recibido y con él decide a qué CPU manda la interrupción
- Permite paralelizar entre varias CPUs el procesado del tráfico recibido

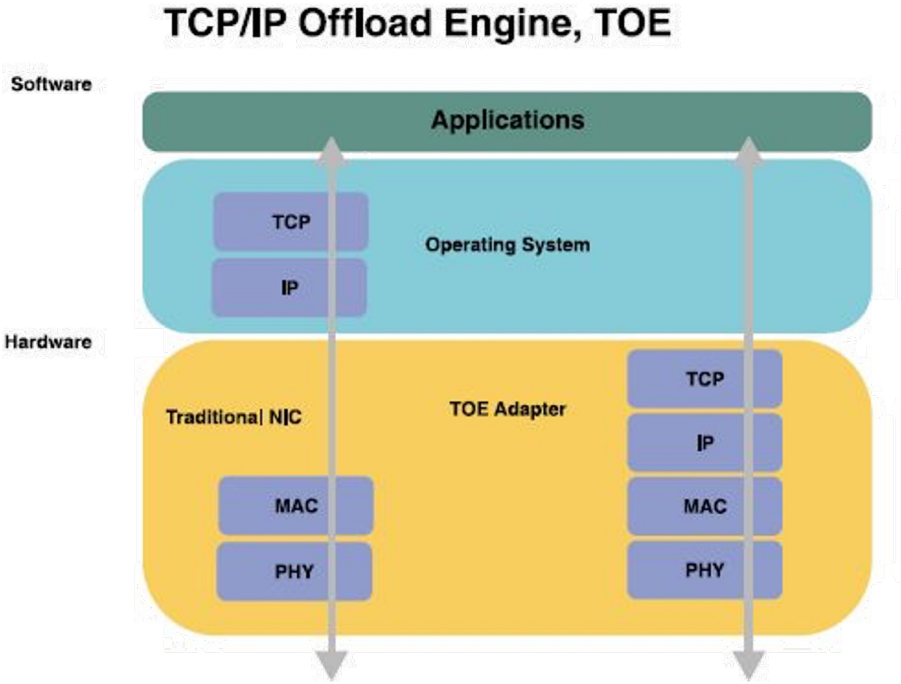
LSO (*Large Receive Offload, Receive Segment Coalescing)*

- Busca reducir carga de trabajo a la CPU en transmisión
- TCP entrega a la NIC paquetes más grandes que la MTU
- La propia NIC hace la segmentación de nivel TCP
- Eso le obliga a crear nuevas cabeceras TCP e IP, descargando a la CPU
- Requiere que la NIC sepa segmentar el protocolo (solo TCP)
- Problemas con encriptación (IPSec)
- Genera ráfagas de tráfico

TOE

- *TCP/IP Offload Engine*
- Los datos pueden pasar directamente de la aplicación a la NIC
- La NIC puede emplearse para todas las tareas de la fase de transferencia y emplear la CPU para el establecimiento y terminación
- O se puede emplear la NIC para todo
- Requiere soporte del sistema operativo
- Puede mejorar el throughput
- Reduce la carga sobre la CPU





upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

Redes de Nueva Generación
Área de Ingeniería Telemática

Virtualización

Application Silos

- En el entorno distribuido una nueva aplicación (software servidor) se desplegaba sobre un hardware independiente
- Una relación 1:1 entre la aplicación y el hardware servidor
- O como mucho 1:N porque tengamos múltiples servidores
- A esto habría que añadirle el almacenamiento
- Y la electrónica de red
- ¿Y si tenemos otra aplicación?
- Cada una con sus servidores y almacenamiento
- El hardware no es reutilizable por otras aplicaciones
- La utilización (CPU) de los servidores es muy baja
- Esto es así para soportar incrementos de carga
- Si no es baja entonces ante un incremento de carga no es rápido provisionar nuevo hardware
- Lo mismo sucede con la utilización de los discos
- Esto se multiplica por el número de aplicaciones
- Pero ocupan todo el tiempo el espacio
- Y están encendidos, consumiendo potencia y necesitando refrigeración
- Esto ha cambiado con la virtualización
- Consolidación

Kernel, Sistema Operativo

- Estamos tratando de un programa (o conjunto de programas)
- Controla el acceso de otros programas a los recursos hardware
- Oculta los detalles del hardware al software, proporcionando independencia de los mismos
- Proporciona planificación de procesos, control de acceso a memoria, acceso a dispositivos, etc

Procesos

- Para que pueda ejecutarse el código de una aplicación de usuario el S.O. debe dejar de usar la CPU para que ejecute ese otro código
- El S.O. necesita mecanismos para recuperar el control de la CPU
- Un proceso es un programa en ejecución
- El Kernel crea una entidad de ejecución que llamamos ”proceso”
- Tiene un hilo de ejecución (thread) así como otras estructuras asociadas en el kernel (fds, memoria, etc)
- Podemos estar ejecutando el mismo programa en varios procesos

¿Virtualización?

Multiproceso

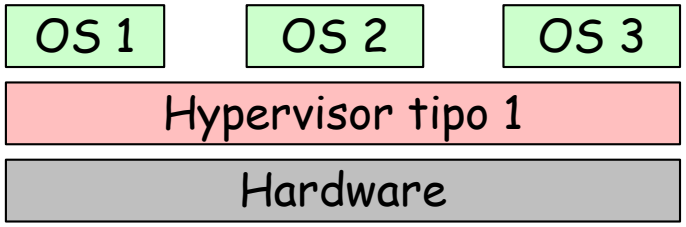
- Cuando varios procesos se ejecutan pero no disponemos de varias CPUs
- Cada proceso cree que dispone de la CPU pero se va alternando
- De nuevo se le está haciendo creer a alguien que dispone de ciertos recursos de forma exclusiva cuando no es así
- El Kernel del S.O. se encarga de gestionar el uso de los recursos físicos

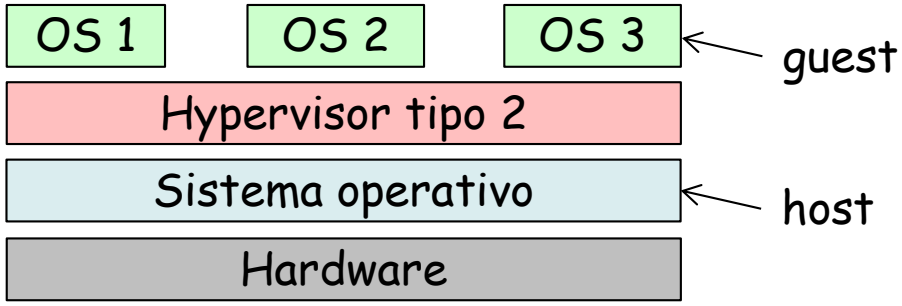
Virtualización de memoria

- El Kernel gestiona también el uso del recurso físico RAM
- Se puede conseguir que un proceso crea que dispone de toda la memoria
- Ve un espacio continuo de direcciones; puede ser más RAM que la existente
- No puede acceder a la memoria en uso por otros procesos
- Cuando la CPU intenta acceder a una dirección de memoria se debe convertir la dirección *virtual* en la dirección física
- Con esa dirección física se puede acceder a la RAM (¿caches?)
- Esto se hace por “páginas” (hoy suelen ser de 4 KiB)
- Esta conversión la hace la MMU (*Memory Management Unit*), parte de la CPU
- El mapeo podría no llevar a memoria RAM sino a datos guardados en disco
- El disco es un dispositivo mucho más lento así que lo normal es mover los datos frecuentemente utilizados a RAM y los poco utilizados a disco

Hypervisor

- Es una capa software entre el hardware y el sistema operativo “guest”
- También llamado “*Virtual Machine Monitor*” (VMM)
- Oculta el hardware real y puede presentar diferente hardware a cada VM
- Esas VMs no necesitan cambios para funcionar en otro hypervisor aunque emplee un hardware diferente siempre que les presente el mismo virtual
- La VM puede ser un solo fichero, sencillo de copiar a otra máquina
- Tipo 1, nativo o “*bare-metal*”
  - Se ejecuta directamente sobre el hardware y lo controla
  - Consume poco espacio y memoria; el mejor rendimiento potencial
  - El hypervisor debe contar con drivers para el hardware
- Tipo 2 o “hosted”
  - El hypervisor corre como una aplicación sobre un sistema operativo convencional
  - El sistema operativo guest sobre el hypervisor
  - El sistema operativo host tiene un impacto en el rendimiento
  - Es más frecuente la existencia de drivers para el hardware





Virtualización

De la CPU

- El kernel de un sistema operativo está pensado para ejecutarse con máximos privilegios
- Ciertas instrucciones de la CPU no son sencillas de virtualizar y no se pueden dejar ejecutar a un proceso
- *Full virtualization*
  - Hace traducción (*on-the-fly*) de instrucciones (*binary translation*)
  - Se sustituyen las instrucciones no virtualizables por otras equivalentes
  - No requiere modificar el OS instalado
- *Paravirtualization (OS assisted virtualization)*
  - Se modifica el sistema operativo guest sustituyendo las instrucciones no virtualizables
  - Requiere menos sobrecarga en ejecución pero hay que poder modificar el código de ese sistema operativo guest
- *Hardware-assisted virtualization*
  - El hardware se encarga de la traducción de instrucciones privilegiadas
  - Requiere soporte por el hardware (Intel VT-x, AMD-V)

De la RAM

- El sistema operativo guest emplea memoria virtual y la mapea a lo que él cree que es memoria física
- Eso no puede ser la auténtica memoria física, así que debe ser de nuevo mapeada
- *Shadow page tables* para hacerlo por soft o *nested paging* (Second Level Address Translation) por hardware si lo soporta la CPU
- Hay que virtualizar la MMU

upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

Redes de Nueva Generación
Área de Ingeniería Telemática

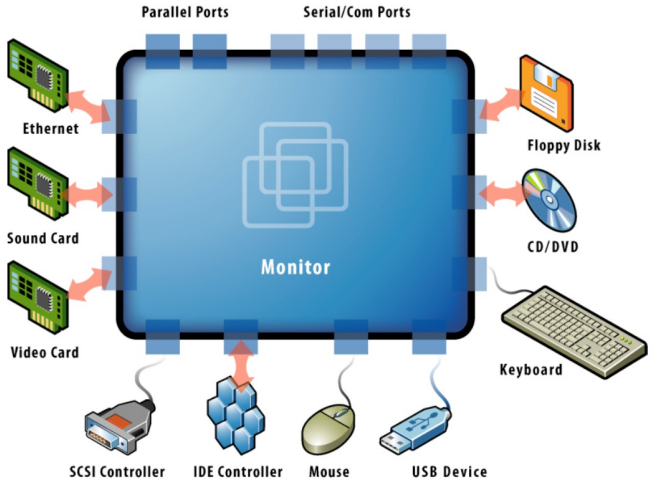
Virtualización

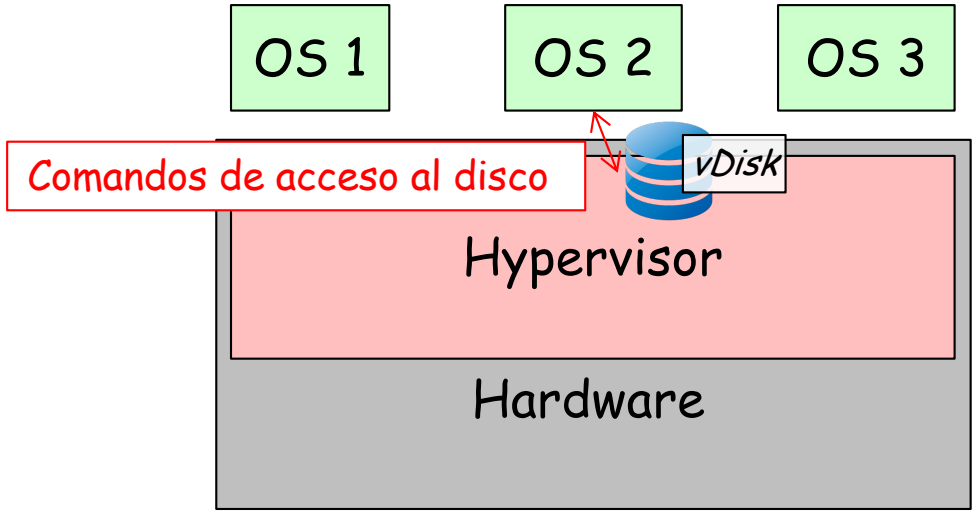
Virtualización de dispositivos

- El VMM presenta a la VM unos dispositivos comunes, de forma que sean fácilmente soportados
- Puede tener varias opciones, por ejemplo ofrecerle al guest diferentes modelos de tarjeta de red
- El hardware puede tener soporte para ser virtualizado

Acceso a disco desde la VM

- El hypervisor ofrece un HD virtual al guest que responde a algún tipo de interfaz de comunicación con discos
- Por ejemplo un interfaz SCSI (más sobre esto más adelante)
- Son mensajes de lectura y escritura en bloques del disco
- De la máquina virtual se reciben los comandos y los bloques a leer se mapean por ejemplo en bloques de un fichero en disco del host

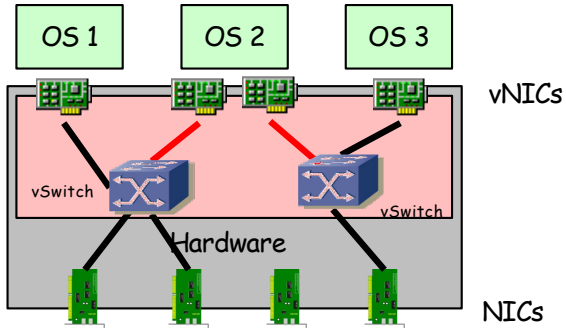




VMs: Networking

vNIC

- Las NICs reales pueden ser de diferentes modelos que las virtuales
- Puede haber una relación 1:1 entre NIC y vNIC
- Puede implementarse un conmutador Ethernet en software
- Se suele llamar un vSwitch o VEB (Virtual Ethernet/Embedded Bridge)
- La dirección MAC de la vNIC suele ser diferente de la MAC de la NIC
- OUI reservado para la empresa desarrolladora del hypervisor
- Puede haber varias vNICs en la misma VM
- Puede haber varios vSwitches

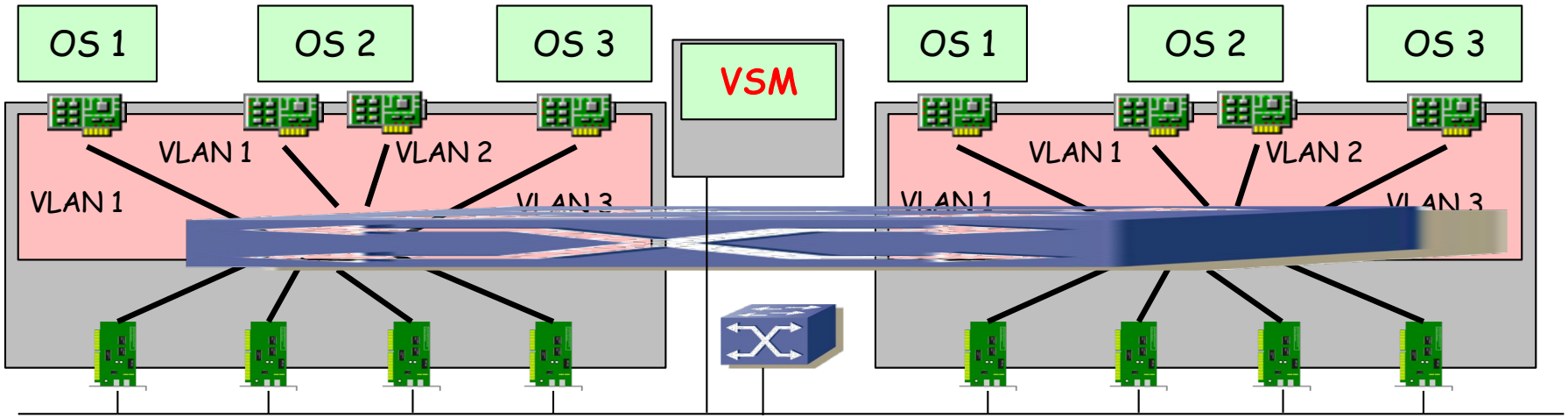


vSwitch

- Se pueden asignar los puertos a VLANs diferentes
- Las NICs soportan 802.1Q y agregación o *NIC teaming*
- El vSwitch tiene más información sobre los hosts que la que puede tener un puente hardware (sabe sus MACs sin usar aprendizaje)
- Puede estar implementado enteramente en software o parte en hardware (normalmente funcionalidades en la NIC)
- Puede estar desarrollado junto con el hypervisor o por otra empresa y así gestionarse como parte del entorno de virtualización o de red
- No reenvía entre los puertos hacia la infraestructura de red, no participa en el STP
- No necesita hacer aprendizaje, solo tiene las MACs de las VMs estáticamente
- Pero hay que configurar políticas en sus puertos lógicos
- Probablemente no tenga las funcionalidades de un switch físico (QoS, ACLs, etc)

vSwitch modular

- Este virtual switch puede estar compuesto, igual que uno hardware de:
  - Módulo controlador/supervisor virtual (plano de control)
  - Módulos con los puertos Ethernet virtuales (plano de datos)
- En ese caso, el elemento en cada host es el módulo de puertos
- El supervisor corre como una máquina virtual (Ej: Cisco 1000v)
- Vale con un supervisor para controlar varios hosts y entonces es como si todos formaran un switch virtual
- Ese supervisor podría correr en su propio hardware (Ej: Cisco 1100)
- Cada host mantiene su propia tabla de reenvío
- El switch de un host no tiene conocimiento de las MACs aprendidas en otro, ni aunque formen parte del mismo switch distribuido
- Es decir, aunque hablemos de un switch distribuido NO hay una base de datos de filtrado única
- Eso quiere decir que una dirección MAC puede aparecer más de una vez, dado que puede aparecer en todas las tablas de host

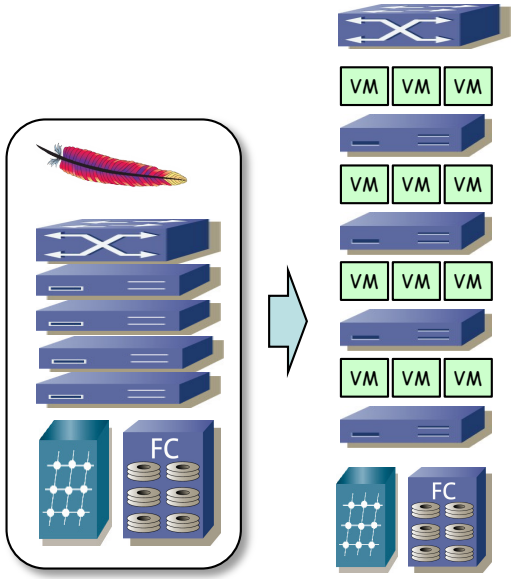


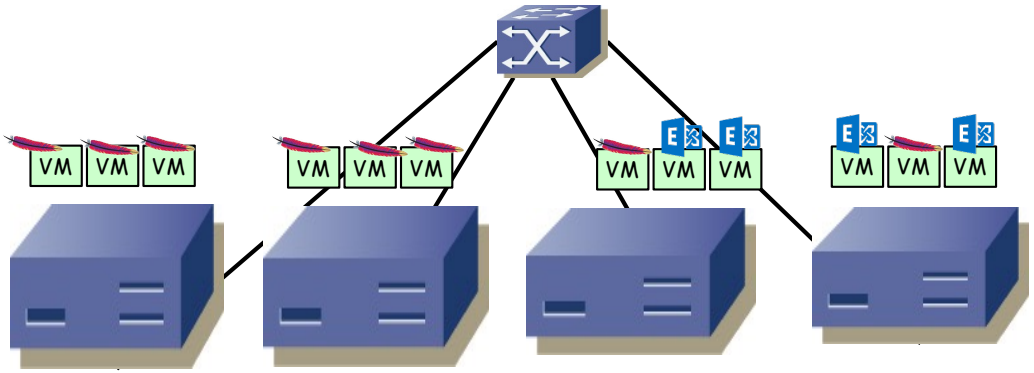
upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

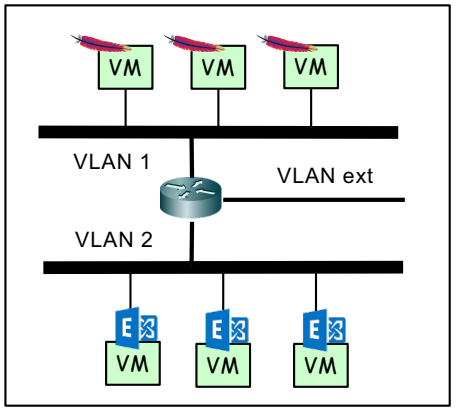
Redes de Nueva Generación
Área de Ingeniería Telemática

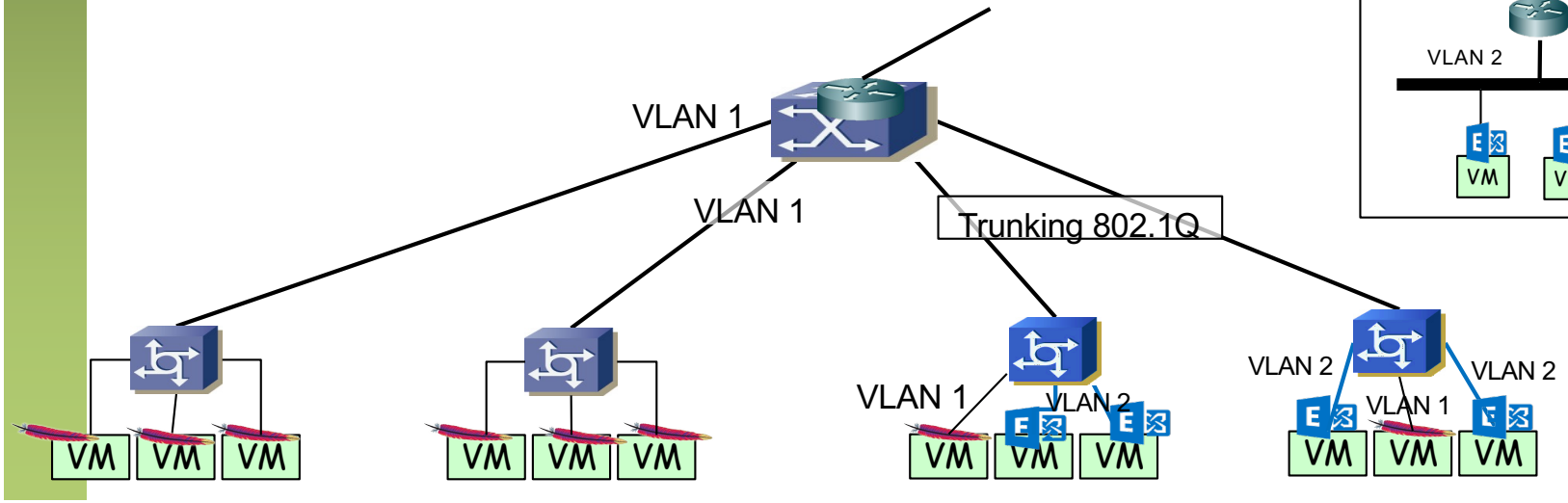
Virtualización

Evolución del App. Silo









Beneficios de la virtualización

Ventajas

- Consolidación
  - Ahorro en hardware para correr los servicios (y espacio)
  - Ahorro en consumo eléctrico; ahorro en refrigeración
- Sencilla separación de entornos de desarrollo, pruebas y producción
- Sencilla creación, backup y replicación
- Instalaciones menos atadas al hardware (drivers)
- Permite mantener software (S.O.) antiguos sobre hardware moderno
- Esas VMs pueden ser
  - Firewalls; antivirus en red; inspectores de contenido; balanceadores y publicadores
  - Caches; puentes (sí, en vez de un vSwitch en el hypervisor estaría como una VM)
  - Cualquier cosa que en el fondo sea software en un sistema operativo
  - Por otro lado la funcionalidad de router podría ser llevada a cabo por el Kernel de un host en lugar de por una VM

Inconvenientes

- Pérdida de rendimiento en apps que hacen un uso intensivo del hardware
- Hardware especializado para el que no exista drivers en el hypervisor
- Un fallo hardware tiene efecto en múltiples VMs
- Depuración del sistema global más compleja, mayor acomplamiento
- Nuevas herramientas de gestión, nuevas habilidades requeridas al personal IT

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

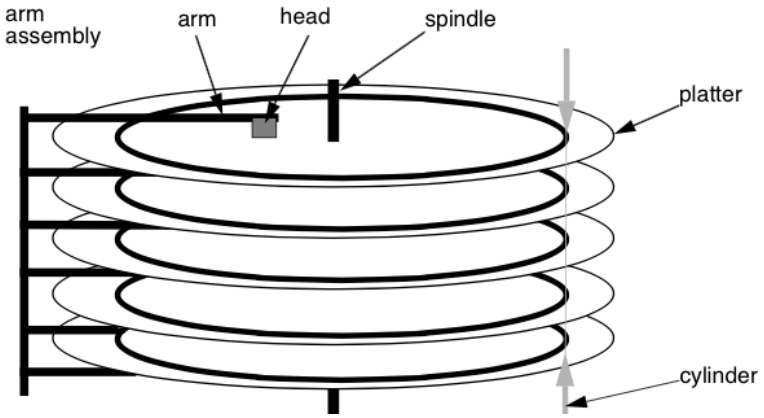
Almacenamiento

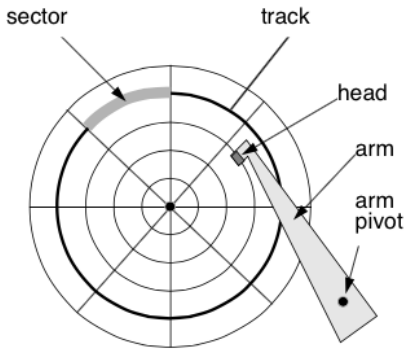
Sistemas de almacenamiento

- Almacenamiento primario
  - Memoria RAM; volátil
  - Accesible directamente por la CPU
  - Acceso aleatorio
  - Pequeña capacidad y bajos tiempos de acceso
- Almacenamiento secundario
  - No volátil
  - No es accesible directamente por la CPU
  - Requiere dispositivos de entrada salida (I/O)
  - Mayor capacidad y mayores tiempos de acceso
  - Acceso aleatorio
  - Discos duros
- Almacenamiento terciario
  - Sistemas de almacenamiento removibles
  - Tiempos de acceso aún mayores pero coste por GB menor
  - Empleados para almacenamiento "*long term*"
  - Cintas (desde los 50s) de acceso secuencial
  - Pueden almacenar centenares de petabytes (y hasta exabytes)

Arquitectura del disco

- Platos (*platters*): material magnético. Pistas (*tracks*): trayectoria circular
- Sectores: unidad mínima direccionable, todos del mismo tamaño, tradicionalmente 512 Bytes (hay ahora unidades con 4 KiB). Cilindros: pila vertical de pistas
- El tamaño del disco (en pulgadas) condiciona su capacidad y consumo
- Brazo y cabeza de lectura/escritura (por cada plato)
- Eje de rotación (*spindle*). Todos los platos rotan al unísono
- Típicas velocidades de rotación: 7.200 rpm, 10.000 rpm, 15.000 rpm
- La cabeza debe avanzar hasta la pista donde están los datos
- Debe esperar a que el plato rote hasta que el sector que busca se encuentre debajo
- Entonces podrá leer o escribir (no a la vez)
- Lee del cilindro, así que cuando termina la pista pasa a la del mismo cilindro en otro disco
- Esas 4 operaciones llevan tiempo (posicionarse, esperar a que gire, leer/escribir y opcionalmente cambiar de plato o de pista)





(rpm)	Avg. rot. (ms)
5400	5.5
7200	4.2
10000	3
15000	2

Tiempos básicos

- "*Seek time*": para colocar la cabeza lectora en la pista deseada
- "*Rotational latency*": espera a que el sector deseado alcance la cabeza
- "*Transfer time*": Tiempo para transferir los datos del/al sector
- "*Bus transfer time*": el protocolo del bus (SCSI, SATA) añade mensaje

Discos internos

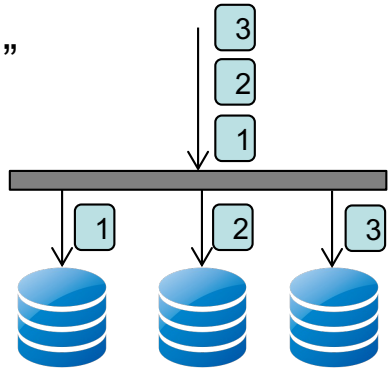
- En caso de fallos hardware complican y enlentecen la reparación
- En el entorno servidor lo más común es que sean "cambiables en caliente"
- En el caso del servidor suelen mantener el sistema operativo y caches
- Los datos estarán en discos externos

Cabinas de discos

- Las cabinas de discos (*disk array*) pueden incluir una *controladora*
- Puede estar integrada con la cabina o con el servidor
- La cabina controla los discos y ofrece algún interfaz de acceso para el servidor (o los servidores)
- Los servidores podrán acceder a volúmenes lógicos creados en esos discos
- La controladora contará con una cache (RAM o flash)
- Puede contar con fuentes de alimentación redundantes

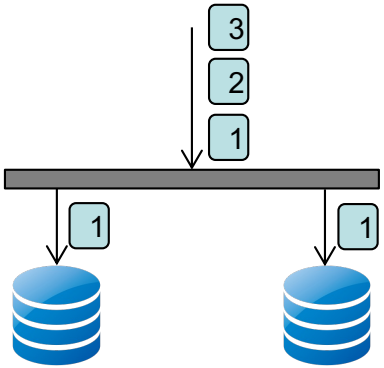
RAID

- *Redundant Array of Independent (Inexpensive) Disks*
- Varios discos que de cara al usuario (el servidor) se comportan como un solo volumen
- Los diferentes tipos de RAID se denominan mediante un “nivel”
- RAID level 0, RAID level 1, etc
- En comparación con *Just a Bunch Of Disks* (JBOD)



RAID 0 (*Disk stripping)*

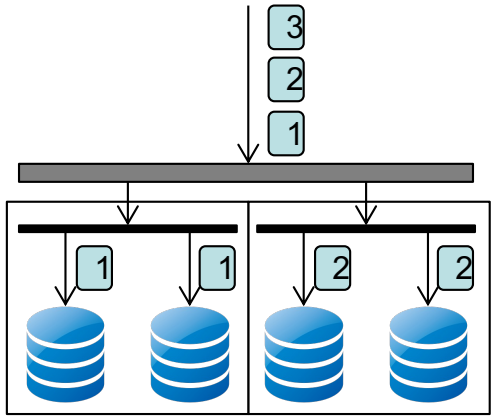
- Se reparten los datos entre varios discos
- Esto permite mayores velocidades de transferencia
- No hay redundancia. Un fallo en un disco es irrecuperable
- Cualquier número de discos (mayor que 1)



RAID 1 (*Mirroring)*

- Los datos se replican. Requiere al menos 2 discos
- No mejora la velocidad pero sí da protección
- Ante un fallo en un disco el RAID puede seguir funcionando
- Se puede sustituir el disco defectuoso y la controladora reconstruye el *mirror*
- La reconstrucción reduce el rendimiento del disco

RAID levels



RAID 1+0 (RAID 10)

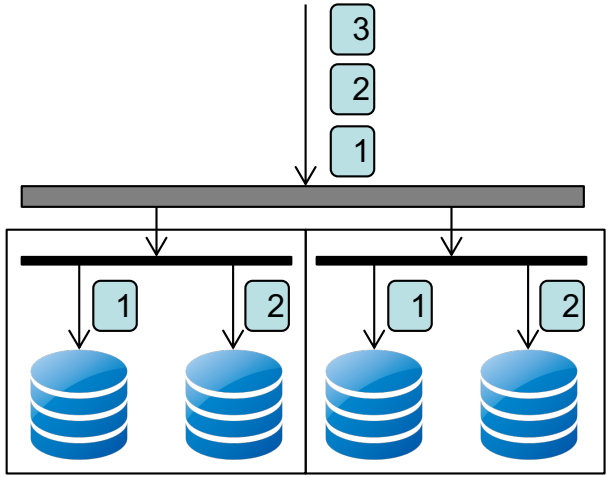
- Combina *mirror* y *stripe*
- También se habla de RAID 0+1
- Requiere al menos 4 discos
- Soporta un fallo doble según qué discos fallen

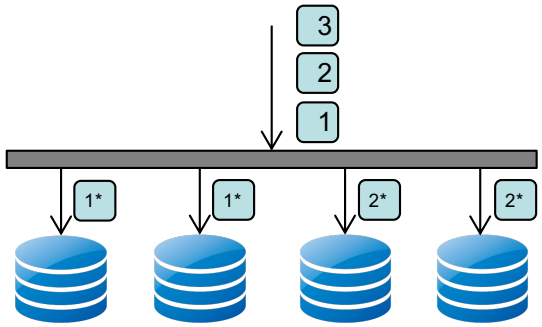
RAID 5

- Se hace *striping* a nivel de bloques
- Se guarda paridad pero no está en el mismo disco la paridad de todos los bloques sino que se reparte por los discos
- Mejora la velocidad porque los datos y la paridad están repartidos
- Menor probabilidad de coincidir múltiples operaciones en el mismo disco cuantos más discos (y así mayor velocidad)

RAID 6

- Requiere al menos 4 discos. Sobrevive a fallos dobles
- Se calcula doble paridad, distribuida por los discos





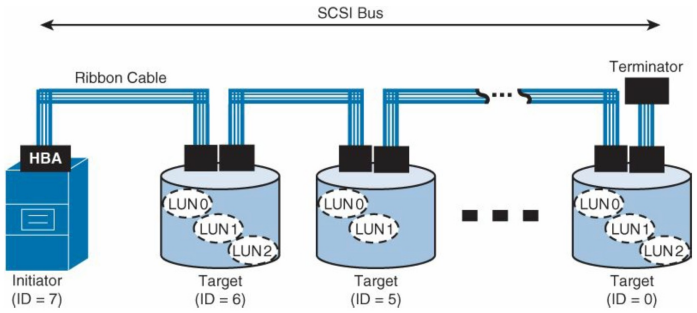
Interfaces

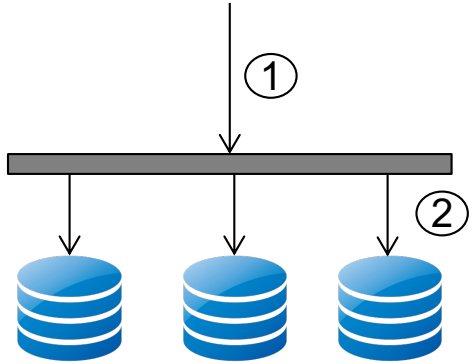
Interfaces

- El primero de ellos (*front-end*) es desde el host a la controladora
- El segundo (*back-end*) es desde ésta a los discos
- Cuando el disco es interno, simplemente no existe el *front-end*
- El acceso desde el host puede ser a bloques, a ficheros o a registros

SCSI (Small Computer System Interface)

- Desarrollado por el *International Committee for Information Technology Standards (INCITS)* en los 80s
- Define cómo transferir datos entre ordenadores y periféricos
- Se transfiere a nivel de bloque
- Eso implica comandos, protocolos e interfaces físicos
- Los periféricos más habituales son discos duros pero también ha habido impresoras, scanners, etc.



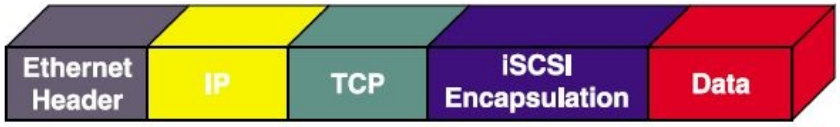


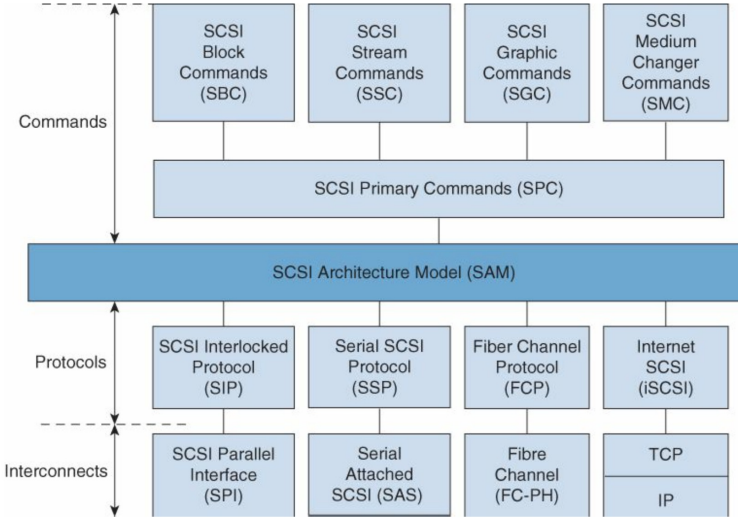
SCSI bus

- El medio físico es un bus paralelo. Requiere un terminador
- Dispositivos encadenados. La comunicación es half-duplex
- Un elemento es el "*Initiator*", normalmente la controladora en el ordenador que accede a los periféricos
- Cada dispositivo (*target*) tiene un identificador numérico que implica también la prioridad del mismo
- El iniciador direcciona unidades lógicas ("*logical units"*)
- Cada una de las cuales se identifica con un *Logical Unit Number* (*LUN*)
- Discos duros pueden tener más de un LUN
- La controladora SCSI es lo que se llama un *Host Bus Adapteer (HBA)*
- Los comandos principales en el bus son simplemente "Read" y "Write" aunque hay otros para diagnóstico, formateo, etc.
- SCSI-1 o el SCSI original
  - 5 Mbytes/s. Conector Centronics, 50 pines
  - Hasta 8 dispositivos en el bus (numerados de 0 a 7) incluyendo el iniciador
- Versiones posteriores: SCSI-2, SCSI-3, Fast SCSI, Wide SCSI, Ultra SCSI, Ultra Wide SCSI, Ultra 160 SCSI, Ultra 320 SCSI…
- Aumentan la anchura del bus, la velocidad, el número de dispositivos
- Versiones hasta 640Mbps, 16 dispositivos, 25 metros
- Se encuentra con limitaciones debidas al cable paralelo

SCSI hoy en día

- Se ha independizado el protocolo (los comandos) del interfaz físico
- Ha evolucionado hacia nuevos medios físicos
- SAS = *Serial Attached SCSI*
  - Se abandona el cable paralelo por un medio serie
  - Se abandona el bus por enlace punto-a-punto
  - Hoy en día tasas de hasta 12 Gbps y hasta 10 m
- *Fibre Channel*
  - Normalmente sobre fibra (no necesariamente)
  - Una tecnología de red principalmente para almacenamiento
  - Que transporta comandos SCSI
  - Velocidades hoy en día de decenas de Gbps, distancias de kms
  - Transportable sobre WAN
- iSCSI (*Internet SCSI*)
  - Comandos SCSI sobre una conexión TCP





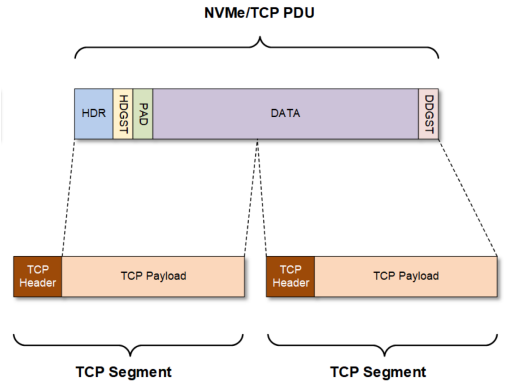
Discos de estado sólido

SSD (Solid State Disk)

- En realidad no es un “disco” sino memoria, pero la utilizamos como almacenamiento permanente (NAND Flash)
- No hay posicionamiento de cabeza lectora ni rotación del disco
- Estamos hablando de tiempos de acceso en el orden de los microsegundos
- Hasta hace unos pocos años no han tenido un precio que fuera compatible con sus ventajas
- Envejecimiento y limitaciones de escritura/borrado

NVMe (Non-Volatile Memory express)

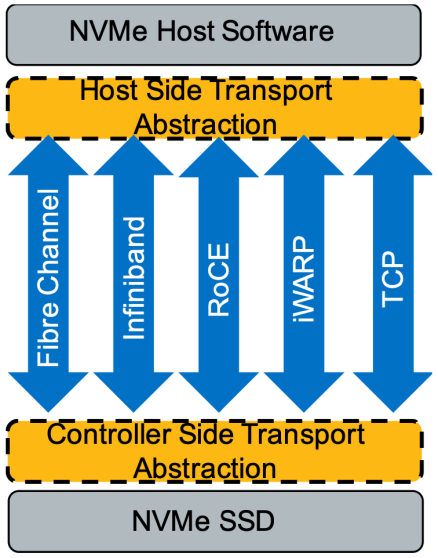
- Protocolos de transporte y acceso a almacenamiento para SSD
- Conexión directa a CPU a través de PCIe
- Aumenta el paralelismo de peticiones a disco (64K colas de 64K peticiones c.u.)
- Menor latencia
- NVMe-oF : NVMe over Fabrics
  - Uso de DMA (zero-copy)
  - NVMe sobre RDMA (Infiniband, RoCE, iWARP)
  - NVMe/FC : comandos NVMe sobre FC en lugar de SCSI
  - NVMe/TCP



NVMe

https://nvmexpress.org/

- Non-Volatile Memory express
- Protocolos de transporte y acceso a almacenamiento para SSD
- Conexión directa a CPU a través de PCIe
- Aumenta el paralelismo en las peticiones al disco (64K colas de 64K peticiones cada una)
- Menor latencia
- NVMe-oF : NVMe over Fabrics
  - Uso de DMA (zero-copy)
  - NVMe sobre RDMA (Infiniband, RoCE, iWARP)
  - NVMe/FC : comandos NVMe sobre FC en lugar de SCSI
  - NVMe/TCP



https://nvmexpress.org/wp-content/uploads/NVMe_over_Fabrics_Sept_2017_Brandon_Hoff.pdf

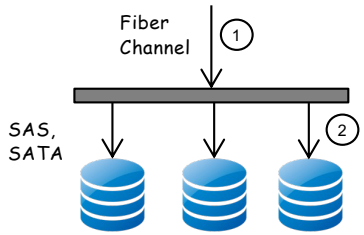
upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

Redes de Nueva Generación
Área de Ingeniería Telemática

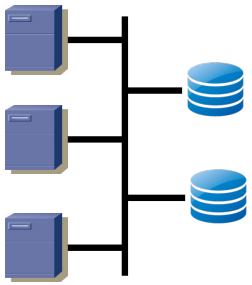
SAN

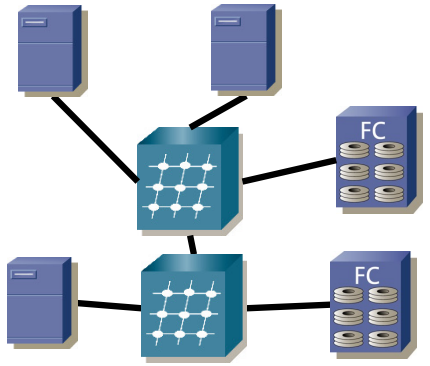
Acceso al almacenamiento

- *Direct Attached Storage (DAS)*
  - Cada servidor necesita su sistema de almacenamiento (ej: backups)
  - Un backup nocturno significa que el resto del día esos discos están inutilizados
- O puede ser a través de una red
  - Estamos hablando del *front-end* de acceso a los discos
  - El *back-end* es común que sean discos SAS o SATA



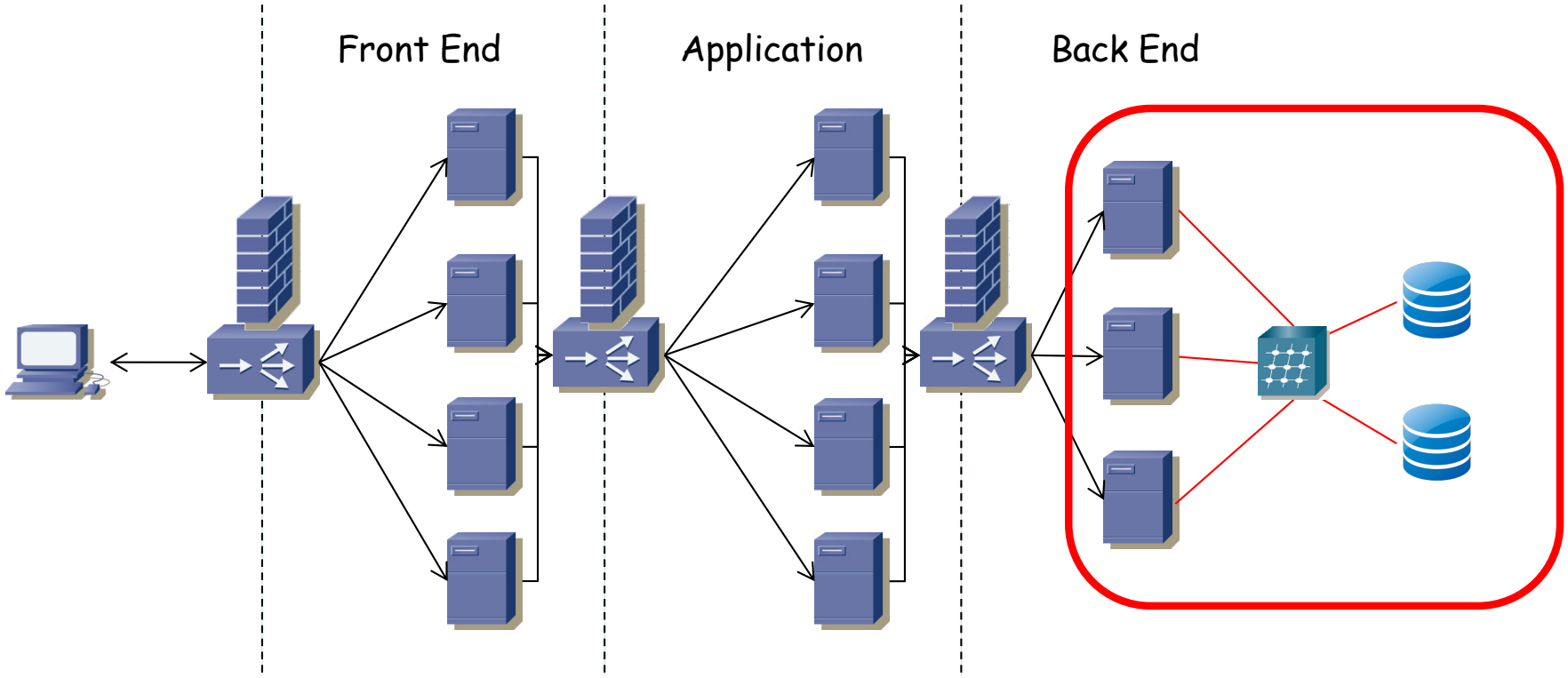
- **SAN**
  - Estamos hablando de una tecnología de red
  - Cuenta con sus propios conmutadores
  - Se dice que forman un *fabric*
  - Cuenta con su propia pila de protocolos
  - El interfaz en el host es el *Host Bus Adapter (HBA)* que sería el equivalente a la NIC en una LAN



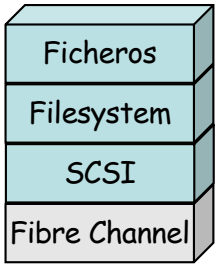


Es decir…

- Y recordemos que lo estamos poniendo en el backend del servicio porque es donde suelen estar los datos
- Pero nada impide que los servidores de cualquier otra capa…



Fibre Channel



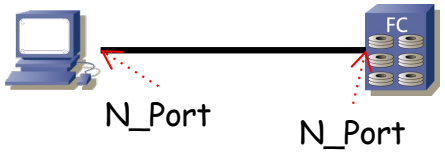
- Desarrollo comenzado a finales de los 80s
- Lo normal es que sea sobre fibra pero puede ser sobre cobre
- Soluciona el transporte pero no fija lo que transporta
- Conmutación de paquetes
- Así, puede transportar comandos SCSI pero también IP o ATM
- Inicialmente una de sus ventajas era su alta velocidad en comparación con las tecnologías LAN de la época
- Para los nodos el comité T11 del INCITS tiene estandarizado: 1GFC, 2GFC, 4GFC, 8GFC, 16 GFC, 32GFC y 128GFCp (128Gb/s mediante 4 canales)
- Para la conexión entre conmutadores tiene 10GFC, 20GFC, 40GFCoE (FC over Ethernet), 100GFCoE y 128GFCp

Layers

- FC-0: Capa física
- FC-1: Codificación, sincronización, control de errores
- FC-2: Formato de trama, señalización para gestión
- FC-3: Ofrece un conjunto único de servicios aunque por debajo haya varios puertos físicos (*name, login, address manager, alias server, fabric controller, management, key distribution, time*)
- FC-4: Capa de adaptación para protocolos superiores como puede ser SCSI, NMVe o IP (ULP = Upper Level Protocol)

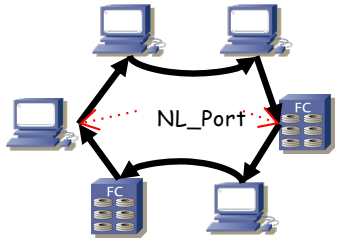
Topologías

Point-to-Point



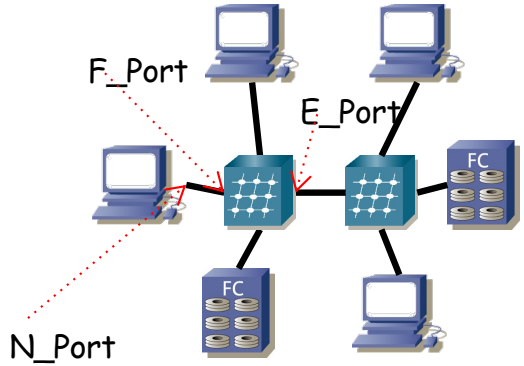
- No hace falta un conmutador o crear “una red”
- Igual que en una Ethernet, podríamos tener simplemente un enlace punto-a-punto
- Ganamos algunas de sus características técnicas pero desde luego no la de compartir el uso de los discos
- Es una conexión directa entre los puertos de 2 nodos, que se vienen a llamar “N_Ports”

Arbitrated loop (FC-AL)



- Anillo compartido
- 2-126 dispositivos pero muchos menos por rendimiento
- Los puertos se llaman “L_Ports” (NL_Port o FL_Port)
- Se negocia cuál de los nodos actúa como *master*
- Un nodo establece un circuito entre dos puertos que monopoliza el anillo
- No puede hacerlo de nuevo hasta haber dado turno a todos los demás

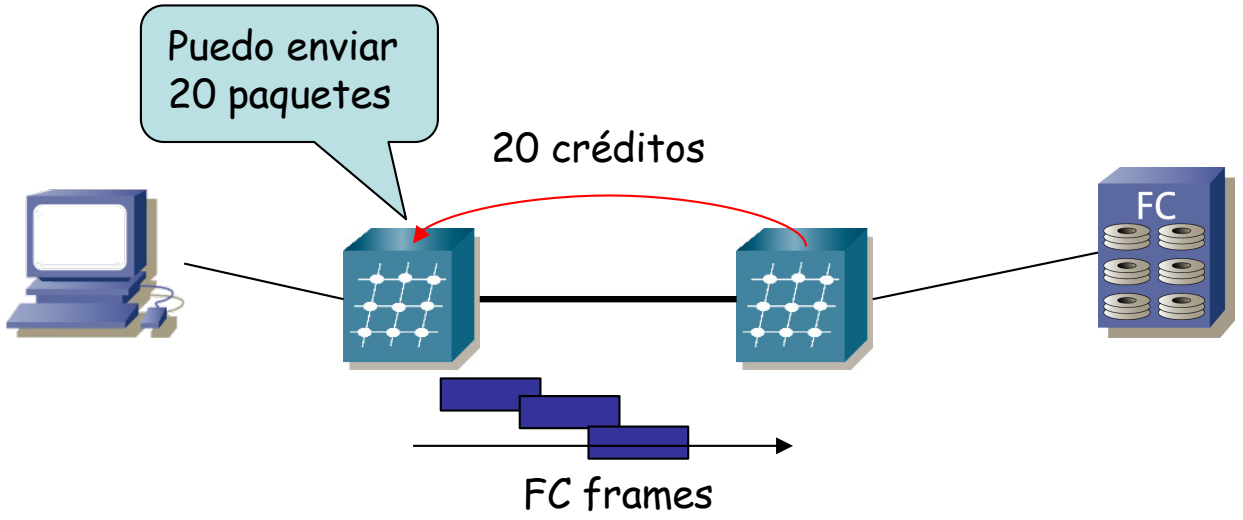
Crosspoint switched (fabric)



- Uno o más conmutadores, interconectando múltiples nodos
- El direccionamiento permite en teoría hasta 2^{24} nodos
- “F_Port” (Fabric Port) a los nodos, “E_Port” (Expansion) entre switches
- ISL = *Inter-Switch Link*
- “FL_Port” para conectar a un FC-AL
- “G_Port” puerto genérico que se comporta según lo que se le conecte
- Topologías edge-core similares a acceso-agregación

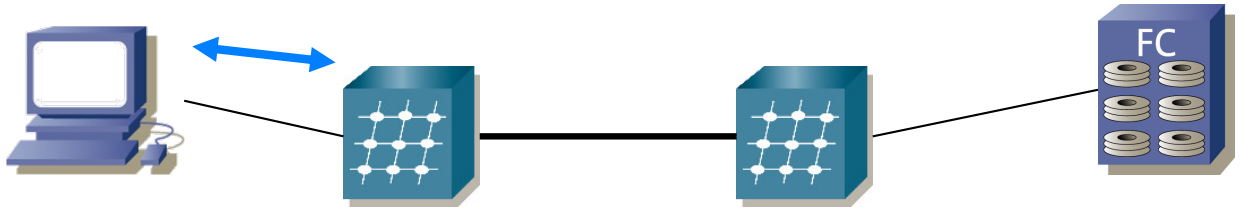
FC-2: Service Classes

- Diferentes “clases de servicio” (Class 1, Class 2, etc), con circuitos virtuales o datagramas
- Con ACKs y NACKs o sin ellos; con garantía de orden o no; con control de flujo en cada salto y/o e2e
- Lo habitual es clase 3 para almacenamiento: Sin conexión, no garantía entrega en orden, best effort, sin ACKs ni NACKs, flow control salto a salto
- Se busca un escenario sin pérdidas
- No se envía un paquete si receptor no tiene espacio para almacenarlo (ventana deslizante)
- Con flow control salto a salto el receptor es el otro extremo del enlace
- Receptor ofrece una cantidad de ”créditos” (*buffer-to-buffer credits*) (=ventana)
- Hasta que reciba un nuevo anuncio de créditos (*R_RDY*) (porque el siguiente salto haya vaciado buffers)
- Problemas de *slow drain*



Login

- Cada dispositivo tiene el equivalente a las direcciones MAC de Ethernet:
  - WWNN: World Wide Node Name (identifica al dispositivo)
  - WWPN: World Wide Port Name (identifica al puerto)
- Cada switch tiene un 'Domain ID'
- Conectar un N-Port a F-Port no garantiza comunicación
- El nodo debe hacer login en el *fabric* (en el switch)



SAN ≠ NAS

SAN ≠ NAS

En una SAN

- Se accede de forma serie a bloques de disco
- Normalmente mediante comandos SCSI-3
- Los protocolos están optimizados para baja latencia y nulas pérdidas
- La solución de transporte más habitual es Fibre Channel
- Acceso de varios servidores al mismo volumen requiere sistemas de ficheros especiales

En una solución NAS

- Se accede a ficheros (no a bloque)
- Se suele transportar sobre una tecnología LAN (o LAN + IP)
- Los protocolos no garantizan baja latencia ni nulas pérdidas (su recuperación aumenta la latencia)
- NFS, SMB/CIFS, AFP, etc

Rendimiento haciendo backups

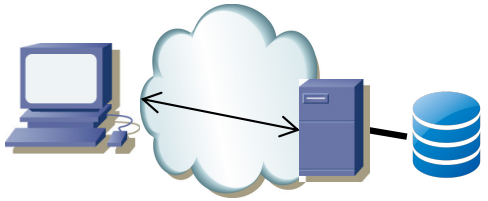
- Mayor en SANs, generalmente más rápida
- En algunos casos se pueden mover datos del disco de un servidor a una cabina sin intervención de la CPU
- En sistemas de ficheros con gran cantidad de ellos es más eficiente una copia del dispositivo *raw*

Bases de datos

- Algunas requieren un acceso al disco a nivel de bloques por rendimiento e integridad

Acceso a ficheros

- Solución cliente-servidor, hoy en día sobre TCP/IP
- Integrado con el sistema operativo en el cliente
- Permite que las apps vean el sistema de ficheros remoto como si fuera local
- Ejemplos de protocolos: SMB, NFS, AFP

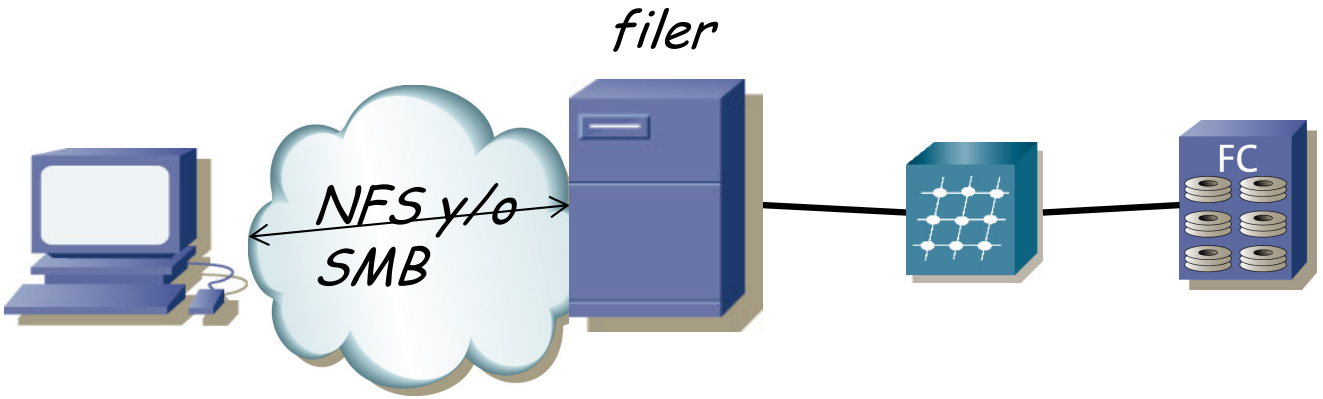


Ejemplos:

- **SMB (*Server Message Block)***
  - Desarrollado por IBM y Microsoft, principalmente en sistemas Windows
  - Empleado para resolución de nombre, navegar recursos compartidos, compartición de ficheros, acceso a impresoras, autentificación, etc
  - Sobre TCP, UDP y otros
- **NFS *(Network File System)***
  - Principalmente en sistemas UNIX (desarrollado por Sun Microsystems)
  - Emplea los protocolos XDR (External Data Representation, RFC 1832) y RPC (Remote Procedure Call, RFC 1831)
  - NFSv2 (RFC 1094) sin estado (UDP), tamaños de 32 bits
  - NFSv3 (RFC 1813) sin estado (UDP aunque también TCP), tamaños de 64 bits
  - NFSv4 (RFC 3530) con estado (TCP), seguridad
  - NFSv4.1 (RFC 5661) introduce *parallel NFS* (pNFS)
- Otras alternativas menos frecuentemente integradas en el cliente: FTP, HTTP (WebDav), SSH/SFTP

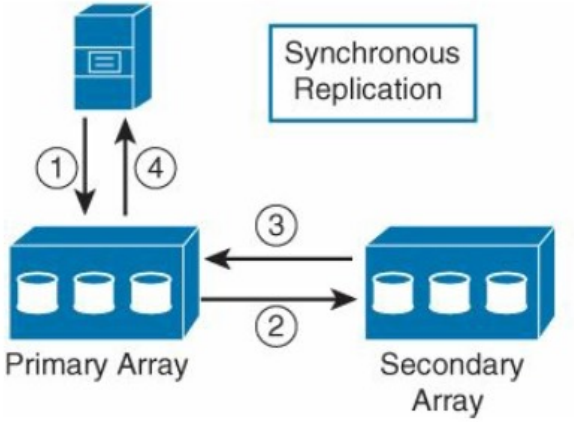
NAS

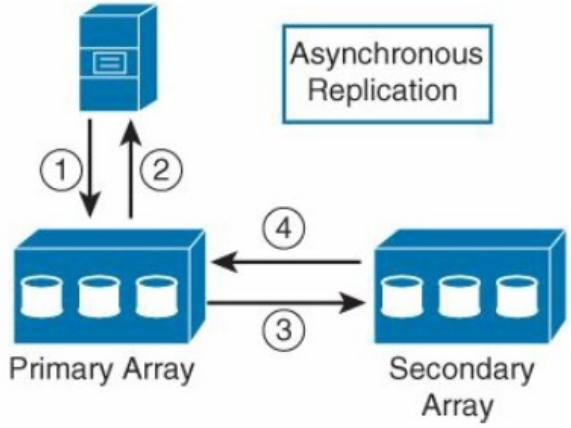
- NAS = *Network Attached Storage*
- Inicialmente el servidor es un hardware de propósito general
- Hoy en día hardware dedicado (*appliances*) para esta tarea
- Optimizado para ello
  - Simplifica la gestión
  - RAIDs, discos *hot-swappable*
  - Capaces de compartir los mismos recursos mediante varios protocolos simultáneamente (SMB + NFS)
  - Mejoras de rendimiento en la implementación del soft servidor
- El disco en vez de ser local puede estar en una SAN
- SMB Direct (empleando RDMA), NFS over RDMA



Disk Array Virtualization

- *Partitioning*
  - Una cabina puede soportar subdividirse en dispositivos lógicos
  - Cada uno tendría asignados recursos de: discos, cache, memoria, puertos
  - Cada partición puede crear sus propias LUNs
- *Array-based data replication*
  - Múltiples cabinas pueden trabajar juntas en replicación
  - La *replicación síncrona* se basa en devolver confirmación de haber almacenado el dato cuando se ha escrito en las dos
  - La *replicación asíncrona* se basa en copiar después o periódicamente los datos (no bloquea la respuesta al usuario)

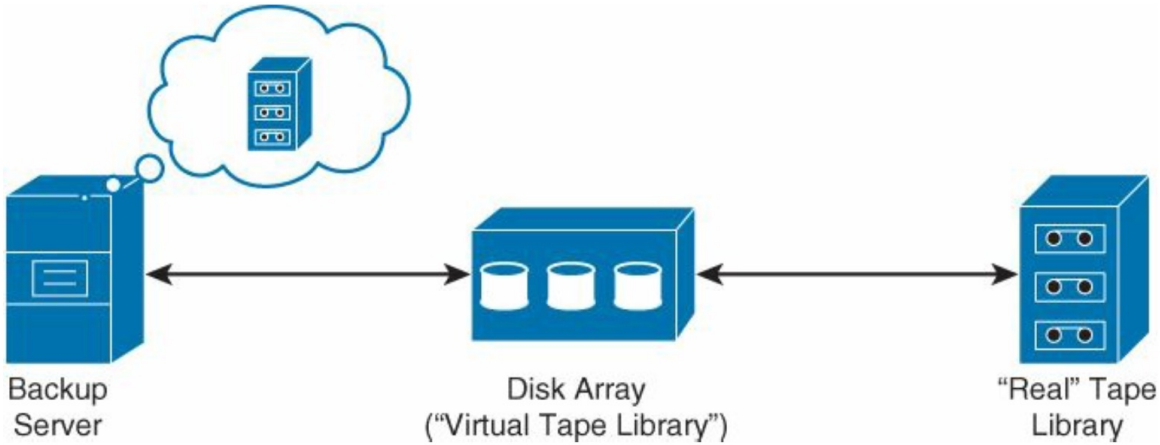




G.A.A.Santana,"Data Center Virtualization Fundamentals", Cisco Press

Virtual Tape Library

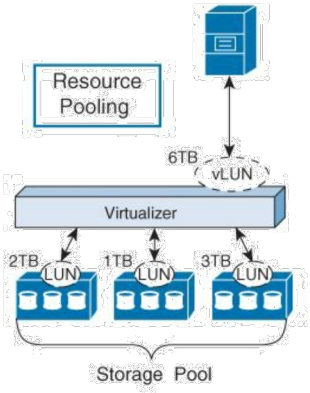
- El servidor accede a la cabina como si fuera la biblioteca de cintas
- La cabina actúa como una cache
- *Deduplication*
  - No manda al almacenamiento una segunda vez algo que ya existe
  - Apunta simplemente una referencia
  - Si luego uno de los dos se modifica puede guardar solo las modificaciones
  - También en el escenario de cabina de discos independiente
  - Ahorra por ejemplo bastante con imágenes de OS
  - También puede hacer compresión

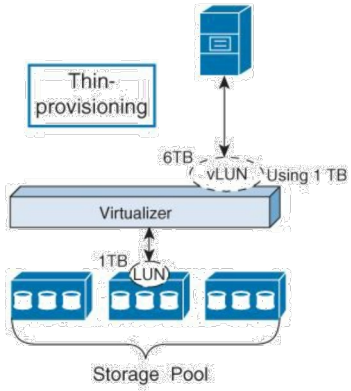


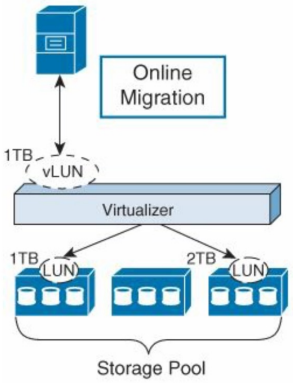
G.A.A.Santana,"Data Center Virtualization Fundamentals", Cisco Press

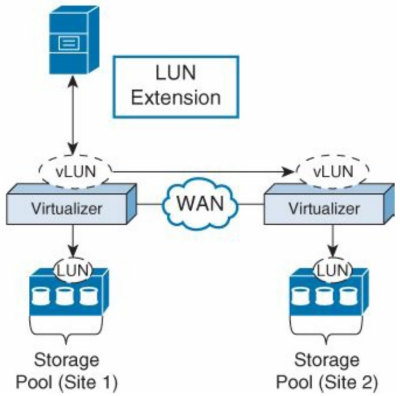
Virtualización de LUNs

- Un virtualizador se interpone entre el servidor y la LUN
- Ofrece una vLUN al servidor
- Eso le permite modificar cómo la implementa sin alterar al servidor
- Puede agregar varias LUNs en una (*storage resource pooling*)
- Puede ofrecer una vLUN de mayor capacidad que la que realmente está empleando (*thin-provisioning*)
- Esto puede llevar a *over-subscription* y como tal funciona mientras todas las vLUNs no quieran usar toda la capacidad que anuncian
- Permite la migración de la LUN de una cabina a otra de forma transparente (*online migration*)
- Por ejemplo para cambiar a discos o un RAID más rápido
- El virtualizador puede dar la funcionalidad para la replicación entre dos cabinas, por ejemplo en DCs alejados
- Un virtualizador en cada DC puede estar ofreciendo la vLUN a los servidores de ese DC (*LUN extension*)





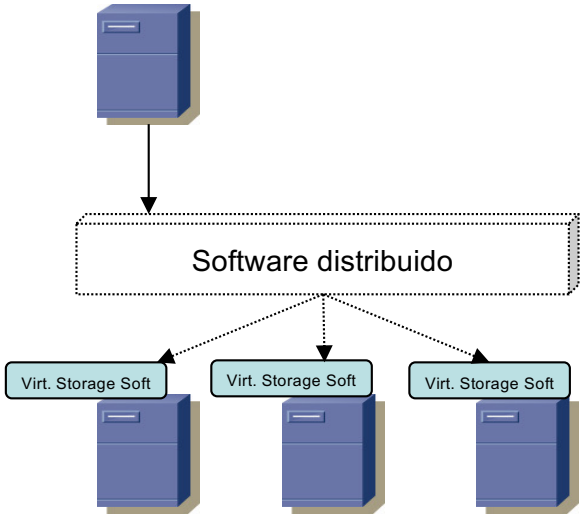




G.A.A.Santana,"Data Center Virtualization Fundamentals", Cisco Press

Evolución

- Hemos supuesto que el almacenamiento son sistemas dedicados a ello (cabinas, disk arrays)
- El almacenamiento puede estar en hosts convencionales (magnéticos o SSD)
- El virtualizador en puede ser software distribuido corriendo en esos hosts
- El "usuario" es probablemente una VM que puede estar ejecutándose en uno cualquiera de esos hosts
- *Software defined storage*
- Puede dar acceso a ficheros, objetos o bloques
- Un mismo volumen pude estar repartido entre varios hosts
- Puede haber replicación de datos para protección
- Puede seleccionar el tipo de almacenamiento más rápido para los datos más accedidos



FC-3

Login

- Cada dispositivo tiene el equivalente a las direcciones MAC de Ethernet:
  - WWNN: World Wide Node Name (identifica al dispositivo)
  - WWPN: World Wide Port Name (identifica al puerto)
- Cada switch tiene un 'Domain ID'
- Conectar un N-Port a F-Port no garantiza comunicación, debe "hacer login"
- FLOGI: Fabric Login
  - Es un intercambio de paquetes: FLOGI (Fabric Login)
  - Cada nodo obtiene del fabric un FCID (con el Domain ID del switch)
  - Establece créditos
  - El fabric enruta el tráfico hacia el FCID
- PLOGI: Port Login
  - Login en el target
  - Permite establecer los créditos para flow control si no hay *fabric*
  - Name server en el switch registra el mapeo entre WWPN y FCID

Multipath

- Permite caminos redundantes (S.O. host debe soportar ver los dos como uno)
- Varios HBAs en host y varios interfaces en sistema de almacenamiento
- FC dispone del protocolo FSPF (*Fibre Channel Shortest Path First)*
- Caminos alternativos para reparto de carga o redundancia

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

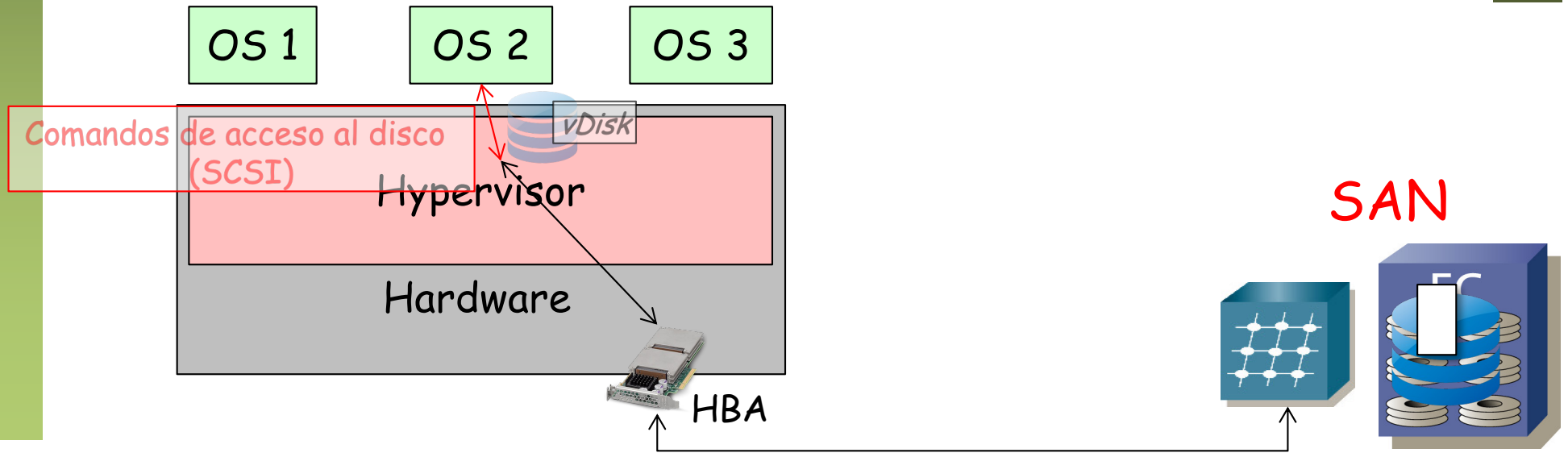
Gestión y *provisioning* de máquinas virtuales

Gestión

- El hypervisor y sus VMs se pueden gestionar remotamente
- Tareas como crear una VM, arrancarla, detenerla, clonarla, hacer un backup, migrarla, etc
- Virtualization Infrastructure Management (VIM)
- Software que corre en un controlador, normalmente un ordenador independiente
- Puede que el host tenga alguna NIC dedicada a la gestión
- Se pueden crear VMs a partir de *templates*

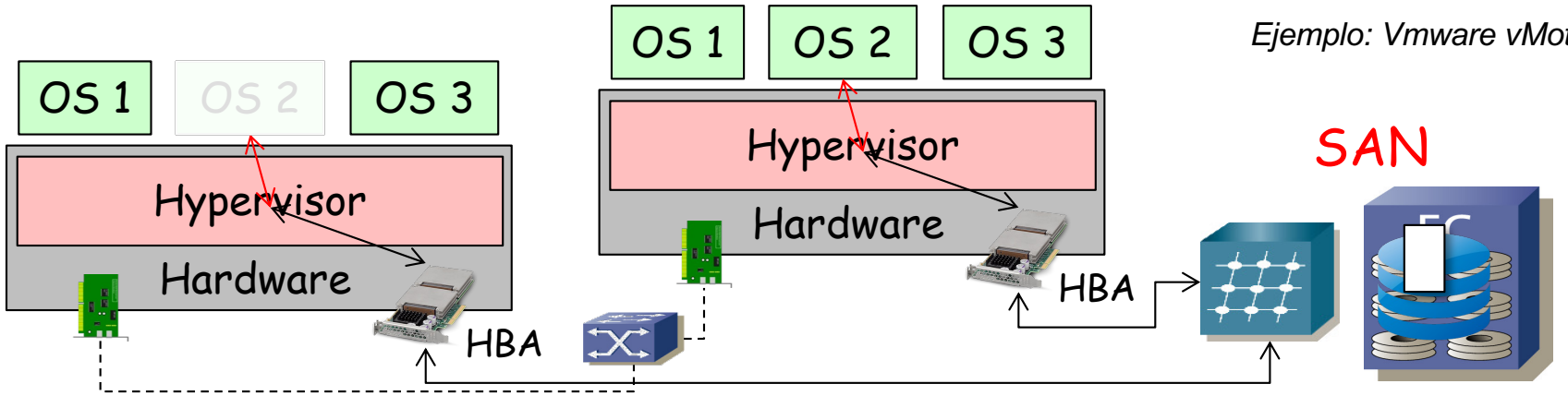
Acceso a disco desde la VM

- En el caso del almacenamiento lo más común es presentarle al guest dispositivos virtuales que responden a comandos SCSI
- De la máquina virtual se reciben comandos SCSI, que se responden obteniendo los datos del sistema de ficheros virtual
- El sistema de ficheros virtual puede almacenarse en un fichero
- Ese fichero puede estar en un disco local (SCSI o no)
- O puede estar en una SAN, por ejemplo mediante un HBA Fibre Channel
- Es decir, el S.O. entero (todo su sistema de ficheros) podría estar en la SAN
- El HBA también se puede virtualizar y ofrecer un HBA virtual a la VM
- También podría estar en un NAS (entonces con una NIC)



Virtual Machine Mobility

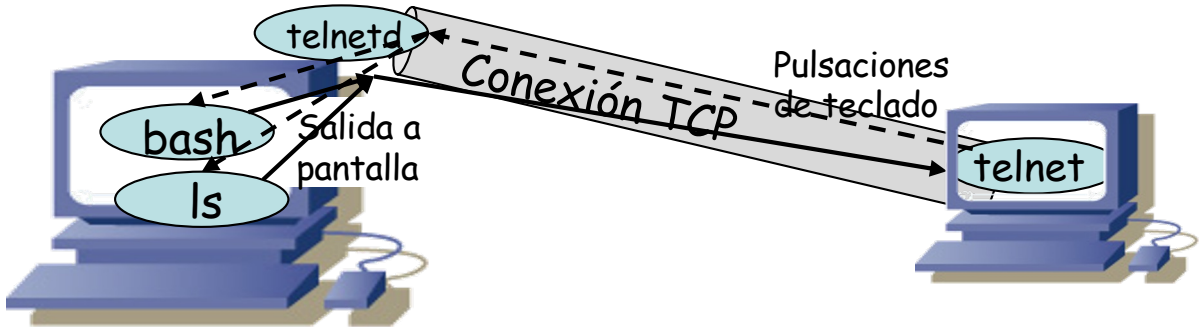
- Una VM se puede trasladar (en funcionamiento) a otro host
- No cambia su identidad ni detiene sus conexiones de red, mantiene su estado
- Hypervisor manda un ARP gratuito al reanudarla en el otro host
- La imagen de la máquina virtual está en una disco accesible por ambos hosts
- Esto puede requerir un sistema de ficheros que permita acceso concurrente desde los dos hosts (*clustered*)
- Consolidar las VMs puede permitir apagar servidores, incluso su refrigeración
- Hay que mover la RAM (por partes)
- Las VMs en un host pueden cambiar y pertenecer a diferentes VLAN
- Eso hace que el host deba recibir el tráfico de múltiples VLANs
- El vSwitch no tiene forma de informar al switch físico de las VLANs que necesita
- Así que se acaba configurando para que reciba el tráfico de todas
- Eso implica que debe procesar el broadcast de todas ellas



Ejemplo: Vmware vMotion

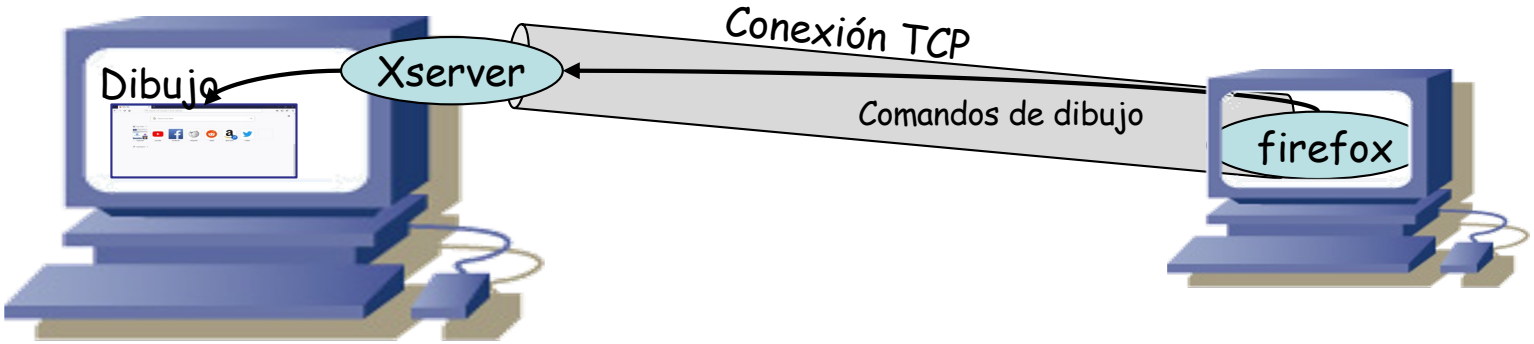
Acceso remoto

- Telnet, rsh, ssh... línea de comandos
- Conseguimos emplear una Shell que se ejecuta en otra máquina



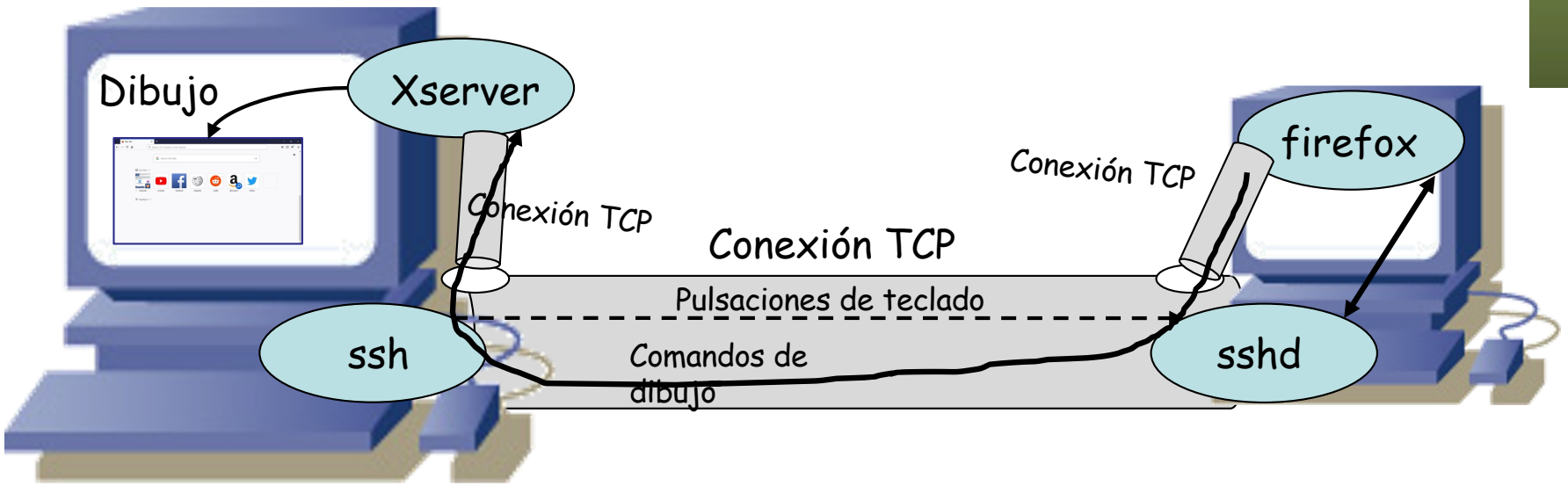
X Window System (X11)

- Arquitectura cliente-servidor
  - Servidor es responsable de dibujar en la pantalla
  - Clientes envían los comandos de dibujo al servidor
- La comunicación entre cliente y servidor es a través de un socket UNIX (local a la máquina) o a través de un socket TCP/IP



X Window System (X11)

- Podemos combinarlo con la redirección de puertos de ssh
- Convencemos a las aplicaciones de que busquen el servidor de X11 en el puerto reenviado por sshd

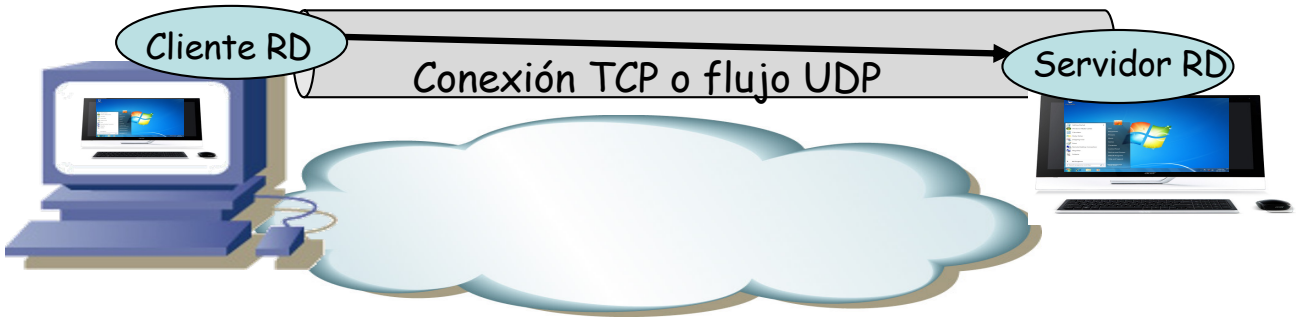


Escritorio Remoto

- Remote Desktop, Remote Frame Buffer
- Misma idea general pero para ver todo el escritorio
- Ejemplos: VNC (Virtual Network Computing), RDP (*Remote Desktop Protocol*), Citrix XenDesktop (ICA protocol = *Independent Computing Architecture*), VMware Horizon (with View), PCoIP (Amazon Workspaces), TeamViewer, Oracle Secure Global Desktop
- En algunos copia lo que va a la pantalla
- Puede no ir físicamente a la pantalla, permitiendo varios escritorios simultáneos

VDI = *Virtual Desktop Infrastructure*

- El escritorio que muestra es el de una máquina virtual en un servidor
- Y seguramente ni siquiera vaya a una pantalla física
- El PC del usuario pasa a ser un *thin client*
- La experiencia del usuario depende del servidor y de la red
- *Virtual Desktop Clouds*



Application Virtualization

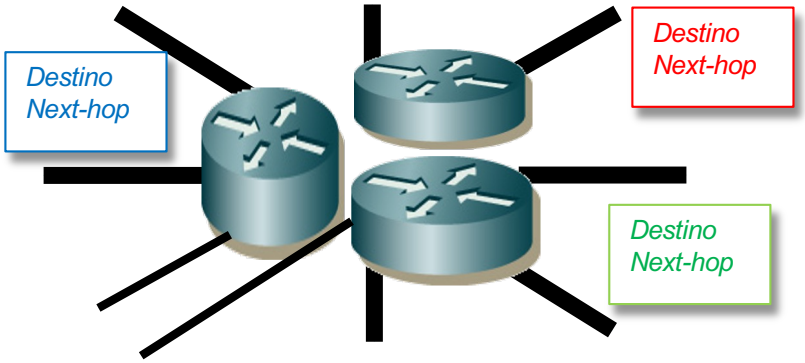
- En este caso el usuario ve solo la aplicación en cuestión
- Puede ejecutarse en el servidor o puede enviarse el binario al cliente
- Si se ejecuta en el servidor es más sencillo que funcione en cualquier plataforma del usuario
- Si se envía al PC del usuario debe ser capaz de ejecutarlo (nativamente o virtualizado) y le puede permitir modo offline
- Ejemplo: Citrix XenApp, Oracle Secure Global Desktop

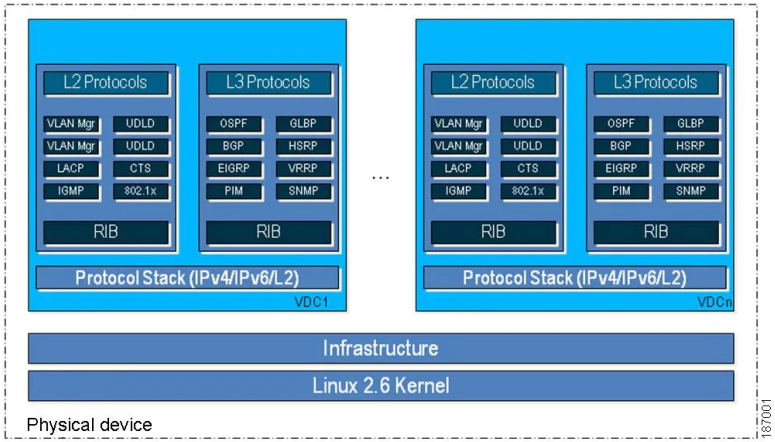
Cloud gaming

- Google Stadia (R.I.P.), NVIDIA GeForce Now, Xbox xCloud Gaming, Amazon Luna ...
- Pueden emplearse VMs con acceso a aceleración gráfica hardware
- Entonces la colocación de las VMs en hosts puede ser crítica

VRF-lite

- VRF = VPN Virtual Routing and Forwarding (entorno MPLS)
  - Versión ”lite” suele estar desligada de MPLS
  - Un router físico con múltiples interfaces físicos
  - Las VRFs aíslan los interfaces con tablas de rutas independientes
  - Los interfaces pueden ser virtuales (sobre VLANs)
  - No se enrutan paquetes de un interfaz de una VRF a un interfaz de otra VRF
  - Cada VRF podría emplear diferentes demonios de enrutamiento
- Contextos virtuales (VDCs en Cisco)
  - Separación de recursos físicos del equipo en contextos
  - Cada uno su conjunto de procesos y configuración
  - Posiblemente diferentes administradores que no ven el resto de VDCs
  - Por ejemplo el uso de unas tarjetas de puertas u otras o el reparto de TCAM
  - Más allá de las VRF (se pueden crear VRFs en cada VDC)
  - La comunicación entre VDCs tiene que ser a través de una interconexión física de dos puertos, uno en cada VDC





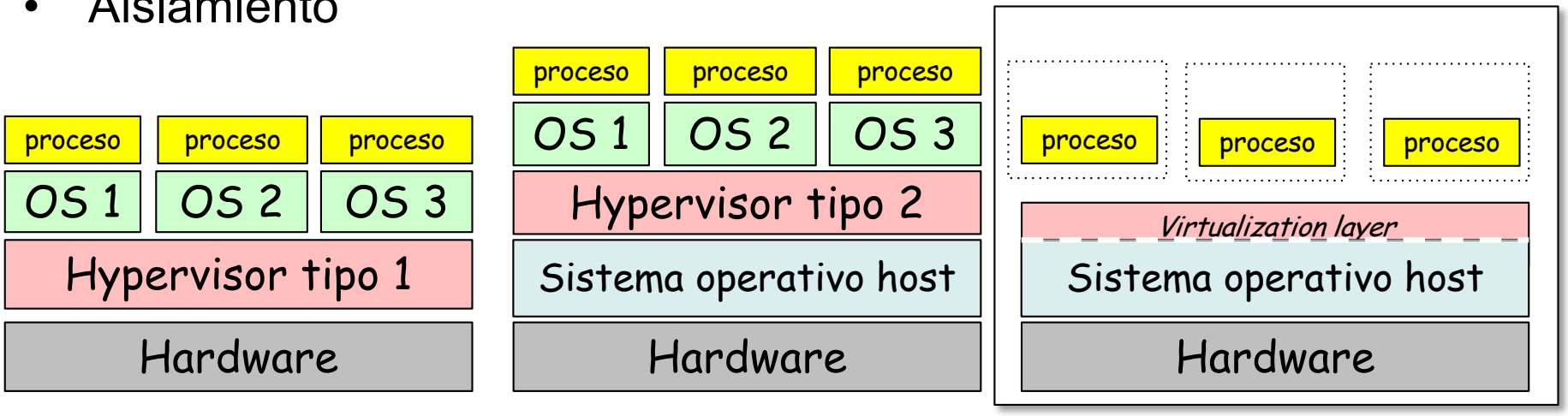
https://www.cisco.com/c/en/us/td/docs/switches/datacenter/nexus7000/sw/vdc/config/cisco_nexus7000_vdc_config_guide_8x/overview.html

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

Virtualización a nivel de sistema operativo

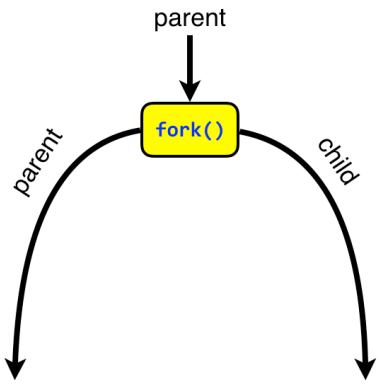
Operating-system-level virt.

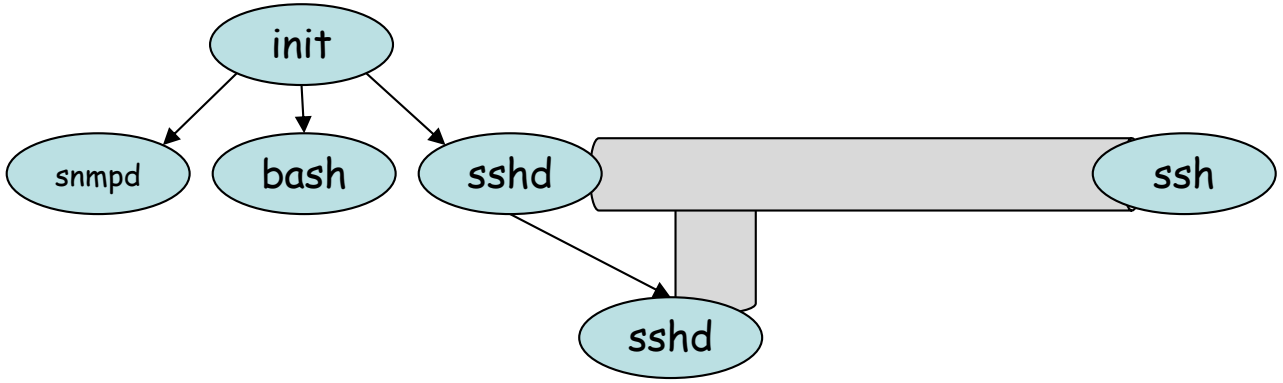
- *Containers, Virtualization Engines, Virtual Private Servers, Jails*
- Ejemplos: Docker, containerd, runc, crun, rkt, LXC, OpenVZ, Solaris Zones, FreeBSD Jails, etc
- En entornos UNIX, ahora mayormente en kernel Linux, también en Windows, también mediante virtualización de OS
- Virtualización de subsistemas del sistema operativo
- No hay múltiples kernels de múltiples OSs sino un solo kernel que virtualiza elementos suyos
- En realidad no son más que un conjunto aislado de procesos
- ¿Por qué? ¿Para qué? ¿Cómo?
- Aislamiento



Origen de los procesos

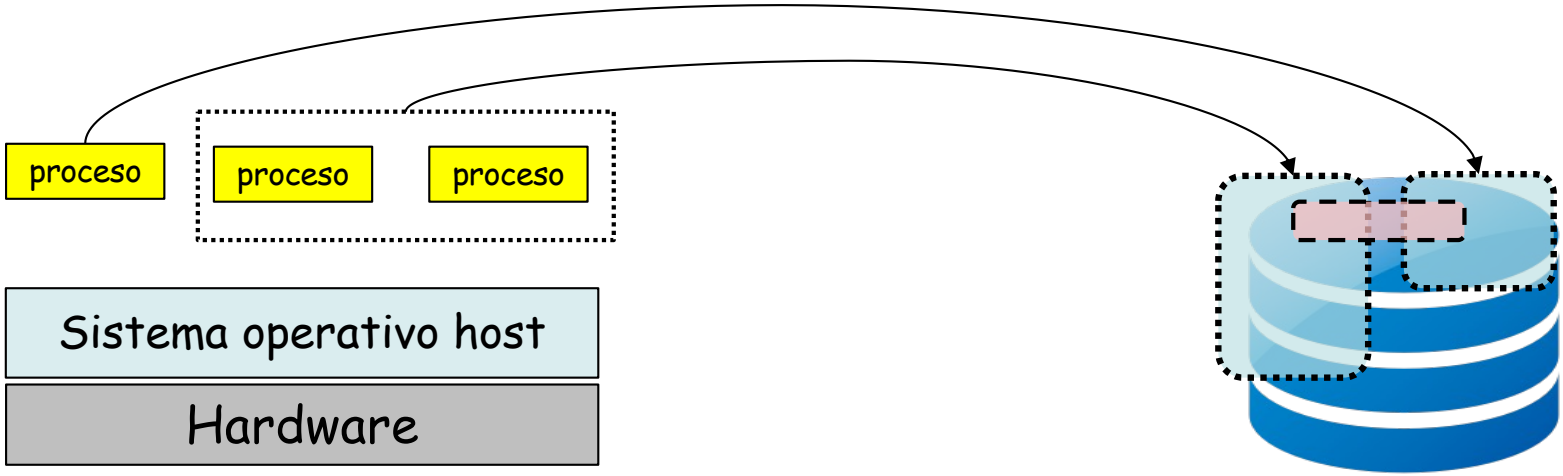
- Suele haber un proceso inicial que lanza el kernel (init, systemd, launchd...)
- Nuevos procesos son el resultado de otro proceso clonándose
- Normalmente en el proceso hijo pasa a ejecutar otro programa
- Tenemos un árbol de procesos; el proceso inicial tiene el Process ID (PID) 1
- El proceso hijo tiene una copia de la memoria en uso por el padre
- También una copia de estructuras en el kernel
- Eso incluye los *file descriptors* (descriptores de ficheros abiertos, sockets)
- Entonces por ejemplo si el padre tenía una conexión TCP y crea ahora un proceso hijo, el hijo tiene una copia de ese fd
- Los dos procesos pueden leer y escribir en la conexión TCP
- Habitualmente el padre cierra su extremo para que se haga cargo el hijo (pero esto ya es decisión del programador)
- En el ejemplo de servidor ssh el hijo podría cambiar ahora a ejecutar otro programa





Precursores: Chroot jail

- Normalmente un proceso ve todo el sistema de ficheros (dentro de los permisos otorgados)
- Podemos hacer que vea como / un subdirectorio del sistema
- Descendientes de ese proceso tienen la misma limitación
- Otros procesos pueden estar en otro *chroot jail*
- El sistema de ficheros que ve cada uno es un subdirectorio del original pero para ellos es todo el sistema de ficheros
- Dentro estarían todas las utilidades y programas
- Pueden compartir disco (por ejemplo mismo inodos)



Linux namespaces

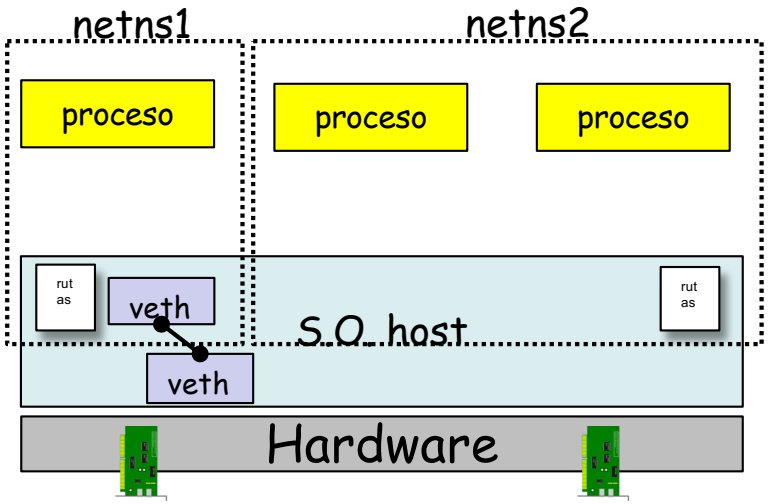
- Permiten que ciertos recursos del Kernel sean vistos por los procesos miembros del *namespace* como solo suyos
- Provee de una virtualización de dichos recursos

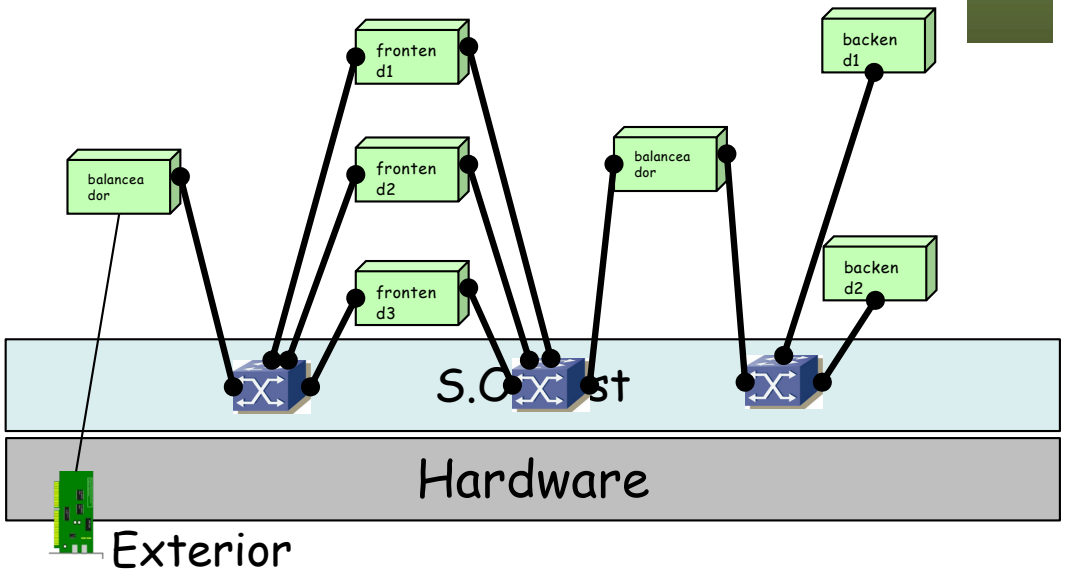
Network namespaces

- Permiten virtualizar la parte de red del Kernel
- Por ejemplo:
  - Que un conjunto de procesos vean algunos de los interfaces de la máquina y no el resto
  - Que vean una tabla de rutas que solo afecta a esos interfaces
  - Una tabla propia de reglas de filtrado
  - Lo mismo con la cache de ARP
  - Solo pueden crear sockets sobre direcciones de esos interfaces
- Los procesos hijo heredan los *namespace* del padre
- El *network namespace* desaparece cuando no quede ningún proceso en él (los interfaces volverán al *namespace* raíz)
- Comandos como: unshare --net, nsenter --target <PID> <CMD>

veth - Virtual Ethernet Device

- Interfaces Ethernet enteramente software
- Se crean en parejas, la trama que se envía por uno se recibe inmediatamente en el otro
- Podemos crearlos y después asignarlos a otro netns
- Podemos usar un par de veth para comunicar el namespace raíz con el host o dos namespaces entre ellos
- Pueden encaminar tráfico
- Compartir dominio capa 2 entre ellos uniéndolos a vSwitch





Linux namespaces

PID namespaces

- Permite crear jerarquías de procesos independientes
- Un proceso y sus descendientes no ven al resto de procesos
- Tenemos un nuevo proceso “1”, aunque desde la jerarquía global no se ve con ese PID

User namespaces

- Permite crear sets de usuarios autorizados independientes
- En el nuevo namespace hay un nuevo “root” (uid=0)
- Se mapean los usuarios del namespace en usuarios del padre

Otros: cgroup, IPC, Mount, UTS

Linux cgroup

nice

- Permite especificar un parámetro de prioridad en el *scheduling* de un proceso
- Es común a todos los sistemas Unix
- *ionice* permite algo parecido con el acceso a dispositivos de I/O (discos)
- Varios sistemas Unix implementan más de un *scheduler* (por ejemplo *priorities* o *round-robin*)

cgroups (Control groups)

- Permiten limitar el uso de recursos hardware que hacen los procesos
- Versión 2 desde Linux 4.5 (diferentes controles en v1 y v2)
- Un cgroup puede aplicar a un proceso o a un grupo de procesos, compartiendo ese límite
- Los descendientes de un proceso heredan su pertenencia a grupos
- cpu: Permite controlar el uso de tiempo de CPU
- cpuset:Permite asociar un grupo de procesos a una CPU
- memory: Limita la cantidad de memoria para procesos, kernel y swap
- devices: Controla la creación y el acceso a devices (por ejemplo HDs)
- net_cls: Permite clasificar los paquetes creados por un cgroup de cara a su planificación (tc)
- blkio: Limita los accesos a dispositivos de bloques

Contenedores y runtimes

- Los contenedores se construyen con varios de estos mecanismos
- Por ejemplo:
  - Independizar el *network namespace* pero no independizar los PIDs
  - O independizar los *namespaces* pero no limitar recursos
  - O limitar uso de CPU a un grupo de procesos pero no independizar *namespaces*
- Los *container runtimes* son utilidades (programas) para crearlos en base a estas funcionalidades en el Kernel
- Cada uno tiene su propia visión de cómo quiere que sean los contenedores
- *System containers*
  - El volumen (espacio en disco) del contenedor es toda una distribución de Linux sin el kernel (Alpine, CentOS, Ubuntu, Debian, Fedora, OpenWRT,...)
  - Librerías, programas, utilidades, Shell, su propio init
  - Corren un init (o similar) con PID 1 y los demonios típico de esa distribución
  - Ejemplo: LXC/LXD
- *Application containers*
  - Corren un solo programa en cada contenedor
  - Ejemplo: Docker

Runtimes

Runtimes

- runc, crun, containerd, CRI-O, kata-runtime, LXC/LXD, Docker
- OpenVZ, Firecracker, gVisor, runnc, libcontainer, podman
- runV, Clear Containers, rkt, virtcontainers
- Algunos incluyen un demonio que crea y gestiona los contenedores y utilidades para comunicarse con el demonio
- En sistemas no-Linux se puede emplear un hypervisor para virtualizar un Kernel de Linux (WSL2 o HyperKit)
- Muchos cumplen con la runtime-spec de OCI (Open Container Initiative)

Windows containers

- Windows 10 con Hyper-V o Windows Server
- Emplea Docker para su gestión
- Se pueden desplegar en Azure (Azure Kubernetes Service)
- Windows-based o Linux-based (virtuliza un Kernel Linux en una Ubuntu)

Orchestration

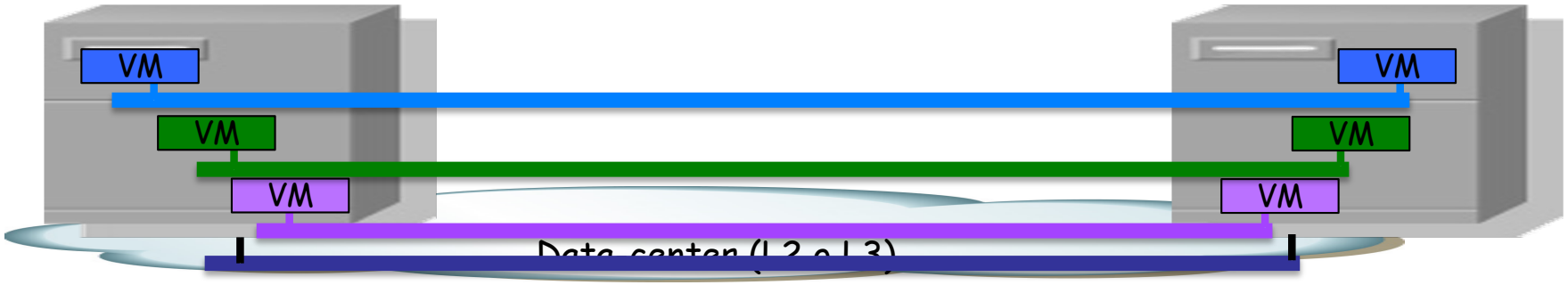
- Software para la gestión de contenedores
- Vigila el correcto funcionamiento de los contenedores para reiniciarlos
- Controla el networking (incluyendo balanceadores)
- Ejemplos: Docker Swarm, Kubernetes, Apache Mesos ...

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

Overlays en el data center

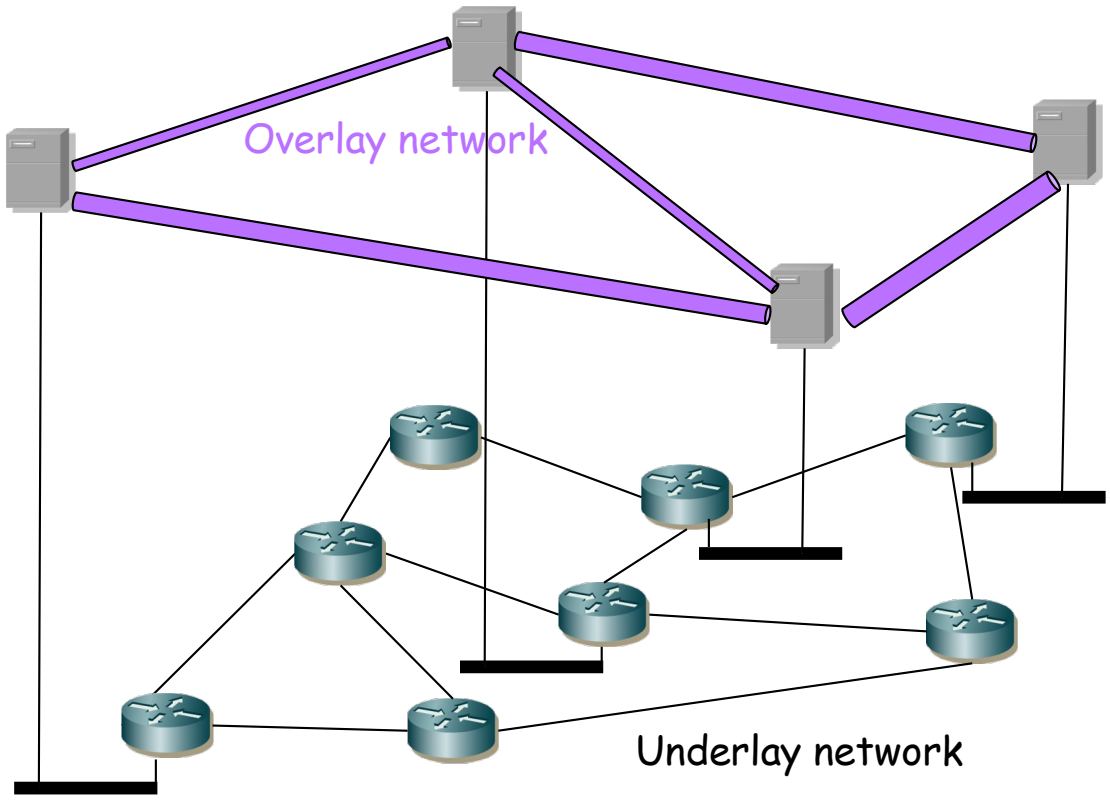
Multi-tenancy

- VMs (o contenedores) de diferentes clientes del DC o de diferentes departamentos de la empresa
- Las del mismo cliente deben poder estar en su propia red, aislada del resto
- Deben poder utilizar el direccionamiento que quieran sin colisión con otros clientes o con la *underlay network*
- Deben poder migrarse, sin hacer cambios en las VMs, por todo el DC
- ¿Podríamos emplear VLANs?
  - Mapear cada red de tenant a una VLAN del DC
  - Limitado a 4094 tenants (menos las VLANs que requiera el DC)
  - Se ven las MACs de todas las VMs en los conmutadores del DC
  - Todo el DC capa 2
  - Implica extender los dominios de broadcast por todo el DC
- IETF WG nvo3 (Network Virtualization over Layer 3)
- RFC 7364: “Overlays for Network Virtualization”
- Overlay Network: una red virtual con **separación entre *tenants*** (inquilinos) **sin conocimiento por parte de la *underlay network*** de dichos tenants para el forwarding
- Desacople entre underlay y overlays



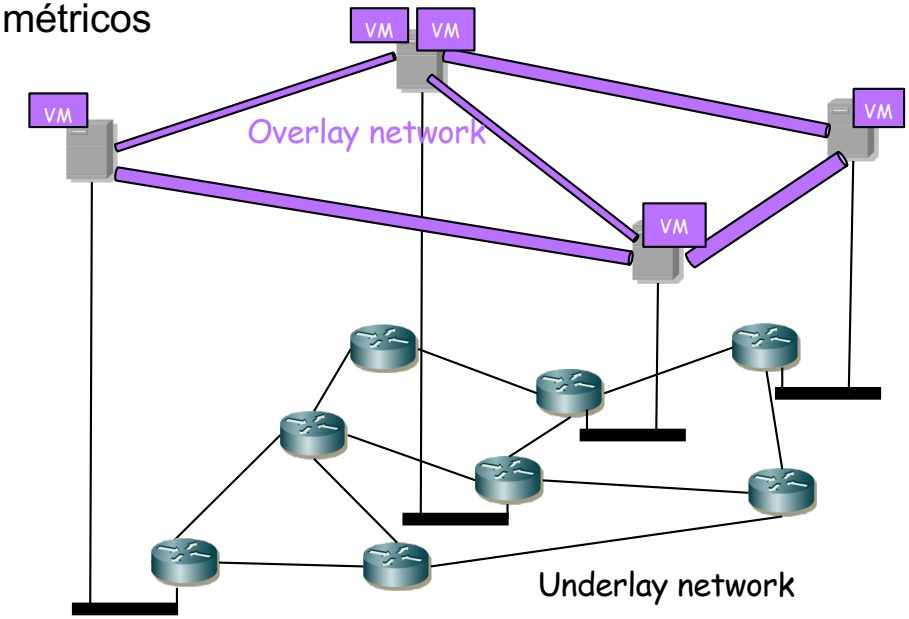
Overlay network

- El DC dispone una red (underlay network)
- Combinación de Ethernet, IP, MPLS, etc
- Los hosts donde corren las VMs del cliente están distribuidos por el DC
- Esos hosts crean la overlay mediante túneles



Overlay network

- El DC dispone una red (underlay network)
- Combinación de Ethernet, IP, MPLS, etc
- Los hosts donde corren las VMs del cliente están distribuidos por el DC
- Esos hosts crean la overlay mediante túneles
- Comunicación entre VMs:
  - Paquete de una VM de overlay se encapsula en el primer salto o NVE (*Network Virtualization Edge*) (Switch, router, vSwitch)
  - Túnel hasta el NVE remoto
  - La red reenvía en base a esta encapsulación, ignorando el contenido
  - El NVE de egreso desencapsula y entrega a la VM (o host físico) destino
- El paquete transportado puede ser IP o Ethernet
- El tráfico de cada túnel sigue el camino que elija la underlay
- Esos caminos podrían ser simétricos o asimétricos
- Transparente para la overlay
- Cada Virtual Network (VN) es una *overlay*



Overlays

- Debe permitir gran número de overlay networks
- Miembros de la overlay muy dispersos por el DC
- VMs de la overlay muy dinámicos (creación, destrucción, on, off, move)
- Sin requerir cambios en la underlay network
- Permiten que las tablas de direcciones MAC de los conmutadores de la underlay no crezcan con el número de VMs
- Para ello intentan evitar que los conmutadores del núcleo aprendan las direcciones MAC de las VMs (hosts de overlay)
- Encapsulando las tramas Ethernet de los hosts extremo
- Para entornos con mucho tráfico este-oeste en vez de norte-sur
- Comunicación con el exterior
  - Un equipo hará de *gateway*
  - Puede ser por ejemplo un equipo con interfaces en dos overlays
  - O con otro interfaz en una subred de la underlay
  - Puede ser una VM, un vSwitch o un equipo físico
  - Si enruta a otra overlay deben no colisionar sus espacios de direcciones
- BGP/MPLS o Ethernet VPNc, TRILL, SPB, NVGRE, OTV, VXLAN, FabricPath, LISP, Geneve, etc.

Overlay mediante túneles clásicos

Sobre IP

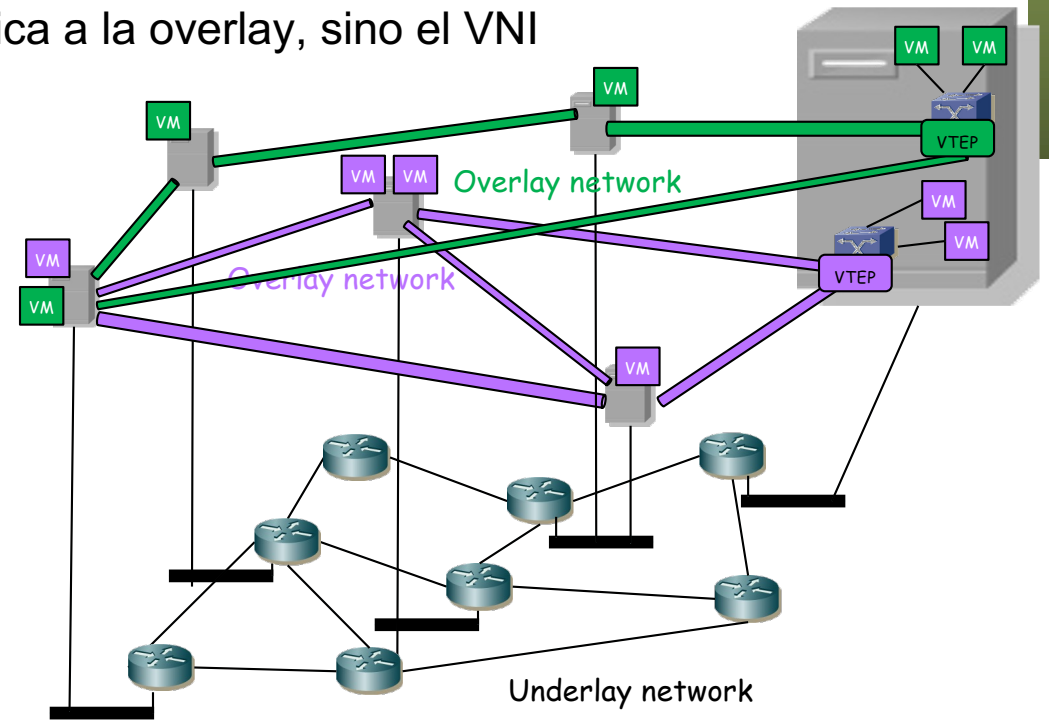
- Un extremo introduce el paquete a transportar en un paquete IP
- El paquete IP va dirigido a la dirección del otro extremo del túnel
- La red intermedia encamina en función de esa dirección destino, independiente del contenido
- ¿Qué podemos transportar dentro de IP? IPv4, IPv6, Ethernet (EthoIP)
- No existe un identificador de la overlay en el paquete recibido
- Se podría identificar por la dirección IP a la que va dirigido (una dirección IP en cada host para cada overlay)
- No podemos aprovechar ECMP en la underlay si queremos evitar desorden

Túneles GRE (RFC 2784)

- La cabecera básica GRE ocupa 8 bytes pero no existe identificador de overlay
- Uno de los campos es un Ethertype (*Protocol Type*)
- RFC 2890 “Key and Sequence Number Extensions to GRE”
- “Key” sirve para distinguir flujos dentro del túnel
- “Sequence Number”
  - Si hay “key” entonces el número de secuencia es por “key”
  - Permite dar entrega en orden (aunque no fiable)
  - Si llega uno “anterior” lo descarta
  - Si llega uno que deja un hueco puede guardarlo intentando reconstruir la secuencia
  - Pasado cierto tiempo sin lograr reconstruir los reenvía
- Muchos chips de conmutador pueden calcular el hash para ECMP usando el campo key de GRE, permitiendo el reparto multipath

VXLAN

- RFC 7348 “Virtual eXtensible Local Area Network (VXLAN): A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks” (agosto 2014)
- Emplea un esquema de overlay de capa 2 sobre capa 3 (o sea, un túnel), en el mismo data center o en otro
- Túnel sobre capa 4 pues hace el transporte sobre UDP
- Cabecera VXLAN con identificador de la Virtual Network (VNI)
- Contenido: trama Ethernet entregada por la VM
- El extremo del túnel es el VTEP (VXLAN Tunnel EndPoint)
- Puede estar en un hypervisor o en un switch físico cercano
- La dirección IP origen no identifica a la overlay, sino el VNI
- Otra overlay, otro VNI

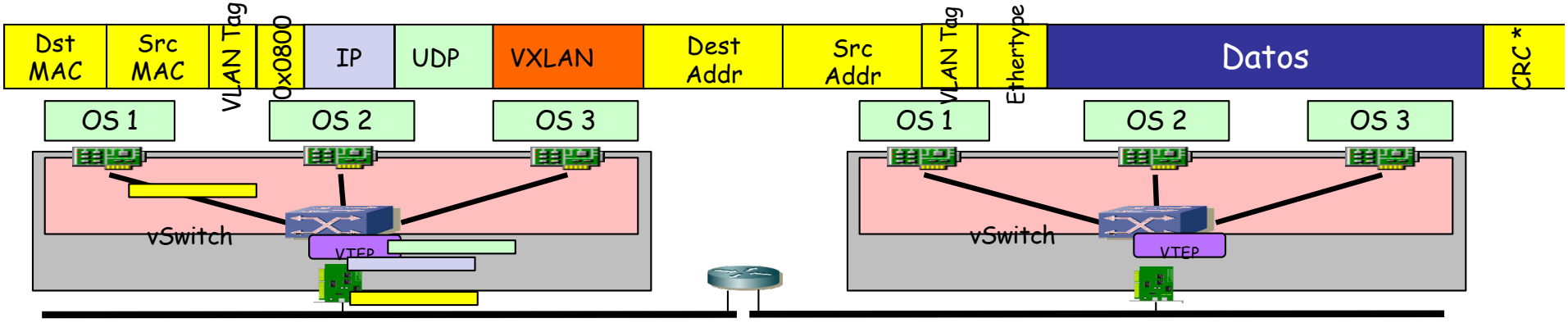


upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa
Redes de Nueva Generación
Área de Ingeniería Telemática

VXLAN Data Plane

VXLAN: Data plane

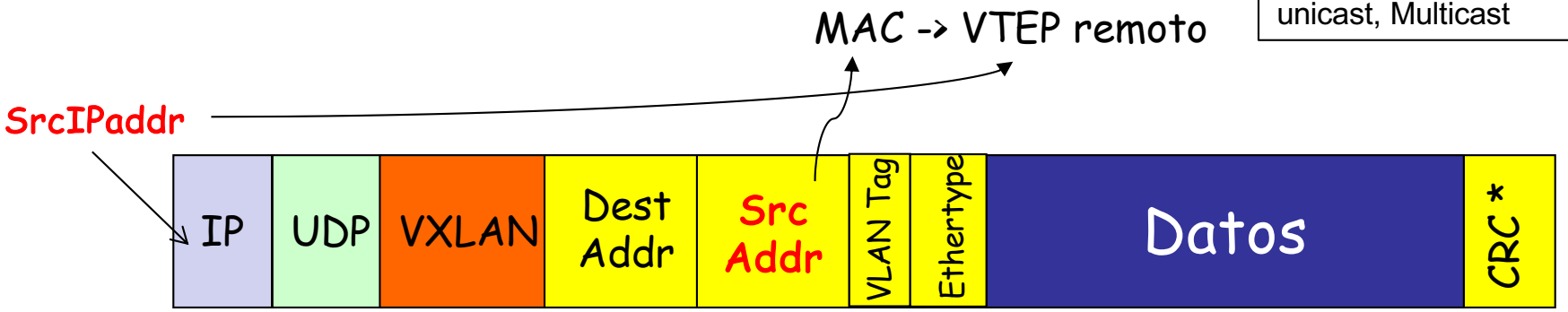
- Puerto destino 4789, puerto origen se recomienda un hash de campos de la trama original para facilitar el balanceo de flujos en la red IP
- La cabecera VXLAN es de 8 bytes y fundamentalmente contiene el VNI
- VNI = *VXLAN Network Identifier* (de 24 bits)
- Los VLAN Tags (trama externa e interna) son opcionales
- Para las máquinas virtuales es transparente
- Cada overlay se conoce como un “segmento VXLAN”
- Los hosts (VMs) de un segmento VXLAN solo pueden comunicarse entre ellos
- Se pueden repetir las direcciones MAC en distintos segmentos
- El VTEP se suele encontrar en el hypervisor o en un ToR switch
- La trama Ethernet la encapsula con el VNI en un datagrama UDP
- Averigua la dirección IP del host que contiene la VM con esa MAC destino
- Le envía el paquete IP que contiene la trama
- Por supuesto en una trama Ethernet



VXLAN: *Control plane*

- Los vSwitch deben aprender la dirección IP del host de la VM
- Aprende con información del plano de datos: al recibir un paquete de datos
- Aprende que la dirección MAC origen en la trama contenido es de un host en el VTEP con dirección IP la origen en el continente
- Cuando tenga una trama para esa MAC sabe a qué VTEP enviarla
- El BUM se envía a un grupo multicast IP (uno por segmento VXLAN)
- Todos los hosts del segmento VXLAN pertenecen a ese grupo
- Esto implica routing multicast en la red IP (algo como PIM-SM)
- El número de grupos multicast soportados por la red puede ser limitado, lo cual llevaría a compartirlos para varios segmentos VXLAN
- Hay soluciones unicast e híbridas, propietarias, mediante algún tipo de controlador o empleando MP-BGP
- Replicación unicast en origen es posible pero requiere que cada VTEP conozca a priori la dirección de cada otro

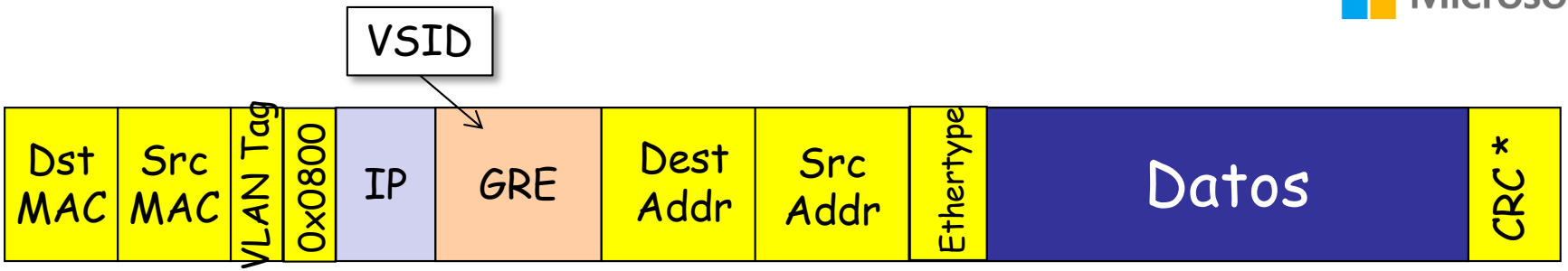
BUM = Broadcast, Unknown unicast, Multicast



NVGRE

- RFC 7637 "*NVGRE: Network Virtualization using Generic Routing Encapsulation",* RFC Informativa (Sept.2015) firmada por Microsoft
- Crea una topología capa 2 virtual sobre una red capa 3
- La trama (sin V-TAG) es encapsulada en el extremo (host, switch virtual, etc) en un paquete GRE y en un paquete IP (protocolo 0x2F)
- El extremo se llama el NVGRE Endpoint
- La cabecera GRE contiene un Virtual Subnet ID (VSID)
  - De 24 bits (parte del campo *key* de GRE)
  - Los 8 bits restantes de la clave se usan para distinguir flujos y poder hacer reparto de carga en routers que entiendan GRE
  - Permite identificar un dominio broadcast capa 2 en un entorno multi-tenant
- Mo detalla cómo el Endpoint conoce la dirección del host destino
- BUM: Se puede emplean encaminamiento multicast IP con una o más direcciones multicast por VSID o se puede implementar con N-way unicast
- Lo soporta Hyper-V (draft propuesto por Microsoft)
- NICs pueden soportar *offloading* del encapsulado NVGRE

Microsoft



Generalizaciones

Geneve

- RFC 8926 (Noviembre 2020): “Geneve: Generic Network Virtualization Encapsulation”, firmada por Intel y Vmware
- Formato estandarizado para el data plane
- UDP, puerto destino 6081, origen un hash de header encapsulado
- Protocol Type es el Ethertype del contenido (0x6558 = Ethernet)
- Deja indeterminado el plano de control (draft-ietf-bess-evpn-geneve-05)
- VNI de 24 bits
- 0 o más opciones en formato TLV

VXLAN-GPE

- draft-ietf-nvo3-vxlan-gpe-12 (2021-09-22, *expired*, Cisco, Arrcus, Intel)
- GPE = Generic Protocol Extension
- Next Protocol (IPv4, IPv6, Eth, NSH, etc)
- OAM, ingress replicated BUM

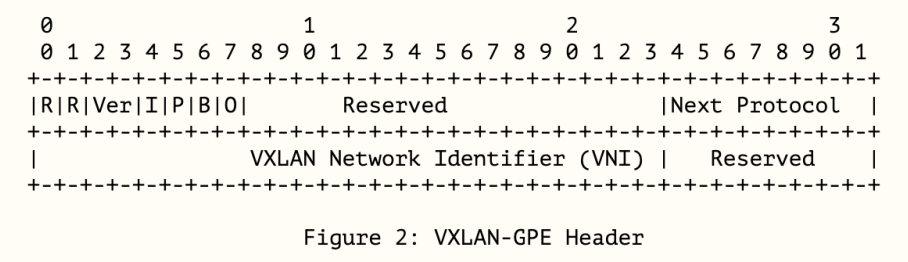


Figure 2: VXLAN-GPE Header

https://datatracker.ietf.org/wg/nvo3/documents/
https://datatracker.ietf.org/wg/bess/documents/

upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

Redes de Nueva Generación
Área de Ingeniería Telemática

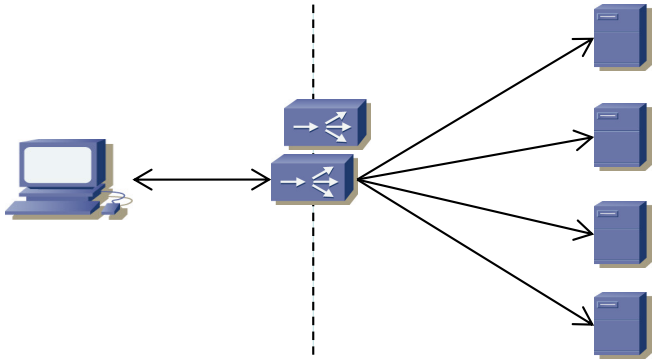
Servicios de red

Balanceo de carga

- Antes se hacía cambiando la respuesta DNS
- Balanceadores carga entre los servidores repartiendo las peticiones
- Normalmente *virtualizan* el servicio ofrecido tomando ellos su dirección IP
- Ocultan las direcciones de los servidores, que normalmente son privadas
- Se puede retirar o añadir un servidor sin detener el servicio
- Hacen un seguimiento del estado de los servidores/el servicio (*health tracking*)
- *In-band health tracking*
  - Balanceador monitoriza el tráfico del servidor
  - Esto comprueba que está activo pero no que el servicio se comporta correctamente
  - No requiere que el balanceador entienda el servicio
  - Por ejemplo le reenvía el SYN y si no contesta puede enviar la retransmisión del SYN a otro servidor
  - Si falla numerosas veces puede retirarlo del servicio balanceado
  - Puede monitorizar los códigos de respuesta (por ejemplo HTTP)
- *Out-of-band health tracking*
  - Balanceador sondea activamente al servidor
  - Puede comprobar solo que la máquina está activa (*server availability*)
  - Puede comprobar que la aplicación está activa (*application availability*)
  - Puede pedir recursos de la aplicación para verificar que entrega lo correcto (*application consistency*)

Balanceadores y *failover*

- *Stateless failover*
  - Pasa a usarse el de respaldo al fallar el principal
  - No se mantiene el estado de las sesiones
  - Es útil cuando no hay sesión y las conexiones son muy cortas
- *Stateful failover*
  - Replica el estado como para seguir mandando las conexiones al mismo servidor
  - Útil para conexiones de larga duración
- *Sticky failover*
  - Replica en el de respaldo suficiente estado para seguir reenviando al cliente al mismo servidor
  - Eso permite mantener la sesión aunque rompa la conexión
  - Útil cuando las conexiones son cortas pero se quiere mantener la sesión



Balanceadores

Dispatch mode

- Modifica solo L2: MAC src por la suya (o no) y dst por la del servidor elegido
- Balanceador y servidores adyacentes en capa 2
- Todos los servidores tienen configurada la VIP como secundaria
- Retorno por router por defecto (si origen en misma LAN debe pasar por balanceador)
- Balanceador no suele tener todas las funcionalidades de un switch capa 2 (ej. STP)

Server NAT mode

- Modifica direcciones MAC y la dirección IP destino
- Los servidores pueden estar en otra subred, a varios saltos
- Los servidores no necesitan tener configurada la VIP
- El tráfico de retorno debe pasar por el balanceador para deshacer el cambio de dirección IP

Client NAT mode

- Se modifica la dirección IP origen
- Simplifica el conseguir que el tráfico de retorno pase por el balanceador
- Los servidores dejan de conocer la dirección IP del cliente (de cara a hacer estadísticas)

Direct Server Return

- El tráfico de retorno NO pasa por el balanceador
- El balanceador entonces no puede más que reescribir direcciones MAC
- Los servidores deben tener todos configurada la dirección VIP
- No debe caducar el estado simplemente por no ver el tráfico en el otro sentido
- Requiere menos trabajo del balanceador y por lo tanto soporta mayores cargas

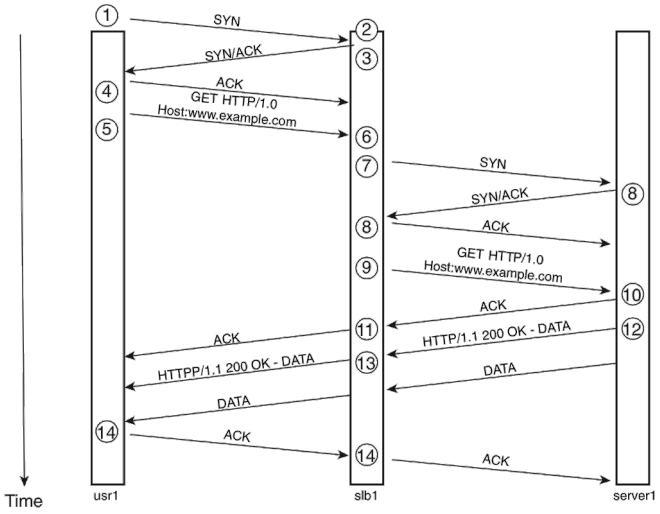
Balanceadores: *Connection Spoofing*

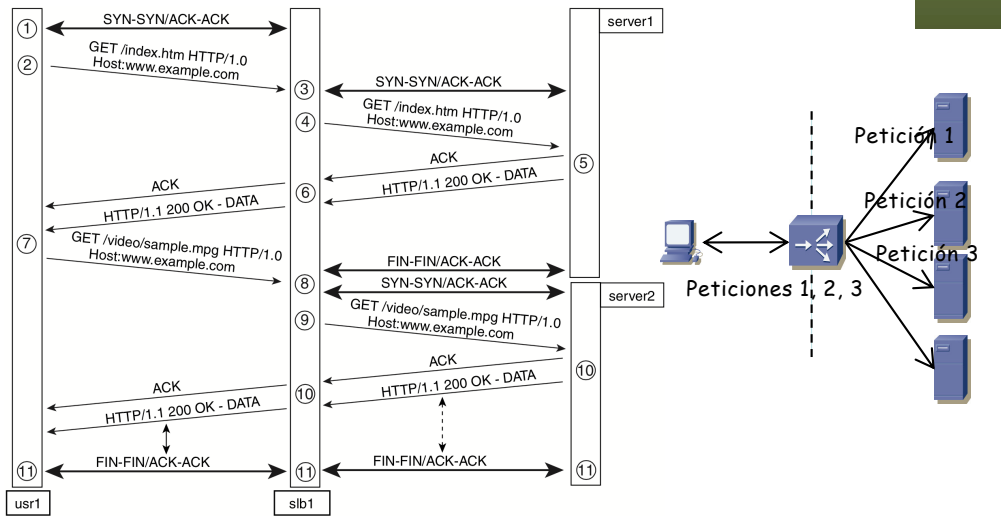
"TCP termination","delayed binding", "connection splicing"

- En estos casos el balanceador hace de proxy
- Termina la conexión de cara al cliente y la inicia él de cara al servidor
- Esto le permite elegir el servidor en función de información de capa 5+
- Por ejemplo en HTTP en función del URL

"Connection remapping":

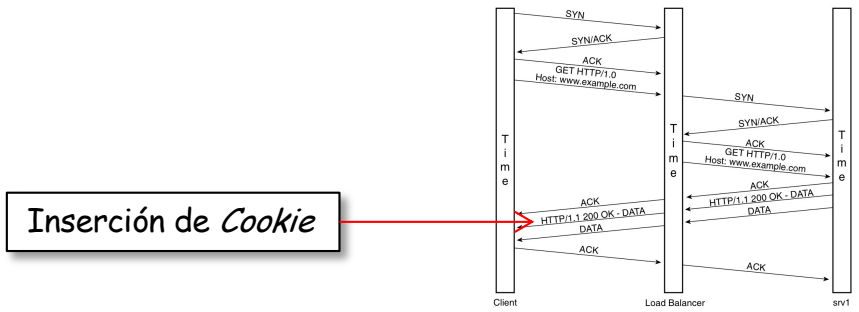
- El balanceador puede repartir la carga de las peticiones entre diferentes servidores (si no requiere mantener la sesión)
- Pueden ser peticiones de recursos diferentes que deban ir a diferentes granjas de servidores aunque se publiquen tras la misma VIP



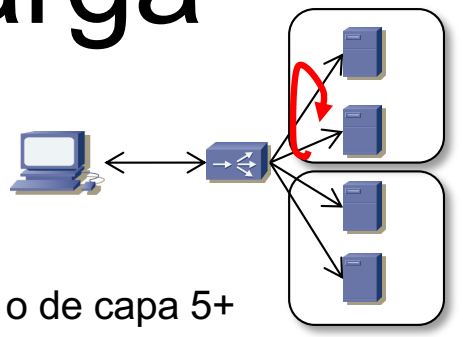


Balanceadores y HTTP

- Suele hacer falta que todas las conexiones del mismo cliente vayan al mismo servidor
- Para que se pueda mantener la sesión sin emplear *clustering* entre servidores
- Con web services no suele haber necesidad
- *"Session persistance", "stickiness", "sticky connections", "afinidad de sesión"*
- *Source IP sticky*: basarse en la dirección IP origen
  - Es factible en entornos donde se tienen controlados a los clientes (una Intranet)
  - Es problemático si los clientes son cualquiera en Internet porque podría el cliente estar empleando una granja de proxies que hicieran que varias conexiones vinieran con diferentes IP origen
  - Pero al menos funciona aunque se emplee HTTPS desde el cliente hasta el servidor
- *Cookie sticky*: inserción de *cookies*
  - El balanceador/servidor inserta una *cookie* en la respuesta (*cookie insert/passive*)
  - Le permite al balanceador reconocer al cliente en la siguiente petición mantener al servidor
  - Comunicación con el servidor sin encriptar para poder ver y/o insertar la cookie
  - O debe disponer de la clave privada para desencriptar y re-encriptar el flujo SSL del servidor
  - Cuando emplean información de capas 5+ se les suele llamar *content switches*
- También puede hacer coalescencia de peticiones de diferentes clientes
- Aprovechar conexiones persistentes HTTP 1.1 con el servidor y evitar fases de *slow start*
- O para reducir el número de conexiones que mantiene el balanceador con los servidores o que mantienen estos últimos



Reparto de la carga



- Pasos
  1. Seleccionar la granja de servidores
     - Pueden estar en distintos CPDs
     - Suele ser en función de parámetros de la conexión o de capa 5+
  2. Seleccionar el servidor (algoritmos de planificación)
- Algoritmos de planificación (lista parcial)
  - *Round robin*
    - Cada nueva petición se entrega al siguiente de la lista de forma circular
  - *Weighted round robin*
    - Un peso por servidor controla cuántas peticiones recibe ese servidor en un turno
    - Equilibrar la carga cuando los servidores son de diferentes capacidades
  - *Least connections*
    - Nueva conexión al servidor con el cual el balanceador tiene menos conexiones
  - *Weighted least connections*
    - Se ordena a los servidores en base a "nºconexiones"/weight
    - Se selecciona al de menor métrica
  - *Fastest*
    - Estimación de lo "rápidos" que son los servidores en contestar al SYN
    - Normalmente al aumentar la carga del servidor aumenta ese tiempo
    - Suele haber un punto a partir del cual empeora rápidamente
    - Puede que el balanceador no reaccione lo suficientemente rápido

Otros servicios de red

Firewalls e IDS

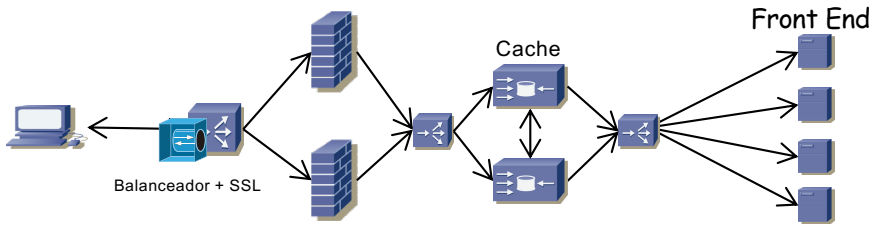
- Reglas de filtrado e inspección para permitir el acceso solo a las direcciones IP y puertos de los servicios
- Pueden estar antes o después del balanceador
- Si no vale con uno se pueden poner varios balanceados (aumenta la complejidad)

SSL offloading

- Portales web seguros y otros servicios sobre un túnel SSL
- Este equipo termina la sesión SSL con el usuario e inicia una conexión sin SSL con el servidor
- El equipo puede disponer de hardware especializado para SSL
- También podríamos poner varios y balancearlos o que el balanceador integre esta funcionalidad

Cache

- Cercanas al servidor ("*reverse proxy cache*", RPC) reduce carga sobre los servidores
- Cercanas al cliente ("*transparent caching*") reducen carga sobre el enlace a Internet y los tiempos de respuesta por cercanía (menor RTT)
- La cache podría implementarse con varias caches balanceadas
  - Aumenta la capacidad (CPU) de la cache
  - Busca maximizar el *cache hit ratio* y así reducir peticiones a servidores
  - Balanceador debería reenviar la petición a la cache con mayor probabilidad de contenerlo (en base al FQDN)
  - O las caches deben sincronizarse (*clustering*), para no acabar haciendo peticiones repetidas
- Puede redirigir parte de la petición a un antivirus o filtro de contenido o a varios balanceados
- Se encargarían de verificar que se puede hacer esa petición o que el documento obtenido no es peligroso
- Protocolos específicos para pasar la petición/respuesta: ICAP = *Internet Content Adaptation Protocol*
- El balanceador puede ser el mismo equipo



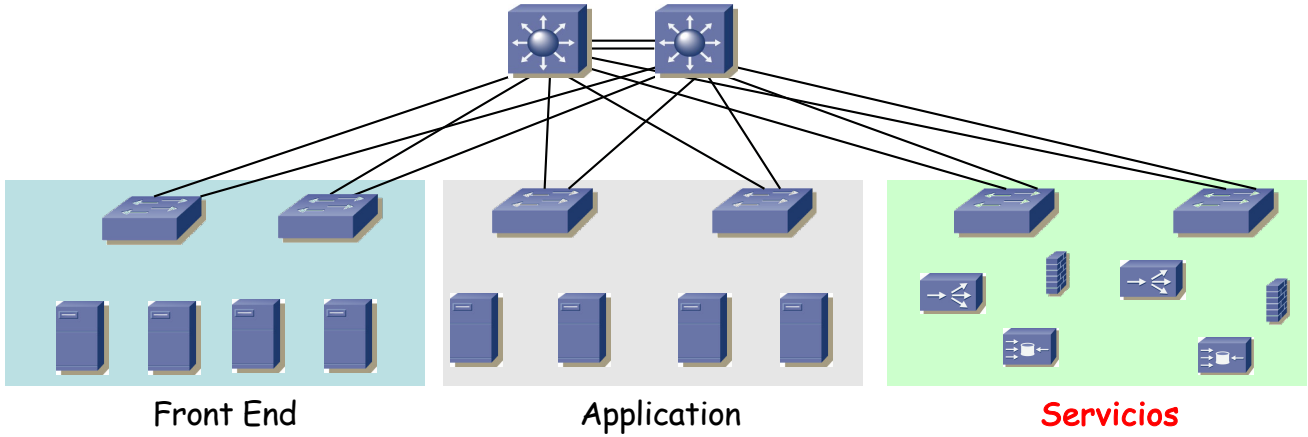
Servicios y redundancia

Redundancia

- Se pueden dar desde equipos independientes (replicados y con posible sincronización)
- También pueden ser módulos en un conmutador (idem)

Colocación

- Es común que los *tiers* estén en la capa de acceso
- Con un diseño colapsado los servicios estarían conectados a los conmutadores de agregación o pueden ser módulos en los conmutadores de agregación
- Pueden ser compartidos entre las diferentes capas
- Esto puede ser gracias a que tenga varios interfaces físicos, en las diferentes VLANs
- Porque emplee trunking en su(s) interfaz(-ces)
- Puede incluso dividirse en varios balanceadores “virtuales”
- Compartirlos reduce costes pero aumenta la complejidad y lwa requiere mayor potencia
- Los equipos pueden ser demasiados para los slots de los conmutadores de agregación
- O demasiados para los puertos de los conmutadores de agregación
- Podemos sacarlos a su propia capa de acceso



Orden de los servicios

Router-Firewall-Balanceador

- Si el balanceador se comporta como un puente el router por defecto será el firewall
- Si el balanceador se comporta como router se vuelve el router por defecto para servidores
- Entre los elementos redundados estará el balanceador
- Si es el router por defecto puede emplear el FHRP
- Pueden configurarse en activo-pasivo o activo-activo
- Activo-pasivo (*active-standby*)
  - Uno de ellos hace todo el trabajo y si falla entra el otro
  - Mantener el estado sincronizado es sencillo (un solo sentido)
- Activo-activo (*active-active*) con reparto de VIPs
  - Las direcciones virtuales de los servicios se reparten
  - Cada dirección es empleada por un balanceador y el otro es el de respaldo
  - Los servidores tendrán de router por defecto al balanceador que gestione como primario la dirección IP de su servicio
- Activo-activo con VIPs replicadas
  - Las direcciones IP de los servicios están activas en los dos
  - Hay que conseguir que el mismo cliente (toda su sesión) vaya siempre al mismo equipo
  - Esto es complejo pues los equipos *upstream* son conmutadores capa 2 y/o 3 que no entienden de sesiones
  - Normalmente eso requiere repartir a los clientes entre las dos instancias de la dirección IP virtual

Orden de los servicios

Router-Balanceador-Firewall

- En este caso tras el router está el balanceador y detrás el firewall
- El firewall debe permitir que el balanceador verifique el estado de los servidores (*health probes*)
- Esto implica configuración
- En esta configuración el router por defecto es el firewall
- Si el firewall actúa como router los *health probes* del balanceador deben ser enrutables

Firewall-Router-Balanceador

- En este caso la entrada es por el firewall
- Es probable que requiera funcionalidades extra de router como por ejemplo integrarse en el IGP
- Es más difícil securizar cada *tier* pues están todos al otro lado del firewall, enrutados sin pasar por el fw
- Según cómo opere el balanceador el router por defecto es él o el router

Firewall-Balanceador-Router

- El router como router por defecto para los servidores
- Eso permite usar funcionalidades habituales suyas como un FHRP, QoS, relay DHCP, etc.
- El balanceador no puede emplear una técnica que le requiera conectividad L2 con los servidores
- Los *health probes* que envíe el balanceador deben ser enrutables
- De nuevo pasar por el firewall entre cada capa requiere volver upstream

I/O Consolidation

I/O Consolidation

- Emplear la misma infraestructura física para transportar múltiples tipos de tráfico
- Parece que finalmente van a converger sobre Ethernet (¿ o sobre IP ?)
- En lugar de NICs y HBAs tendremos solo CNAs
- CNAs (*Converged Network Adapters*) son NICs Ethernet que implementan por hardware parte de los protocolos para la I/O consolidation
- Simplifica el cableado y la infraestructura de red (de 2 redes a solo 1)
- El equipo retirado nos ahorra costes de alimentación, refrigeración y espacio
- Desventajas
  - Una nueva tecnología a dominar y gestionar; hay que cambiar el hardware
  - Posibles acoplamientos y mayor fragilidad
- Transportar los datos de LAN sobre FC no ha sido interesante por la falta de buen soporte de multicast/broadcast en FC
- SCSI requiere transporte fiable porque se diseñó para cables paralelos fiables y su recuperación ante pérdidas es lenta
- iSCSI es una opción pero tampoco se ha desplegado masivamente por el posible sobrecoste de TCP y porque no mantiene la gestión de FC
- En general los transportes sobre IP son más costosos en el hardware
- Finalmente Ethernet parece la solución ganadora, pero hay que modificarlo para evitar las pérdidas

DCB: Data Center Bridging

- Modificaciones a Ethernet para permitir la convergencia con I/O en el data center
- Entornos con un producto RTTxBW limitado, así como un limitado número de saltos

Priority-based Flow control (PFC, 802.1Qbb)

- Como en FC, control de flujo salto a salto
- Para que no haya pérdidas por congestión para los protocolos (clases) que así lo requieran
- Se hace control de flujo de forma independiente para cada clase de servicio de 802.1p
- La congestión en un switch propaga el control *upstream y p*uede afectar a flujos que no son responsables de la congestión

Enhanced Transmission Selection (ETS, 802.1Qaz)

- ETS no concreta el planificador a usar pero sí los requerimientos que debe cumplir
- Por defecto deberían soportar un planificador con prioridades estrictas
- También puede soportar un "*credit-based shaper*", que es un *token bucket* para cada cola
- La tercera opción es el algoritmo ETS (de la familia de los WRR)

Data enter Bridging Exchange Protocol (DCBX, extensión de LLDP)

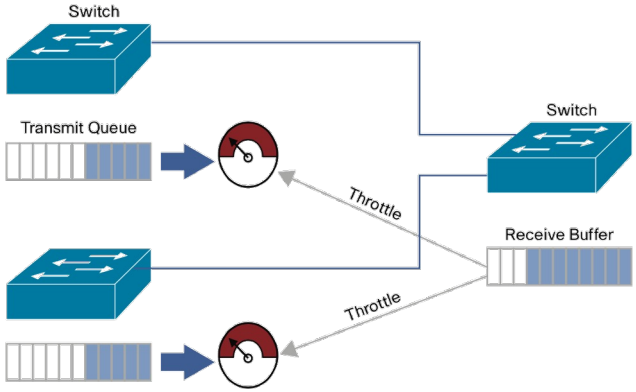
- Permite descubrir las capacidades de los extremos de un enlace; detectar y resolver conflictos en la configuración
- Permite la configuración de parámetros del otro extremo
- Empleado principalmente para parámetros de PFC y ETS

Feature	Benefit
Priority-based Flow control (PFC; IEEE 802.1 Qbb)	Provides capability to manage bursty, single traffic source on a multiprotocol link
Enhanced transmission selection (ETS; IEEE 802.1 Qaz)	Enables bandwidth management between traffic types for multiprotocol links
Congestion notification (IEEE 802.1 Qau)	Addresses the problem of sustained congestion by moving corrective action to the network edge
Data Center Bridging Exchange (DCBX) Protocol	Allows autoexchange of Ethernet parameters between switches and endpoints

DCB: Data Center Bridging

Quantized Congestion Notification protocol (802.1Qau)

- Para dominios con BWxRTT<5 Mbits, con enlaces de 10Gbps quiere decir un RTT<0.5ms
- Es decir, data centers, backplanes, computing clusters, SANs
- Habilita la capacidad en los puentes (y hosts) de enviar señales de congestión a las estaciones finales para que limiten la tasa
- La fuente puede etiquetar las tramas con un CN-Tag (2 bytes Flow ID)
- Se le devuelve en el mensaje de CN
- Le permite identificar el flujo cuya tasa debe reducir
- Al que detecta la congestión se le llama *Congestion Point (CP)*
- Debe enviar el mensaje de notificación antes de llenar el buffer pues en lo que llega el mensaje seguirá recibiendo paquetes (estimación)
- Lo envía a la dirección origen del paquete con el que toma la decisión
- Quien recibe la notificación se llama el *Reaction Point (RP)*
- Al recibir la notificación el RP reduce la tasa del flujo mediante un *rate limiter*
- No recibe indicación de que pueda aumentar la tasa de nuevo así que lo hace unilateralmente



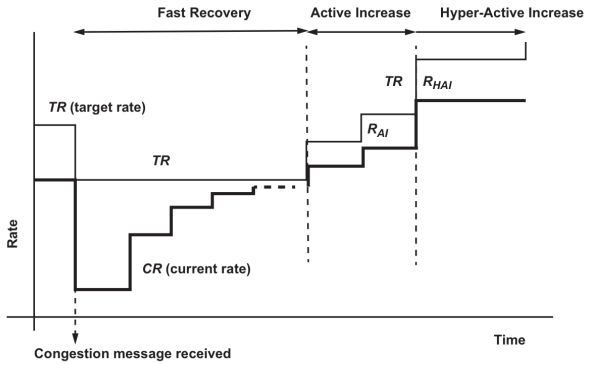


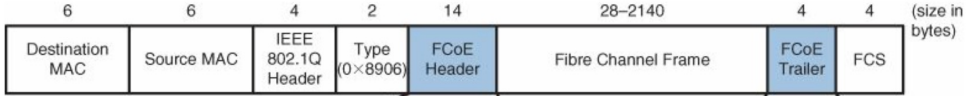
Figure 30-3—QCN RP operation

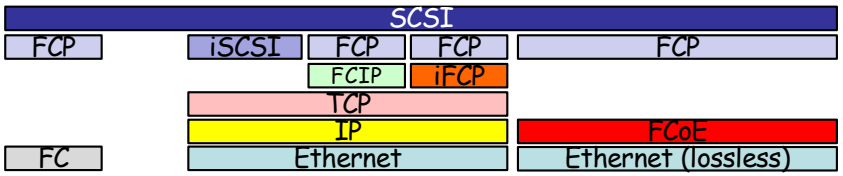
802.1Q

FCoE

InterNational Committee for Information Technology Standards

- *Fibre Channel over Ethernet* (Estándar FC-BB-5 del grupo T11 del INCITS)
- Mantiene el funcionamiento y la gestión de FC
- Tiene sentido a partir de 10GE, cuando está a la par con velocidades en Fibre Channel
- Requiere una Ethernet sin pérdidas (PFC)
- Requiere soporte de *jumbo frames* para transportar en una sola trama Ethernet toda la trama FC (36 bytes cabecera + hasta 2112 de datos)
- FC-0 y FC-1 son la capa física, sincronización, errores
- FC-2 se encarga del formato de la trama, señalización y gestión
  - FC-2V define el interfaz con FC-3 y las funciones que se le ofrecen, independientemente del FC-1
  - FC-2M define la multiplexación para cuando hay múltiples FC-2V sobre un FC-2P o viceversa
  - FC-2P implementa el control de flujo, se sustituye por el de PFC
- Mantener FC-2V hace FCoE transparente para el sistema operativo
- FCoE emplea el FCoE Initialization Protocol (FIP) para convertir la Ethernet multi-acceso en un conjunto de enlaces punto-a-punto virtuales
- Transporta también los servicios de descubrimiento y login de Fibre Channel
- FCoE emplea el Ethertype 0x8906 mientras que FIP el 0x8914
- FCoE es el plano de datos, FIP el de control





Product Naming	Throughput (MBytes/s)*	Line Rate (Gbaud)†	IEEE Standard Complete (Year)‡	Market Availability (Year)‡
10GFCoE	2,400	10.3125 NRZ	2002	2008
25GFCoE	6,000	25.78125 NRZ	2016	Market Demand
40GFCoE	9,600	4X10.3125 NRZ	2010	2013
50GFC0E	12,000	2x25.78125 NRZ	2016	Market Demand
50GFCoE	12,000	26.5625 PAM-4	2018	Market Demand
100GFCoE	24,000	4X25.78125 NRZ	2010	2017
200GFCoE	48,000	4X26.5625 PAM-4	2018	Market Demand
400GFCoE	96,000	8X26.5625 PAM-4	2020	Market Demand

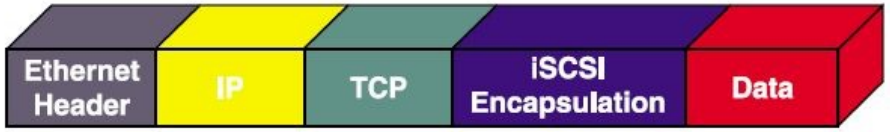
iSCSI

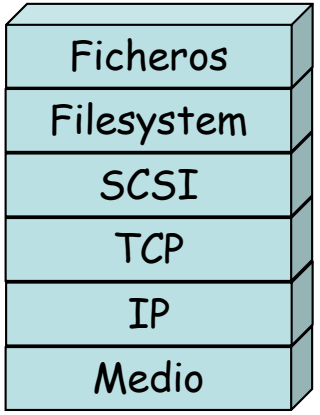
iSCSI

- RFC 7143 *"Internet Small Computer System Interface (iSCSI) Protocol (Consolidated)"*
- Transporte de comandos SCSI sobre una (o varias) conexiones TCP
- Permite crear una SAN atravesando una red IP
- Es una alternativa de bajo coste a Fibre Channel
- El coste puede estar en el rendimiento (retardo, pérdidas, congestión)
- Si por debajo empleamos Ethernet no ha sido interesante hasta que Ethernet ha alcanzado las velocidades de FC

iSCSI vs FCoE

- Ethernet ya tiene velocidades comparables a FC
- Pero hay otra alternativa que es FCoE
- FCoE permite una integración nativa con equipamiento FC pues transporta sus tramas
- iSCSI no transporta las tramas FC sino comandos SCSI
- Hace falta una pasarela para integrar interfaces iSCSI con FC, con sus problemas de escalabilidad y puntos de fallo
- La gestión cambia entre FC e iSCSI
- iSCSI puede ser más apropiado en un escenario *greenfield*





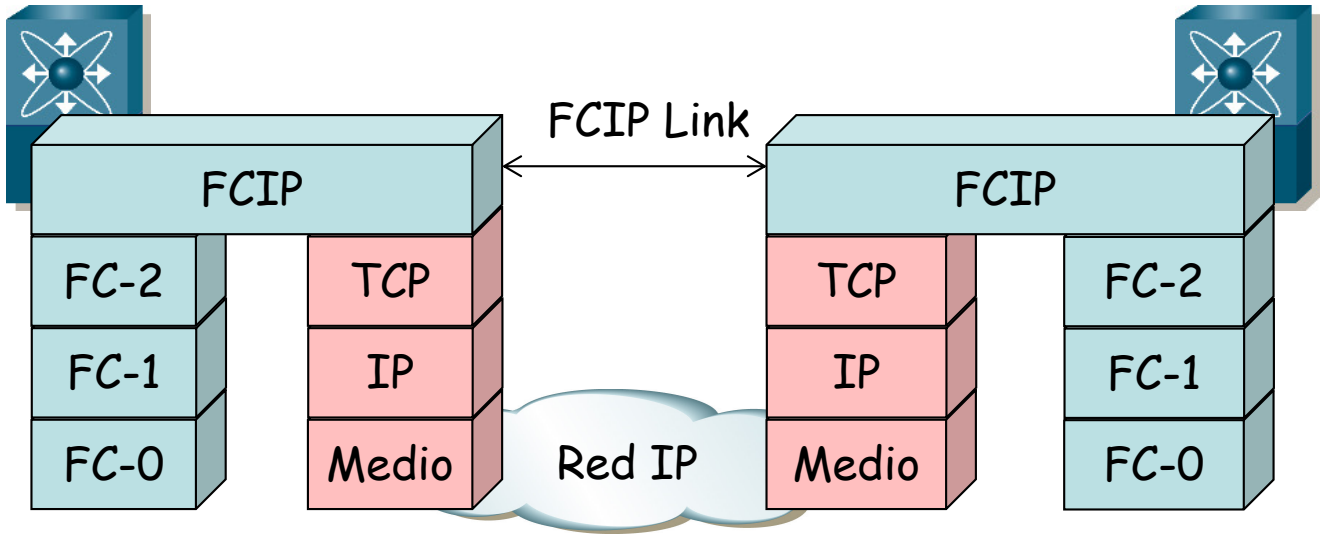
Otras soluciones

FCIP

- RFC 3821 “Fibre Channel Over TCP/IP (FCIP)”
- Es transparente para los equipos FC, solo clases 2, 3 y 4
- Interconexión de islas SAN a través de una red IP; una (o más) conexiones TCP
- Por ejemplo interconexión de SANs en diferentes DCs para replicación de datos
- Los equipos suelen permitir ajustar los parámetros de IP y TCP (timer de retransmisiones, control de flujo, control de congestión, etc)
- También diversos “hacks” para acelerar el protocolo FC

iFCP

- RFC 4172 “iFCP – A Protocol for Internet Fibre Channel Storage Networking”
- Solo clases 2 y 3



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

Arquitectura tradicional en el data center: limitaciones

DC tradicional

2 capas

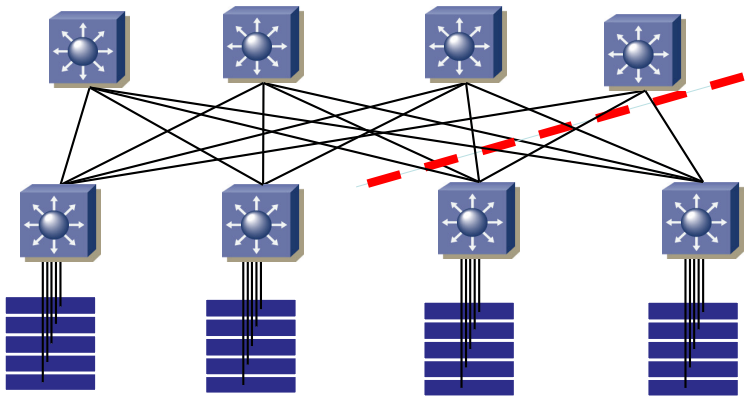
- Conmutación en dos capas: ToR y agregación
- Conmutación capa 2 en agregación si se necesita que los distintos armarios estén en la misma VLAN
- Si los armarios son servicios independientes estarán en distintas VLANs
- En un mismo armario podría haber diferentes componentes de un servicio separados por VLANs
- Podría conmutarse capa 3 en agregación o en el ToR (hoy en día *line-rate)*
- Este esquema puede ser con uno o dos ToR por armario (redundancia de conexión de servidor a switch)
- El límite de escala está en el número de puertos en agregación

3 capas

- Conectamos hosts de tal forma que su tráfico agregado excede el que se puede cursar por los enlaces externos
- Altos ratios de sobre-subscripción son razonables cuando tenemos mucho tráfico norte-sur
- No son tan razonables cuando hay mucho tráfico este-oeste
  - Tráfico entre los servidores en distinto rack
  - Aplicaciones distribuidas, tráfico de SAN, movimiento de máquinas virtuales, tráfico entre tiers de aplicación, clustering, etc

Bisectional bandwidth

- Una **bisección** es una partición de la red en dos subconjuntos con igual número de hosts
- El **ancho de banda de esa bisección** es la suma de las capacidades de los enlaces entre los dos subconjuntos
- En este ejemplo 2x capacidad del enlace de agregación a acceso
- El **ancho de banda de bisección de la red** es el menor ancho de banda de una bisección de la red que se pueda conseguir
- Cuando tenemos mucho tráfico este-oeste queremos un elevado ancho de banda de bisección
- Una topología en 2 capas nos puede dar un alto ancho de banda de bisección si hay muchos enlaces activos entre ellas
- Aumentamos el BW de bisección aumentando el número de caminos
- La conmutación es L3 pues en L2 STP no permite tener caminos alternativos
- Eso quiere decir que no podemos extender VLANs entre los armarios



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

Arquitectura tradicional en el data center: Escalabilidad

Escalabilidad

- Estamos condicionados por el número de puertos en los conmutadores
- El precio de estos equipos de alto nivel de agregación es elevado
  - CAMs/TCAMs/SRAMs muy grandes
  - Alto consumo eléctrico, componentes costosos
  - Hay que pagar el coste de diseño de este hardware/software
- Internamente:
  - Estos conmutadores son redes de conmutación
  - Es decir, están construidos a partir de chips de conmutación interconectados
  - Pueden tener sobre-subscripción interna (bloqueo interno)

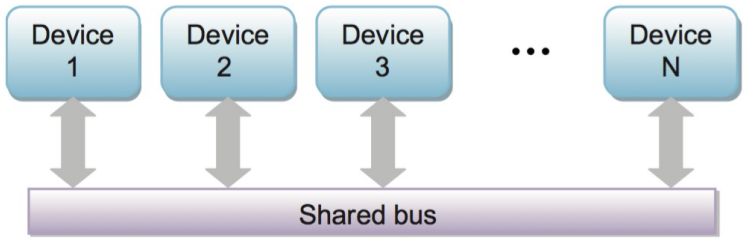
upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

Redes de Nueva Generación
Área de Ingeniería Telemática

Arquitectura de conmutadores

Shared Bus

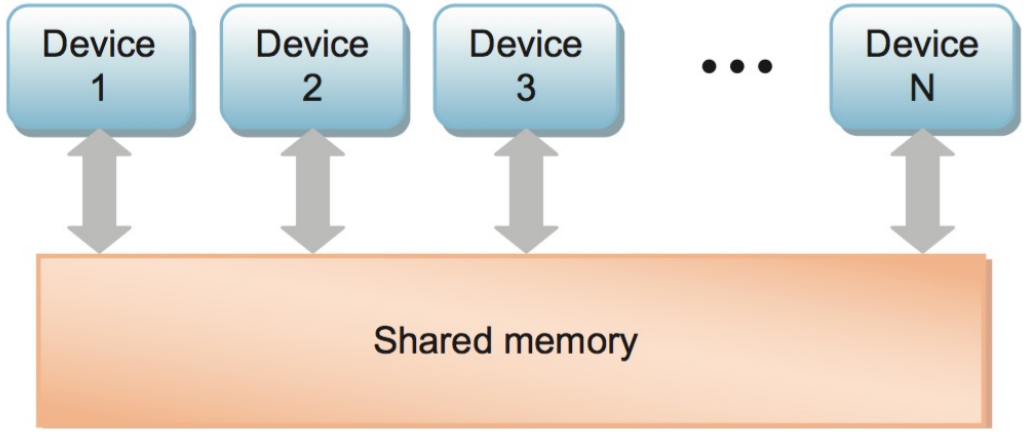
- Típico en primeros sistemas con múltiples procesadores
- Cuando hay una gran cantidad de dispositivos no es económico una malla
- Un bus interconecta en modo compartido a los dispositivos
- Más económico, sencillo añadir dispositivos
- Requiere resolver la contienda y colisiones
- Ejemplos: Ethernet (CSMA/CD), PCI, I^2C, etc
- Un bus puede consistir en múltiples señales en paralelo
- La capacidad del bus está compartida, aunque cada dispositivo debe poder enviar a la máxima velocidad del mismo
- Se puede aumentar la velocidad aumentando la anchura del bus o su frecuencia
- Aumentar la anchura aumenta el número de pines
- Aumentar la frecuencia aumenta la posibilidad de que se desfasen y de que se interfieran
- Para altas velocidades la industria se ha movido hacia interfaces serie



Gary Lee, “Cloud Networking”, Morgan Kaufmann Ed.

Shared Memory

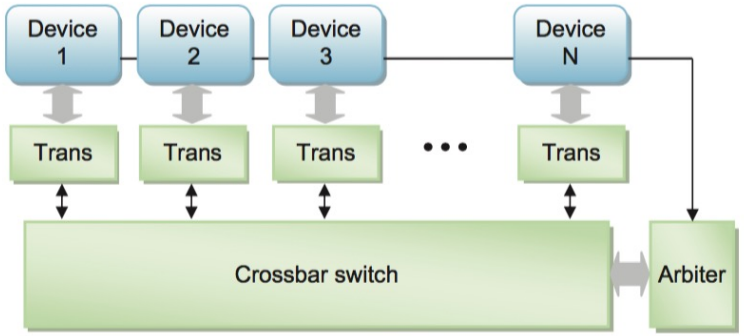
- Intercambian la información leyendo y escribiendo en una memoria compartida
- Las conexiones a la memoria son punto-a-punto y cada dispositivo puede emplear todo el ancho de banda de ellos
- Si hay N dispositivos con interfaces a una tasa R necesitamos una memoria capaz de escribir a una tasa NxR
- Con puertos de dispositivo full-duplex necesitamos ese mismo número de lecturas, así que debe soportar una tasa 2xNxR
- Si la memoria es un cuello de botella necesitamos un árbitro para el acceso a la misma
- También requiere una memoria con una gran cantidad de pines



Gary Lee, “Cloud Networking”, Morgan Kaufmann Ed.

Crossbar Switch

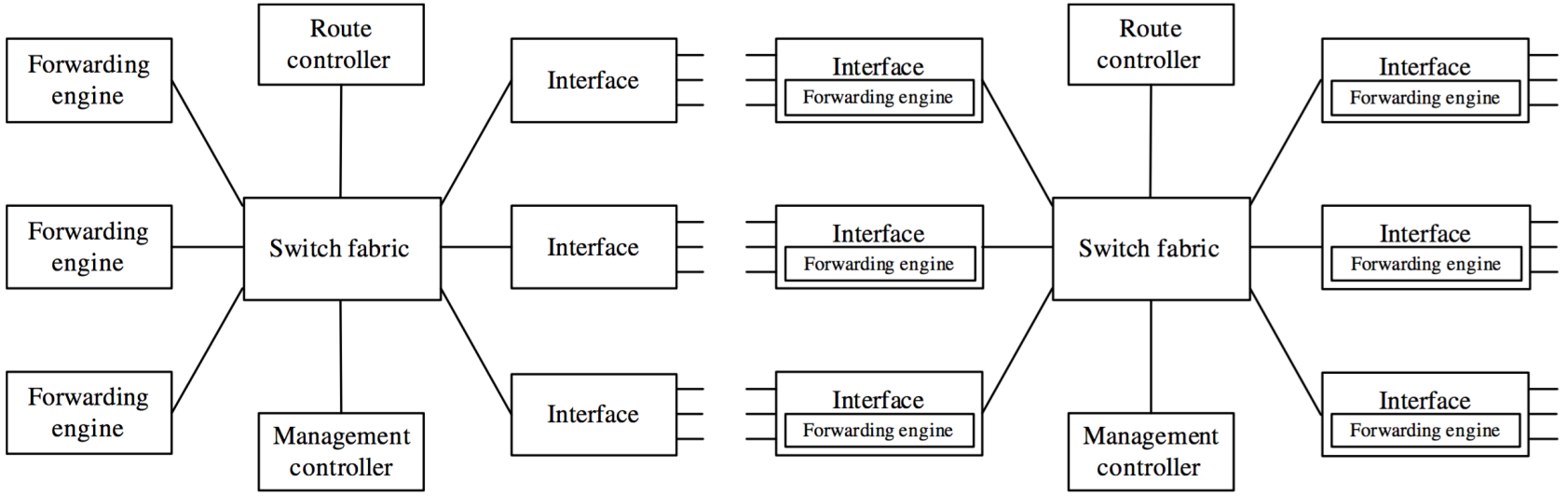
- En los 90s surgen los primeros SerDes en el rango de 1Gbps
- Esto permite emplear chips de conmutación crossbar asíncrona
- Los datos se serializan y entran por un puerto del crossbar
- Un árbitro se encarga de programar el crossbar
- Puede unir cualquier entrada con cualquier salida (una a la vez)
- No hay bloqueo interno
- No hay re-sincronización, se des-serializa a la salida
- Suelen segmentarse los datos en “celdas” de cara a gestionar la configuración temporal del crossbar
- Se requieran memorias y lógica de segmentación y reensamblado (SAR)
- Los buffers también son necesarios en caso de bloqueo externo
- El tiempo de enganche del PLL del receptor a la señal del transmisor es un tiempo desaprovechado
- A finales de los 90s se mejoran con una conmutación síncrona



Gary Lee, “Cloud Networking”, Morgan Kaufmann Ed.

Distribuido o centralizado

- Arquitectura centralizada
  - Los interfaces entregan los paquetes a los motores de reenvío
  - Éstos toman la decisión y pasan el paquete al interfaz correspondiente
  - Puede que esos interfaces sean tarjetas con múltiples puertos
  - Puede que solo se envíe al FE la cabecera y cuando toma la decisión se entregue el paquete al interfaz de destino
  - Las rutas son calculadas por otros elementos
- Arquitectura distribuida
  - Los interfaces son capaces de tomar las decisiones de reenvío
  - Las rutas calculadas se configuran en los interfaces
  - Mejora el rendimiento



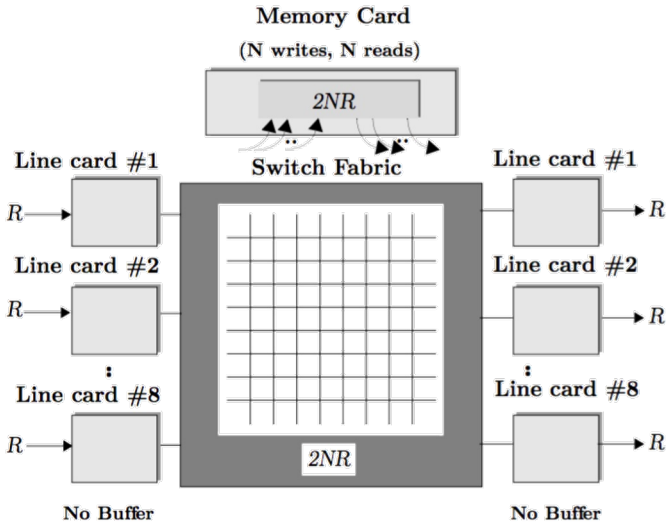
H.Jonathan Chao y Bin Liu, “High Performance Switches and Routers”, John Wiley & Sons

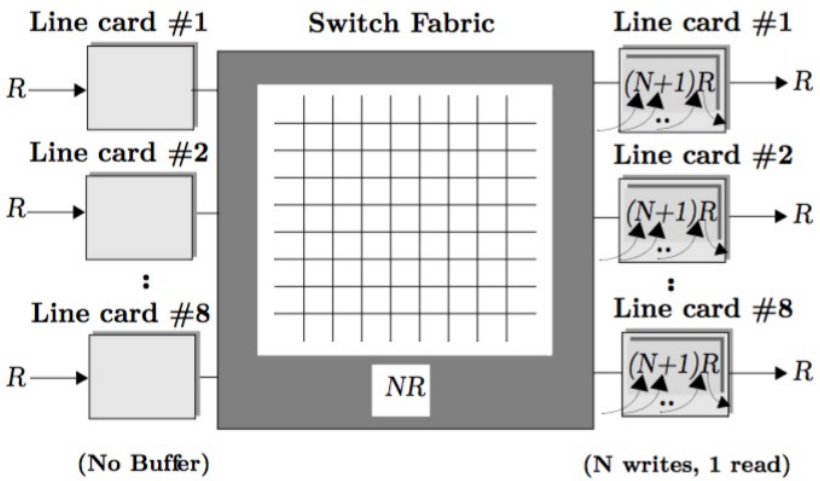
Limitaciones de la memoria

- *Content Addressable Memory*
  - En lugar de dar una dirección a la memoria se le da directamente el contenido
  - La CAM devuelve la dirección en que está almacenado
  - Más rápido que una búsqueda en RAM
  - Empleado en tablas MAC (layer 2)
  - Tamaños pequeños, elevados consumos
- *Ternary Content Addressable Memory*
  - Al almacenar bits pueden indicar que no importa el valor de algunas posiciones
  - Eso permite (si se ordenan bien las entradas) hacer búsquedas equivalentes a longest-prefix-match
  - Empleadas en tablas de rutas IP (MPLS, QoS, ACLs, etc) (layer 3)
  - Tamaños pequeños, elevados consumos
- DRAM
  - Diseñada pensando en el menor coste por byte y aumentar la capacidad
  - Tiempos de acceso en las decenas de nanosegundos
- SRAM
  - Diseñada pensando en el menor tiempo de acceso
  - Tiempos de acceso en los ns (un orden de magnitud inferior)
  - Más cara (mayor nº de transistores por bit)

Output Queueing

- El conmutador con memoria compartida centralizada es un ejemplo de conmutador con colas a la salida
- Puede implementarse con una memoria por cada salida
- Con N entradas una de estas memorias debe permitir en el peor caso N escrituras y 1 lectura por slot, o una tasa (N+1)xR
- Esto reduce a la mitad los requerimientos pero fragmenta su uso
- En el peor caso tramas de 64 bytes, a 10Gbps eso es una trama cada 67ns
- En el tiempo que tarda en llegar otra trama se pueden hacer muy pocos accesos a memoria
- Necesita por ejemplo memorias que hagan *pre-fetching o* con menor tiempo de respuesta (SRAM, memoria on-chip)
- ¿Y si tenemos un conmutador con varios puertos 10G?
- Una “solución” es poner varias memorias en paralelo

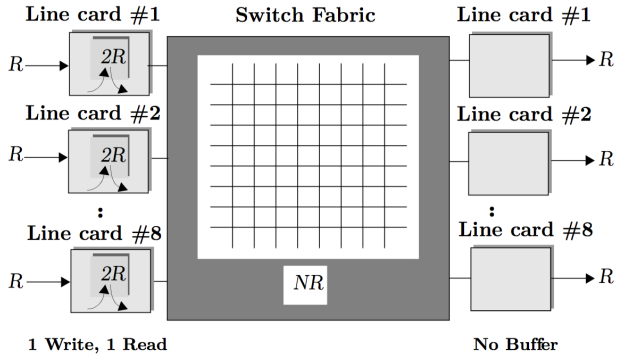


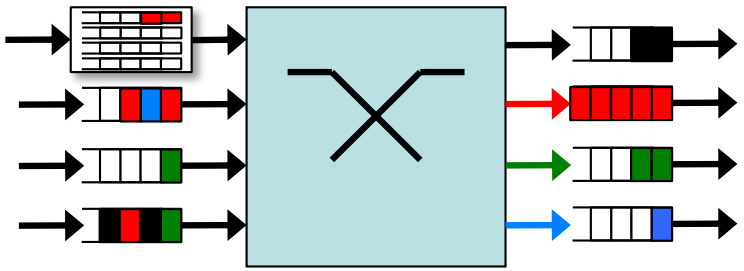


Sundar Iyer, “Load Balancing and Parallelism for the Internet”, PhD Tesis, Stanford, 2008

Input Queueing

- Los paquetes se almacenan en los dispositivos de entrada
- La memoria pasa a estar distribuida y un crossbar switch interconecta los dispositivos
- La memoria tiene requisitos más ligeros pues no depende del número de dispositivos
- No requiere memorias tan rápidas; memorias distribuidas entre los interfaces de entrada
- Colas FIFOy distribuye también las tablas para el *lookup*
- *Head of Line blocking*
  - Degrada el rendimiento
  - Con tráfico distribuido uniformemente por las salidas está limitado a un throughput del 58.6% [1]
- Se elimina el HoL empleando *Virtual Output Queues (VOQs)*
- Cada buffer de puerto de entrada separa el tráfico en tantas VOQs como puertos de salida
- Aunque una VOQ no pueda ser servida porque el puerto de salida esté ocupado se puede servir de otra
- El algoritmo de planificación de la conmutación interna se complica
- En un intervalo de tiempo (*slotted*) solo atravesaría la matriz un paquete para cada puerto
- Pueden atravesar la matriz varios en una ranura de tiempo si trabaja a una velocidad superior a los puertos
- Es un *speedup* (xN), pero entonces requiere buffers a la salida, si no son puertos de velocidad superior
- *Combined Input and Output Queueing (CIOQ):* Input queueing + VOQ
- Está comprobado que para prácticamente cualquier tipo de tráfico se comporta aproximadamente como un FIFO-OQ con tal de que el *speedup* sea de al menos x2





Sundar Iyer, "Load Balancing and Parallelism for the Internet", PhD Tesis, Stanford, 2008

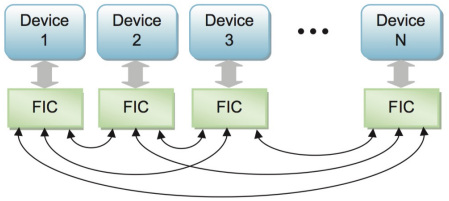
[1] Mark J. Karol, Michael G. Hluchyj, Samuel P. Morgan, "Input Versus Output Queueing on a Space-Division Packet Switch", IEEE Trans. On Comm., Vol. COM-35, No. 12, Dic. 1987

Topologías

- Un data center requiere decenas de miles de puertos
- Un chip puede ofrecer en el orden de decenas, tal vez un centenar
- Quedan entonces las soluciones multi-etapa
  - Soluciones multi-conmutador
  - Y soluciones multi-chip dentro del conmutador

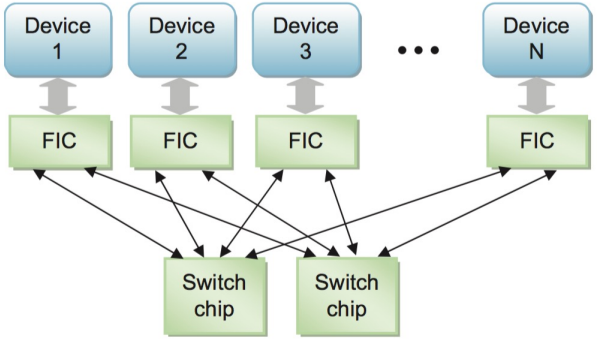
Anillos

- Elementos sencillos (2 puertos) y ofrece protección
- Fácil añadir nuevos dispositivos aumentando el anillo
- Pero al aumentar los dispositivos aumenta la carga



Mesh

- Enlace dedicado entre cada pareja de dispositivos
- Mejora el rendimiento
- Requiere gran cantidad de puertos por dispositivo
- Requiere más decisión de encaminamiento que el anillo

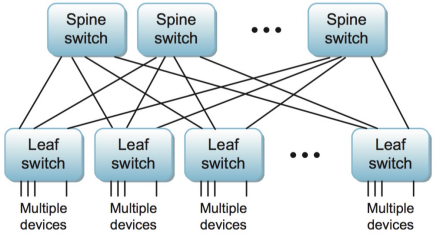


Estrella

- Es la topología típica en conmutadores modulares
- El chasis contiene elementos de conmutación

Fat Tree

- Evoluciona de la topología en estrella con más nodos centrales
- En realidad esta topología es muy conocida en el entorno de conmuta



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

Alternativas a STP

STP

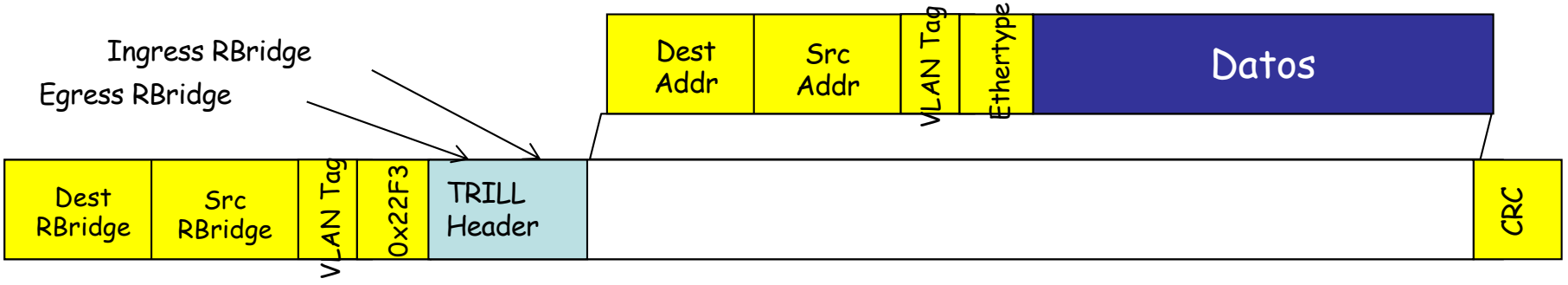
- Limitaciones de STP
  - No soporta multipath para una VLAN
  - Multipath entre diferentes VLANs requiere intervención manual
  - El camino es el más corto solo desde la perspectiva del root
  - Largos tiempos de convergencia y peligro de tormentas de inundación
  - Elección de la raíz no es segura
- Mejoras a STP
  - RSTP, MSTP mejoran los tiempos de convergencia pero siguen en el rango de los segundos
  - Hay otras mejoras a la convergencia, muchas veces sin estandarizar (*Loopguard, BPDU guard, Rootguard, BPDU filter, Storm control*)
  - No cambian que STP desactiva puertos para formar un árbol
- Alternativas clásicas a STP
  - Mutichassis LAG:
    - Limitado a dominios en el orden de los miles de hosts (es razonable tener más?)
  - Routing capa 3
    - Conmutación capa 3 no ofrece continuidad en capa 2
    - Es problema para ciertas aplicaciones, en especial con función de clustering
    - Por ejemplo no permite la movilidad de las VMs

TRILL

- *Transparent Interconnection of Lots of Links*
- IETF, RFC 6325 (Perlman 2011) y otras
- Pretende sustituir a STP
- El conmutador que lo implementa se conoce como un RBridge (Routing Bridge)
- Lo básico
  - Los RBriges y los enlaces o LANs puenteadas que los interconectan forman un “campus” (el dominio de broadcast)
  - Transportan las tramas Ethernet por ese campus encapsulándolas en otras tramas Ethernet (MAC in MAC)
  - Esa cabecera adicional incluye una cuenta de saltos
  - El camino por el campus lo calcula IS-IS (permite ECMP)
  - Se desencapsula en el RBridge de salida hacia el destino
- Está especificado su transporte sobre Ethernet y sobre PPP

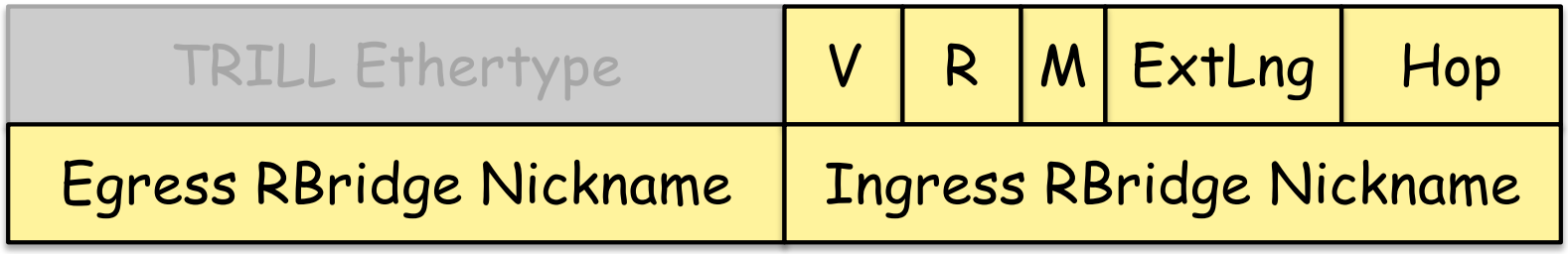
TRILL sobre Ethernet

- MAC origen y destino son de RBridges adyacentes
- Puede llevar etiqueta de VLAN si los conmutadores del campus TRILL la necesitan (Ethertype 0x22F3)
- TRILL añade su propia cabecera
- A continuación la trama que ha llegado al RBridge frontera
- Si la trama original no llevaba etiqueta de VLAN se le añade
- Los conmutadores del Campus TRILL no RBridges van a reenviar en base a las direcciones de la cabecera exterior
- RBridges reenvían en función del RBridge Nickname destino
- En cada salto entre RBridges las direcciones MAC más exteriores son de los RBridges que envían y reciben esa trama
- “Dest RBridge” es la del siguiente salto, “Src RBridge” es la del anterior
- Parecido a que los RBridges fueran routers
- Los RBridges frontera (entrada a la campus y salida) están indicados en la cabecera de TRILL



TRILL Header

- Nicknames
  - Cada RBridge posee un *nickname* con el que se le hace referencia en las PDUs de TRILL y sirve para identificarlo de cara a IS-IS
  - Los nicknames son números de 2 bytes (máx. 64K RBridges)
  - Los nicknames se eligen mediante un proceso automático con información añadida a los mensajes de IS-IS
- Campos de la cabecera
  - V = Version (2 bits), R = Reserved (2 bits)
  - M = Multi-Destination (1 bit)
  - ExtLng = Length of TRILL Header Extensions
  - Hop = Hop Limit (6 bits)
  - Egress RBridge Nickname = nickname del RBridge de salida del campus hacia el host destino
  - Ingress RBridge Nickname = nickname del RBridge de entrada al campus de la trama desde el host origen



TRILL control path

- IS-IS directamente sobre el nivel de enlace (Ethertype 0x22F4)
- Mecanismos añadidos específicos para TRILL
- RBridge frontera aprende direcciones MAC de hosts remotos junto con:
  - RBridge por el que acceden al campus
  - RBridge siguiente salto hacia ese egress RBridge
- Lo hace principalmente en base a los paquetes de TRILL que recibe
- Solo RBridges frontera aprenden direcciones MAC de los hosts
- Pueden aprender también mediante ESADI (opcional)
  - *End-Station Address Distribution Information*
  - Un RBridge puede anunciar MACs de hosts a otros RBridges
  - Se transporta en tramas TRILL
- BUM
  - RBridges construyen árboles para las tramas multicast (bidireccionales)
  - Calcula múltiples, lo cual le permite multipath también para el multicast
  - Lo hace con el mismo IS-IS (no hace falta otro protocolo)
  - Cada árbol incluye todos los RBridges del campus y las VLANs
  - Puede hacer *pruning*
  - El Nickname del egress RBridge especifica el árbol
  - Los nodos hacen una comprobación de RPF

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

Nuevas versiones de Ethernet para LAN y WAN

Provider Bridges

- El objetivo es que un proveedor pueda transportar tráfico Ethernet mediante una MAN/WAN Ethernet
- Con 802.1Q el proveedor puede emplear tags para diferenciar usuarios
- Pero esto impide transportar tráfico de usuario *tagged*

802.1ad (modificación a 802.1Q-2005): Provider Bridges

- Permite diferenciar entre las VLANs del cliente (C-VLAN) y las del servicio (S-VLAN)
- Con 802.1ad los puentes del proveedor ven gran número de direcciones MAC
- Solo permite 4094 clientes simultáneos

802.1ah (modifica 802.1Q-2005): Provider Backbone Bridges

- Posibilita conectar PBNs (*Provider Bridged Networks*) a través de una PBBN (*Provider Backbone Bridged Network*)
- Se define el Backbone Service Instance Tag (I-TAG)
- Se encapsula la trama Ethernet 802.1ad dentro de otra:
  - B-TAG: *Backbone VLAN tag* (idéntico a un S-TAG)
  - I-SID: *Backbone Service Identifier* (24 bits)
  - C-DA: *Encapsulated Customer Destination Address*. La dirección MAC destino de la trama encapsulada (Client Dest Addr en la figura)
  - C-SA: *Encapsulated Customer Source Address*. La dirección MAC origen de la trama encapsulada (Client Src Addr en la figura)
- Direcciones MAC origen y destino son de los equipos frontera de la PBBN
- Los conmutadores de la PBBN NO ven las direcciones MAC de los equipos de cliente (van encapsuladas)

Traffic Engineering

- 802.1Qay-2009 “Provider Backbone Bridge Traffic Engineering”, amendment a 802.1Q-2005
- Define funcionamiento de Ethernet *orientado a conexión* usando trama PBB (PBB-TE)
- Para ello se crearían *Ethernet Switched Paths* (ESPs) desde el plano de gestión (un agente externo configura switches)
- Un ESP es como un LSP, es también unidireccional
- El camino viene identificado por las direcciones origen y destino del ESP y el identificador de VLAN del B-TAG (serían la etiqueta)
- La etiqueta NO cambia en el camino, manteniendo el funcionamiento del plano de datos de Ethernet
- El ESP puede ser punto-a-punto o punto-a-multipunto (entonces la dirección destino es una dirección MAC de grupo)
- Desactiva el *learning* para esos VIDs y descarta tramas en esa VLAN con destino desconocido
- Posibilita servicios tipo *Carrier Ethernet (http://metroethernetforum.org)*
- 802.1Q-2018 tiene ya 1993 páginas

SPB

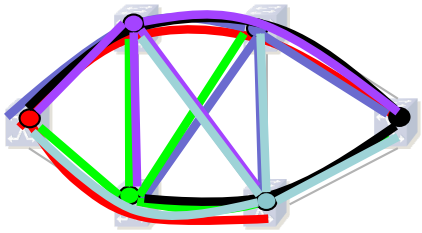
- *Shortest Path Bridging* (802.1aq, ya en 802.1Q-2014)

Objetivos

- Busca escalar la capa 2 al orden del millar de puentes
- Busca emplear múltiples caminos de igual coste
- Evitar aprendizaje de direcciones MAC donde sea posible
- Tiempos de recuperación en los centenares de milisegundos
- Reutiliza ASICs existentes
- Mantener orden y simetría en los caminos (para una VLAN)
- Compatibilidad con el resto de 802.1 (DCB, OA&M, etc)

Control plane

- Sustituye {R|M}STP, pudiendo interoperar con ellos
- Para una VLAN ahora tenemos (al menos) 3 posibilidades:
  - Emplear el Internal Spanning Tree (IST)
  - Emplear una Multiple Spanning Tree Instance (MSTI)
  - Emplear un “SPT set” (set of Shortest Path Trees)
- Cada SPT (Shortest Path Tree) del SPT set tiene como raíz del árbol a un puente diferente del dominio



SPB: Control plane

- En el plano de control emplea IS-IS con algunas extensiones (ISIS-SPB)
- ISIS-SPB mantiene al menos un SPT para cada puente con él como raíz
- Un puente envía tramas solo por uno de esos árboles
- Calcula caminos deterministas y simétricos
- Si hay varios caminos de coste mínimo solo se emplea uno de ellos en un SPT set
- Todos los nodos emplean la misma técnica de desempate para lograr caminos simétricos
- Al calcular SPTs se está distribuyendo la carga más que con un solo árbol de expansión
- Además puede calcular múltiples Equal Cost Trees (ECTs)
- Cada ECT resulta de un algoritmo de desempate (16 algoritmos, luego hasta 16 ECTs)
- SPTs resultado de un tipo de desempate forman un “SPT set”
- El balanceo de carga se hace distribuyendo VLANs por ellos
- No hace balanceo a nivel de paquete ni de flujo
- Loops pueden producirse temporalmente cuando las bases de datos de IS-IS no están sincronizadas
- Emplea RPF (Reverse Path Forwarding) para intentar evitarlos
- También emplea técnicas para evitar tener un loop mientras converge IS-IS
- Pero sigue sin haber TTL

SPB: Modos

- Todos los puentes emplean el mismo modo para una VLAN concreta

SPBM: *Shortest Path Bridging MAC mode*

- Emplea encapsulado 802.1ah (MAC-in-MAC)
- Transporta transparentemente las VLANs de usuario
- El I-SID permite distinguir una gran cantidad de servicios
- Las tramas se transportan solo entre puertos que mapeen al mismo I-SID
- Cada I-SID se mapea a un B-VID (VLAN del backbone)
- La pareja de B-VID y dirección origen del backbone identifican al SPT
- Solo los puentes frontera aprenden MACs de usuarios
- Si no tienen aprendida una dirección de usuario harán multicast
- Los puentes del backbone solo ven las MACs frontera
- Aprenden cómo llegar a esas direcciones mediante ISIS-SPB
- Es decir, el aprendizaje típico de puente se puede desactivar
- Hace una búsqueda de la dirección MAC destino para averiguar a qué puente frontera del Backbone mandar la trama
- BUM
  - Replicación en el nodo frontera y unicast o
  - Construcción de árboles multicast con raíz en cada nodo
- Unicast y multicast sigue el mismo camino
- En caso de árboles para el multicast emplea una dirección MAC origen específica para el grupo
  - SPSourceID identifica al puente
  - I-SID

SPB: Modos

SPBV: *Shortest Path Bridging VID*

- Requiere menos configuración que SPBM
- Un nodo frontera recibe tramas de una VLAN
- Calcula un SPVID (*Shortest Path VLAN Identifier*) en base al VID y el nodo de entrada (un SPVID para cada pareja)
- Se envía la trama con el SPVID como VID
- El cambio se deshace en el nodo de salida
- El mapeo se sincroniza entre los nodos empleando ISIS-SPB
- Es decir, en el backbone se emplea una VLAN para cada SPT
- Eso limita a (nºVLANs_en_backbone x nºNodos) < 4095
- El aprendizaje de bases de datos de filtrado en el backbone es compartido entre todos los SPVID de la misma VLAN de usuario
- Si no puede asignar un SPVID no cambia el VID y emplea el IST
- Si la trama original no tiene encapsulado de VLAN se añade el SPVID con 802.1Q
- Se retira a la salida
- Si tiene encapsulado 802.1Q se modifica el V-TAG
- No separa el direccionamiento de espacio del usuario del empleado en el backbone
- Si la trama original tiene encapsulado 802.1ad (Q-in-Q) se modifica el S-TAG con el SPVID

ECMP 802.1Qbp

- 802.1aq daba tres posibilidades:
  - Emplear el Internal Spanning Tree (IST)
  - Emplear una Multiple Spanning Tree Instance (MSTI)
  - Emplear un “SPT set” (set of Shortest Path Trees)
- 802.1Qbp añade dos posibilidades más
  - Emplear un conjunto de *Traffic Engineered Service Instances* (TESIs)
  - Emplear múltiples caminos de igual coste (hasta 16) que reparten el tráfico desde cada puente (ECMP)
- La solución ECMP solo se puede emplear en modo SPBM
- Permite repartir el tráfico de una misma VLAN
- Deja de garantizar que unicast y multicast sigan el mismo camino
- Añade la posibilidad de un Flow-Tag (F-TAG), que contiene un TTL
- Ese campo contiene un identificador de flujo
- Puede hacer el balanceo en base a ese identificador (las tramas con el mismo valor siguen siempre el mismo árbol)
- De esa forma mantiene el orden en las tramas de un flujo

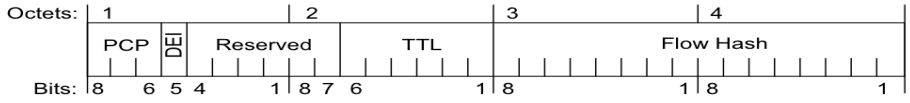
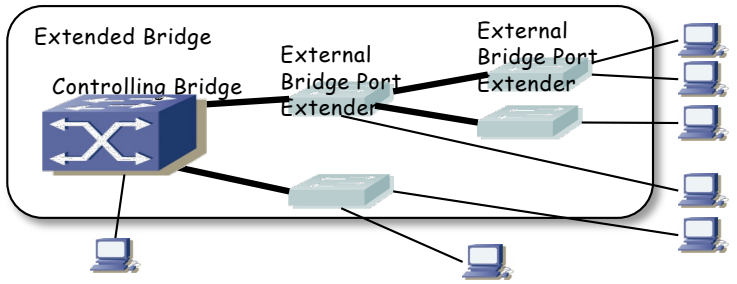


Figure 44-8—Flow FilteringTCI format

Otras modificaciones a 802.1Q

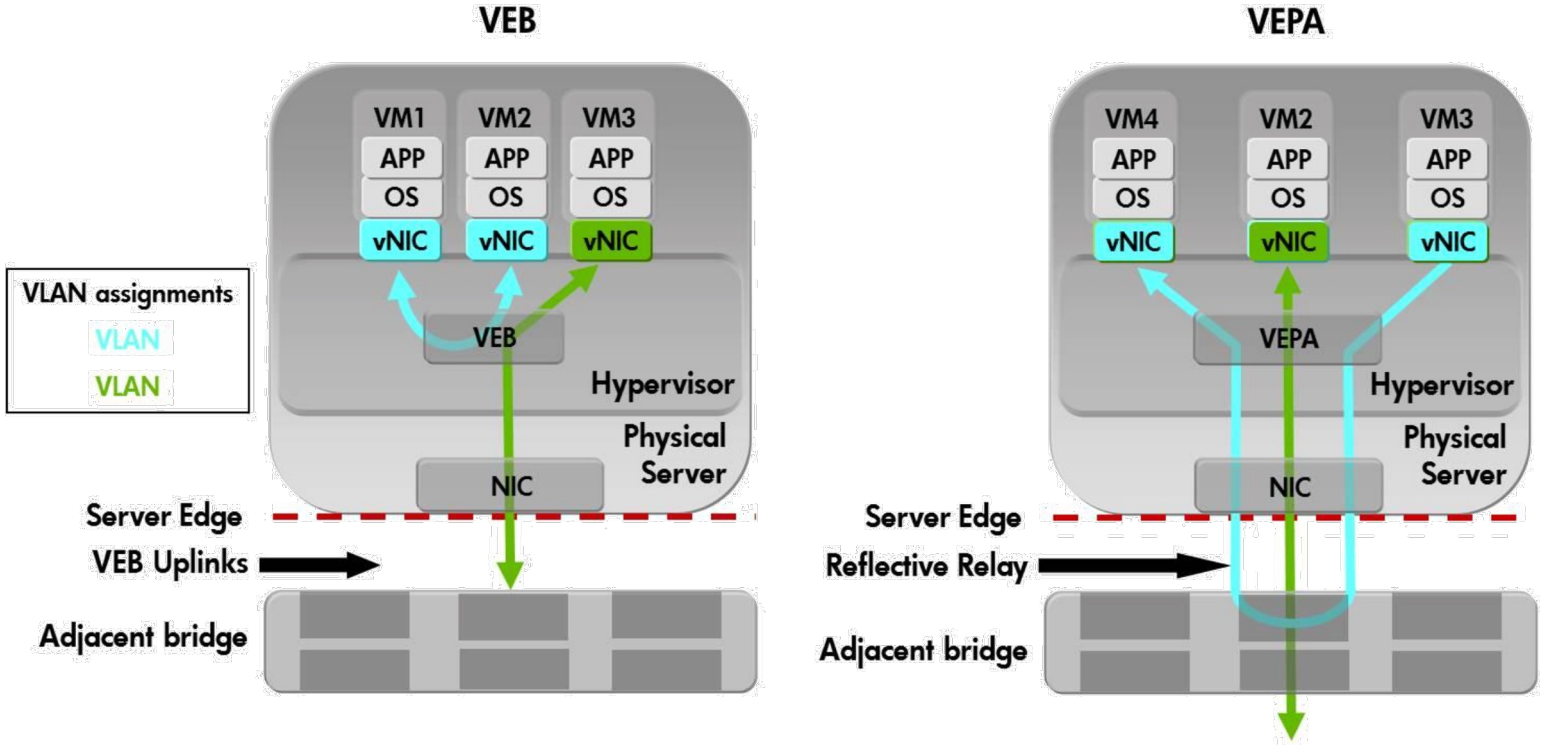
802.1BR Extended Bridge

- 802.1BR-2012 “Bridge Port Extension”
- Inicialmente 802.1Qbh (modificación a 802.1Q) pero lo cancelaron y lo movieron a un estándar independiente
- Introduce el *Extended Bridge*, que es simplemente un puente 802.1Q
- Está formado por *Controlling Bridge* y uno o más *Bridge Port Extenders*
- Forman un árbol con el Controlling Bridge como raíz (no STP interno)
- Todo ello forma un puente 802.1Q (el Extended Bridge)
- Todo el extended bridge se gestiona como una unidad
- Todos los puertos de los PE tienen su imagen en puertos internos del Controlling Bridge
- El camino bidireccional desde el puerto externo al interno en el Controlling Bridge se llama un *E-channel*
- El tráfico se conmuta en el Controlling Bridge
- Un E-channel se identifica por un E-channel Identifier (E-CID)
- A una trama que entra por un Extended Port se le añade un E-Tag
- Entre otros aspectos permite identificar al puerto por el que entró la trama (de hecho al E-channel)



802.1Qbg EVB

- 802.1Qbg-2012 “Edge Virtual Bridging”, ya en 802.1Q-2014
- El hypervisor además de hacer de VEB (*Virtual Edge Bridge*) puede hacer de VEPA (*Virtual Edge Port Aggregator*)
- Se busca que el switch físico se encargue de conmutar el tráfico incluso entre las VMs del mismo host (resulta en mayor utilización de los enlaces)
- Es decir, el VEPA no conmuta internamente sino que manda al exterior
- El switch físico suele poder implementar más políticas que los switches software
- ¡ El switch debe reenviar tráfico por el mismo puerto por el cual le llegó ! (*reflective relay* o *hairpin turns*)



802.1BA AVB

- Audio Video Bridging (AVB) Systems
- *“Profiles that select features, options, configurations, defaults, protocols and procedures of bridges, stations and LANs that are necessary to build networks that are capable of transporting time-sensitive audio and/or video data streams are defined in this standard.”*

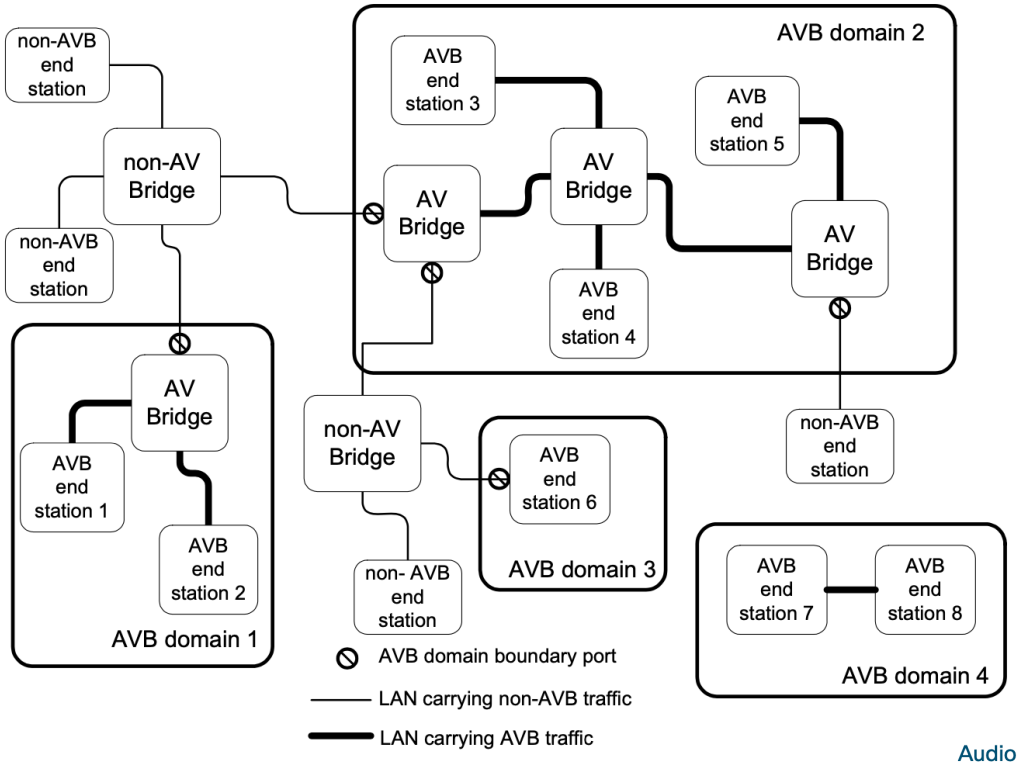


Figure 5-2—AVB domain boundaries created by non-AVB systems

Audio Video Bridging

With Cisco IOS® XE Software Release 16.3, select* Cisco Catalyst 3650 Series models support the IEEE 802.1 AVB standard. This standard provides the means for highly reliable delivery of low-latency, time-synchronized AV streaming services through Layer 2 Ethernet networks. The standard also makes it easier to integrate new services and for AV equipment from different vendors to interoperate. Whether the AV endpoint connections are analog or are inflexible digital one to one, the network transport enables many-to-many transparent plug-and-play connections for multiple AV endpoints.

Modificaciones a 802.1Q

802.1Qca-2015

- *"Path Control and Reservation"*, Amendment 24 a 802.1Q-2014
- *"These extensions involve explicit path control, bandwidth reservation, and redundancy (protection, restoration) for data flows. Thus, this standard specifies bridging on explicit paths for unicast and multicast frames, specifying protocols to determine multiple active topologies."*
- Extensiones a IS-IS: *ISIS-PCR (Path Control and Reservation)*
- Emplea el "Path Control Element" (PCE) definido por el IETF PCE WG
- El PCE es un puente o un host

802.1Qbv-2015

- *"Enhancements for Scheduled Traffic",* Amendment 25 a 802.1Q-2014
- *"Some applications have a need for frame* ***delivery that is highly predictable*** *in terms of the time at which frame transmission will occur, and the overall latency and jitter ..."*
- *Examples include* ***industrial and automotive control applications****, where data transmitted over the network is used to feed the parameters of control loops that are critical to the operation of the plant or machinery involved, ...*
- *In some implementations, this need has been met by the provision of dedicated, highly engineered networks [...] however, [...] it can be desirable to mix time-critical traffic with other classes of traffic in the same network, ...*
- *... ensure that, at specific times, only one traffic class (or set of traffic classes) has access to the network; in effect to create a protected "channel" that is used by that traffic class alone.*

Modificaciones a 802.1Q

802.1Qbu-2016

- *"Frame Preemption"*, Amendment 26 a 802.1Q-2014
- *Enhancements to the forwarding process that support frame preemption*
- ***preemption**: The suspension of the transmission of a preemptable frame to allow one or more express frames to be transmitted before transmission of the preemptable frame is resumed.*
- *eMAC = express MAC; pMAC = preemptable MAC*
- Acompña a 802.3br-2015

802.1Qch-2017

- *"Cyclic Queuing and Forwarding"*, Amendment 29 a 802.1Q-2014
- *"Cyclic queuing and forwarding (CQF) is a method of **traffic shaping** that can deliver deterministic, and easily calculated, latency for time-sensitive traffic streams.*
- *... stream traffic is transmitted and queued for transmission along a network path in a cyclic manner.*
- ***Time is divided into numbered time intervals** i, i+1, i+2, ... i+N, each of duration d.*
- *Frames transmitted by a Bridge, Alice, during time interval i are **received** by a downstream Bridge, Bob, **during time interval i** and **are transmitted** onwards by Bob towards Bridge Charlie **during time interval i+1**, and so on.*
- *A starting assumption is that, for a given traffic class, all Bridges and all end stations connected to a given bridge have a common understanding (to a known accuracy) of the start time of cycle i, and the cycle duration, d."*

Modificaciones a 802.1Q

802.1Qci-2017

- *"Per-Stream Filtering and Policing"*, Amendment 28 a 802.1Q-2014
- *"Enhancements to the forwarding process that support per-stream filtering and policing"*

802.1Qcc-2018

- *"Stream Reservation Protocol (SRP) Enhancements and Performance Improvements"*, Amendment 31 8 802.1Q-2014
- *"...specifies enhancements to protocols, procedures, and managed objects for the configuration of network resources for time-sensitive (i.e., bounded latency) applications. "*
- *"The enhancements address Time-Sensitive Networking (TSN) application requirements beyond audio/video (AV) traffic. "*

802.1Qcr-2020 *"Asynchronous Traffic Shaping"*

- *"...specifies procedures and managed objects for Bridges and end stations to perform Asynchronous Traffic Shaping over full-duplex links with constant bit data rates."*
- *"Asynchronous Traffic Shaping can be modeled as a layer of shaped first-in-first-out (FIFO) queues that are merged into per traffic class FIFO queues in transmission ports."*
- *"Additionally, this amendment provides an informative framework for worst-case delay analysis in static networks with static configurations."*

802.1Qcz-2023 *"Congestion Isolation"*

- *"... support the isolation of congesting data flows within data center environments."*
- *"... identify flows creating congestion, adjust transmission selection for packets of those flows, and signal to neighbors"*
- *"...reduces head-of-line blocking for non-congesting flows sharing a traffic class in lossless networks.*
- *"Congestion Isolation is intended to be used with higher layer protocols that utilize end-to-end congestion control in order to reduce packet loss and latency."*

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

Cloud

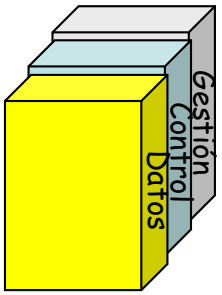
Cloud

- *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

Service Models

- Infrastructure as a Service (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
- Platform as a Service (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*
- Software as a Service (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

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
- Management plane

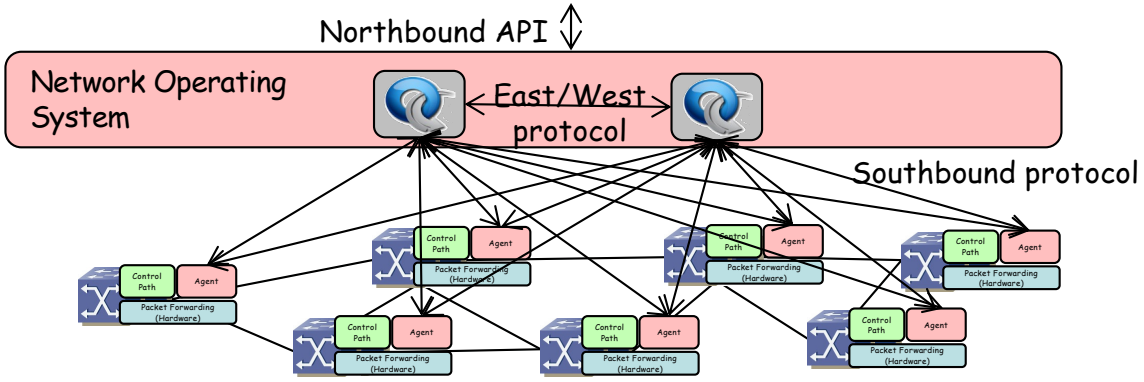
- En Internet el desarrollo se hizo basado en un control distribuido
- Ambos están en el mismo equipo, implementados por el fabricante
- *Software Defined Networking* (SDN) se basa en la separación de ambos y comunicación mediante un interfaz abierto
- La propuesta del Open Networking Forum (ONF) es OpenFlow
- 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
- 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

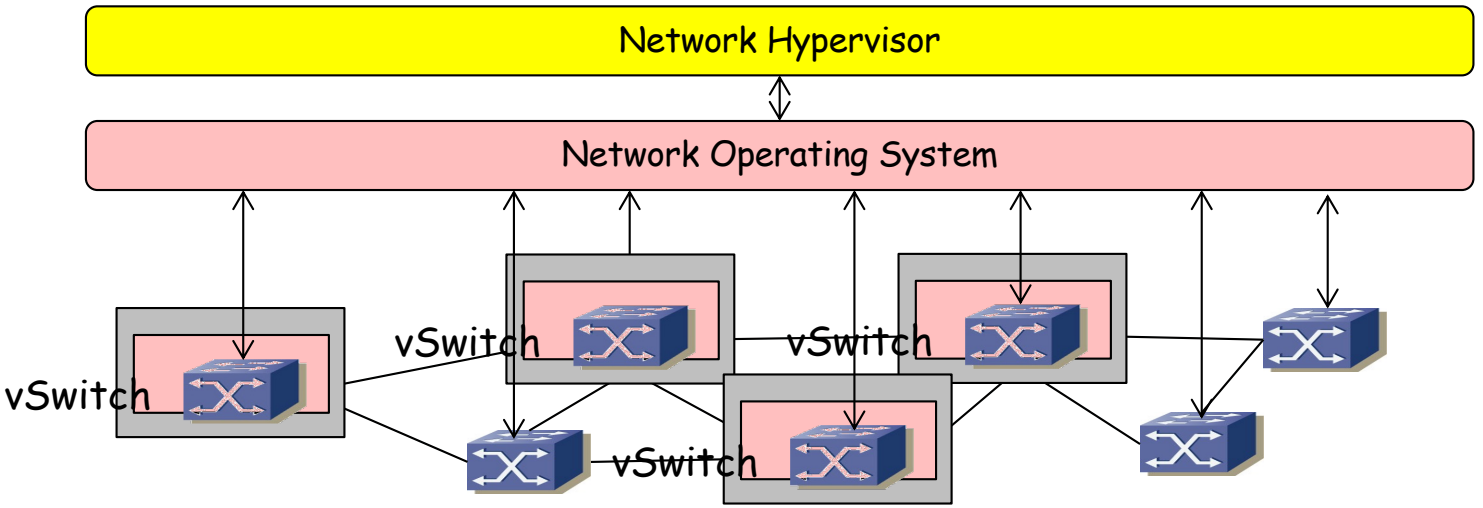
Network Operating System (NOS)

- 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
- 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*



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*
- ¿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



upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

Redes de Nueva Generación
Área de Ingeniería Telemática

OpenFlow

OpenFlow

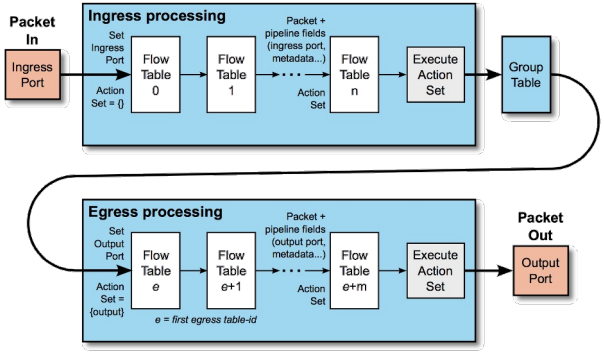
- Su origen en proyectos de investigación en la Universidad de Stanford
- En 2011 se funda el consorcio ONF (*Open Networking Foundation)*
- OpenFlow es un protocolo “southbound”; no hace “nada” sin una aplicación que lo emplee
- Dos tipos de conmutadores:
  - *OpenFlow-only*: solo soportan el modo de funcionamiento OpenFlow
  - *OpenFlow-hybrid*: también soportan funcionamiento “normal” (conmutación L2, conmutación L3, VLANs, ACLs, etc)

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
- “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

Pipeline

- Debe tener al menos una tabla aunque pueden ser más
- 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



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

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

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

Instructions:

- 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)
- Las hay de implementacion requerida y opcional

Timeouts:

- Máximo tiempo inactiva antes de expirar

***Cookie*:**

- Ahí el controlador puede guardar un valor
- El switch no lo emplea para nada

***Flags*:**

- Que envíe un mensaje al controlador al eliminarse o expirar una entrada, que no lleve contadores de bytes o de paquetes, etc

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)
    - Envío al controlador de un paquete recibido
    - Notificación de entrada en tabla eliminada
    - Notificación de cambio de estado de un puerto
  - 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://opennetworking.org/software-defined-standards/specifications/
- 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
- OF 1.2
  - Soporte de campos de IPv6, ICMPv6, ND
  - Mejora la extensibilidad de las reglas de *match*
- OF 1.3.x
  - *Meters* por flujo (limitadores para QoS); soporte de PBB
- OF 1.4
  - Mayor extensibilidad
  - Soporte de puertos ópticos (frecuencias, potencia, etc)
- OF 1.5
  - *Egress tables; s*oporte para más que Ethernet; flags TCP

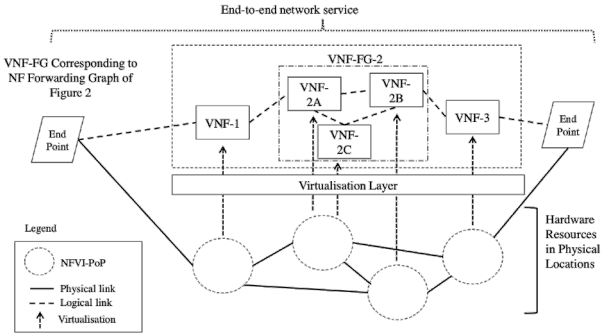
upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

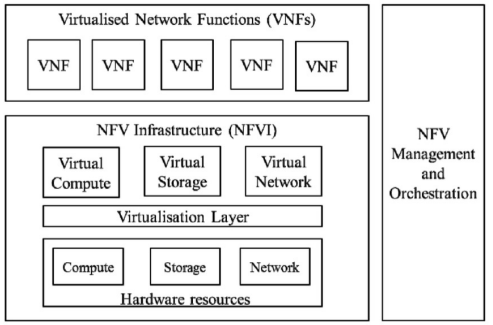
Redes de Nueva Generación
Área de Ingeniería Telemática

NFV

NFV

- Problema de las operadoras
  - Gran cantidad de *appliances*
  - Desplegar un nuevo servicio requiere espacio y alimentación para ese nuevo hardware
  - Nuevas habilidades de la gente para diseñar, integrar y operar el servicio con ese nuevo hardware
  - Ese hardware alcanza su límite de vida con rapidez, lo cual requiere políticas de remplazo que no crean nuevo beneficio
  - Los operadores declaran no estar incrementando sus beneficios pero aumentan sus costes (más tráfico, más servicios)
- NFV
  - *Network Functions Virtualisation* (complementario a SDN)
  - Se busca mover de hardware dedicado a máquinas virtuales
  - Un ISG (*Industry Specification Group*) de ETSI desde finales de 2012
  - Hoy más de 200 compañías
  - NF = Network Function
  - VNF = Virtualised Network Function (implementación de NF)
  - NFVI = Network Functions Virtualisation Infrastructure (donde se despliegan VNFs)





Facilitadores

- *Cloud Computing*
  - Virtualización (hypervisores, vSwitch, smart NICs)
  - *Orchestration*
  - Open APIs

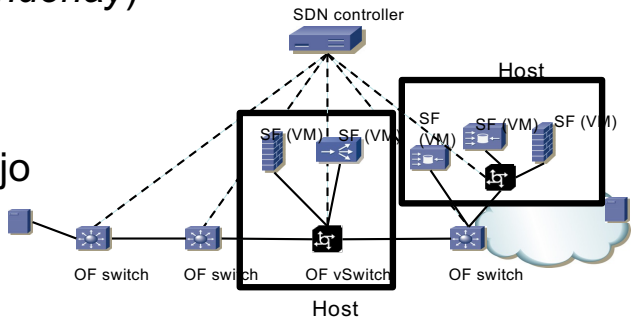
- Grandes volúmenes de servidores
  - Componentes estándar (por ejemplo x86), vendidos por millones (escala) e intercambiables (competencia)
  - En lugar de *appliances* que dependen de ASICs

Service Function Chaining

- RFC 7665: “Service Function Chaining (SFC) Architecture”
- Service functions: firewalls, load balancers, NATs, WAN accelerators, TCP optimizers, DPIs, etc
- SFC: lista ordenada de instancias de estas funciones de servicio
- Service Function Path (SFP)
- Es agnóstico a cómo se transporten los paquetes (por la *underlay*)

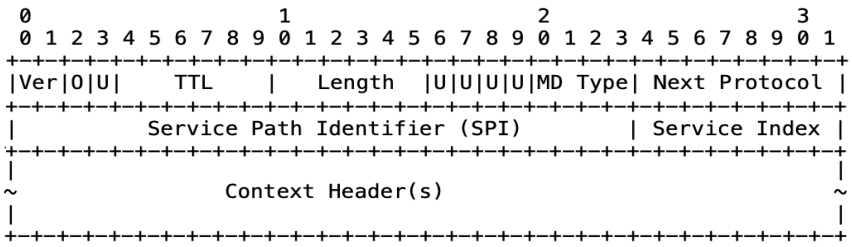
SFC + SDN + NFV

- NFV : La virtualización de los servicios
- SFC : La cadena de servicios por los que debe pasar el flujo
- SDN : El control de la red para llevar a cabo ese camino



NSH

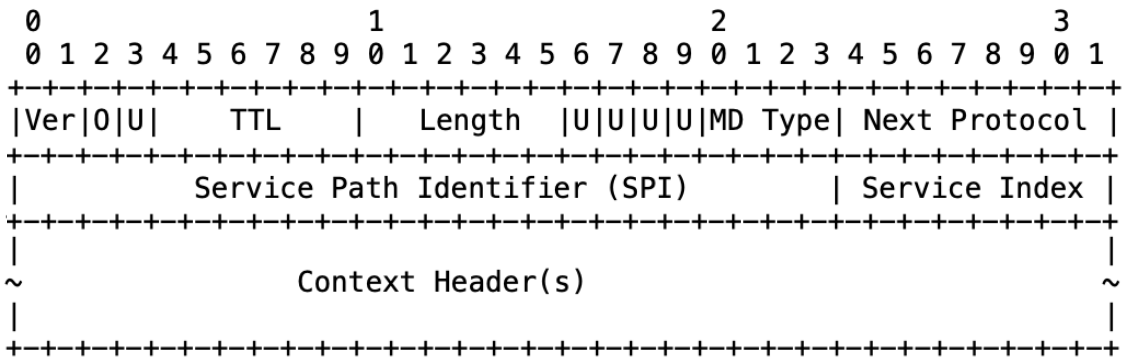
- RFC 8300, “Network Service Header” (Cisco, Intel, 2018)
- TTL (*loop* prevention) indica el máximo número de saltos de servicio (SFFs)
- Next Protocol: IPv4 (0x1), IPv6 (0x2), Ethernet (0x3), NSH (0x4), MPLS (0x5)
- *Service Path Identifier:* identifica al SFP y con él a las SF por las que pasar
- *Service Index:* Da localización dentro del SFP (inicial 255 y lo decrementa cada SF o SFC Proxy)
- Transporta metadatos que pueden necesitar las SFs



SFF = Service Function Forwarder

NSH

- RFC 8300, “Network Service Header” (Cisco, Intel, 2018)
- TTL (*loop* prevention) indica el máximo número de saltos de servicio (SFFs)
- Next Protocol: IPv4 (0x1), IPv6 (0x2), Ethernet (0x3), NSH (0x4), MPLS (0x5)
- *Service Path Identifier:* identifica al SFP y con él a las SF por las que pasar
- *Service Index:* Da localización dentro del SFP (inicial 255 y lo decrementa cada SF o SFC Proxy)
- Transporta metadatos que pueden necesitar las SFs



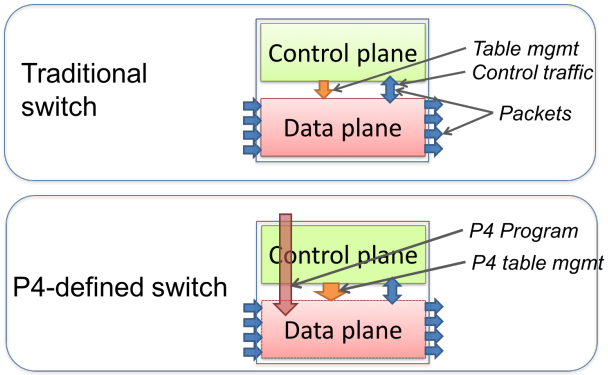
SFF = Service Function Forwarder

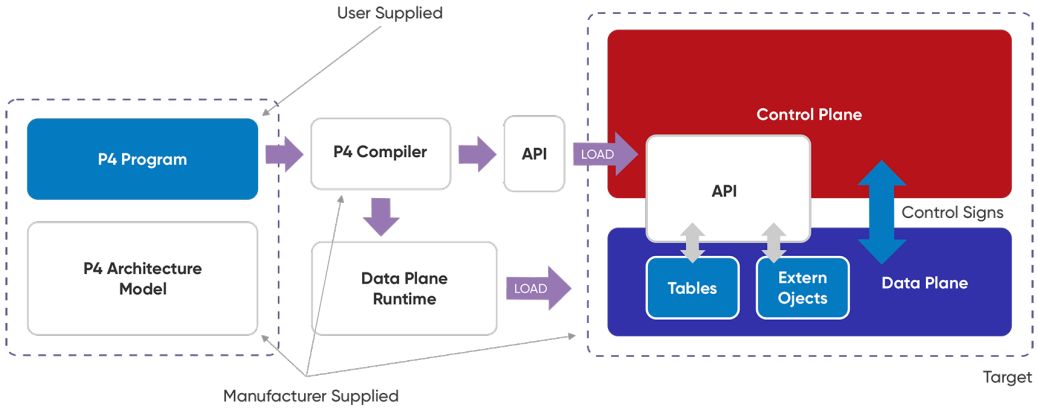
Evolución del hardware

- 40 años de ley de Moore (x2 transistores cada 24 meses)
- Hoy en día ya estamos empezando con la producción en 3 nm (Tick-tock)
- Servidores ciclos de 2 años, networking reutilización 8-10 años
- Los fabricantes de equipos de red están adoptando los ritmos de producción de electrónica
- Empujados por pocos grandes clientes
- Por ejemplo: donde teníamos SerDes a 10 Gb/s los tendremos posteriormente a 25 Gb/s, al mismo coste
- Esto permite interfaces 100GE donde antes teníamos 40GE, al mismo precio
- Hoy en día SerDes a 112 Gb/s
- A día de hoy SoC (Switch on Chip) a varios Tbps
- *Merchant silicon*
  - ASICs creados por una compañía pero switches ensamblados por otra
  - Broadcom, Marvell, Barefoot Networks (ahora Intel), Mellanox (ahora Nvidia)

Plano de datos

- En el propio diseño del hardware
- Nuevos ASICs son reprogramables
- Los objetos en el plano de datos no están fijos, se pueden modificar y con ellos lo que puede modificar el plano de control
- Programming Protocol-independent Packet Processors (P4)





https://p4.org/p4-spec/docs/P4-16-v1.2.1.html

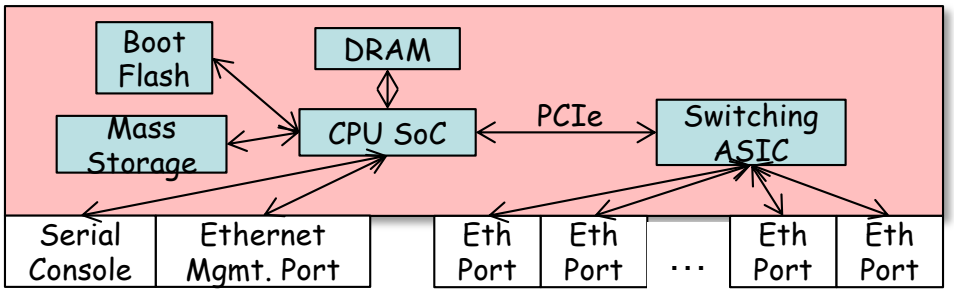
```
parser TopParser(packet_in b, out Parsed_packet p) {
    Checksum16() ck;  // instantiate checksum unit

    state start {
        b.extract(p.ethernet);
        transition select(p.ethernet.etherType) {
            0x0800: parse_ipv4;
            // no default rule: all other packets rejected
        }
    }

    state parse_ipv4 {
        b.extract(p.ip);
        verify(p.ip.version == 4w4, error.IPv4IncorrectVersion);
        verify(p.ip.ihl == 4w5, error.IPv4OptionsNotSupported);
        ck.clear();
        ck.update(p.ip);
        // Verify that packet checksum is zero
        verify(ck.get() == 16w0, error.IPv4ChecksumError);
        transition accept;
    }
}
...
```

¿Evolución?

- La infraestructura se simplifica, principalmente el hard, controlable por soft
- *White boxes* no solo servidores sino también switches
- También se venden ya switches *"Bare metal"* = solo el hardware
- Menores costes
- El mismo equipo un día es un switch, otro un firewall, otro un balanceador… dentro de las limitaciones del ASIC
- Para estos equipos sistemas operativos y gran cantidad de software, generalmente basados en linux, muchos de código abierto
- Es decir, igual que en el entorno de servidor, puedes cambiar el hardware, instalar el sistema operativo que quieras y desarrollar tus aplicaciones
- Para diferenciarse, los proveedores desarrollan software propietario para ofrecer sus servicios
- Porque hoy en día ya es el software por lo que principalmente están cobrando los fabricantes "no-open"
- Muchos modelos ToR de fabricantes conocidos son switches bare-metal que han comprado, cambiado el software y el frontal



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

Interconexión de DCs: Introducción

Availability entre DCs

- Buscamos protección ante desastres:
  - Tsunamis, huracanes, inundaciones, terremotos, incendios
  - Fallos de larga duración de la red eléctrica (*black-outs*)
  - Violaciones de seguridad
- No es solo una cuestión de disponibilidad física sino que la lógica para coordinarlos debe funcionar correctamente también
- Pueden trabajar por parejas en modo **activo-standby**
  - Uno de ellos cursa toda la carga de trabajo; el segundo monitoriza el estado del activo
  - Operaciones que modifiquen datos almacenados se **sincronizan** con el almacenamiento en el de respaldo
- Pueden trabajar en modo **activo-activo**
  - Necesitamos técnicas de reparto de carga entre los DCs
  - Así como (de nuevo) técnicas para **sincronizar** los datos entre ellos

Ubicación

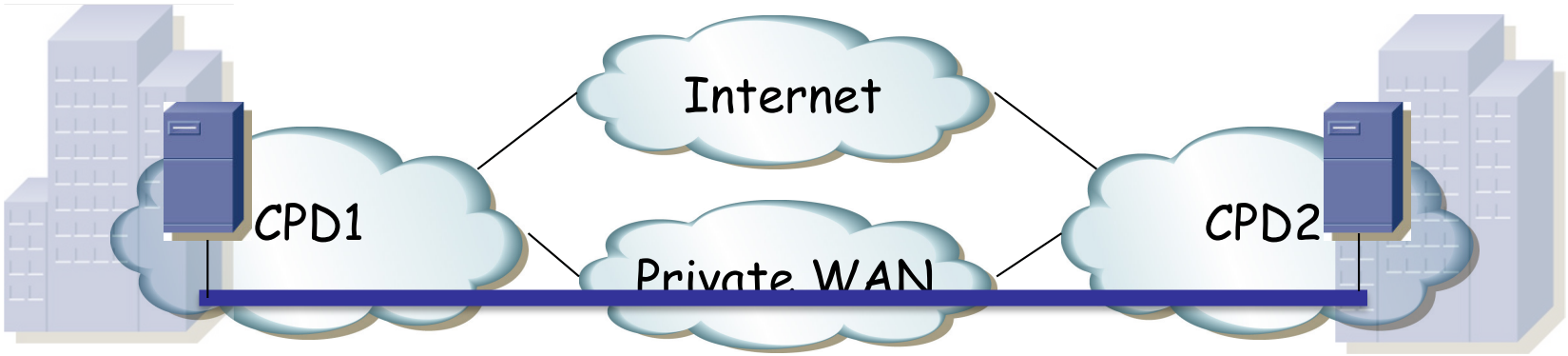
- **Alejados** para que un problema "geográfico" no afecte a ambos
- Sin embargo, podemos toparnos con **limitaciones de retardo máximo** para las aplicaciones distribuidas si están alejados
- Por ejemplo, la ***replicación síncrona*** se basa en devolver confirmación de haber almacenado el dato cuando se ha escrito en dos cabinas
- También protocolos como FC deben ajustarse para altos retardos (mayor RTT requiere mayor número de créditos para sacar provecho al BW)

Balanceo mediante DNS

- *Content routing, request routing, Global Server Load Balancing (GSLB)*
- Usuario emplea un Proxy DNS (acepta *query* recursiva)
- Respuesta en función de geolocalización (al más cercano)
- Respuesta incluye al otro CPD como alternativa
- Distribuir en activo-activo entre los CPDs debe emplear algún método de *stickiness*

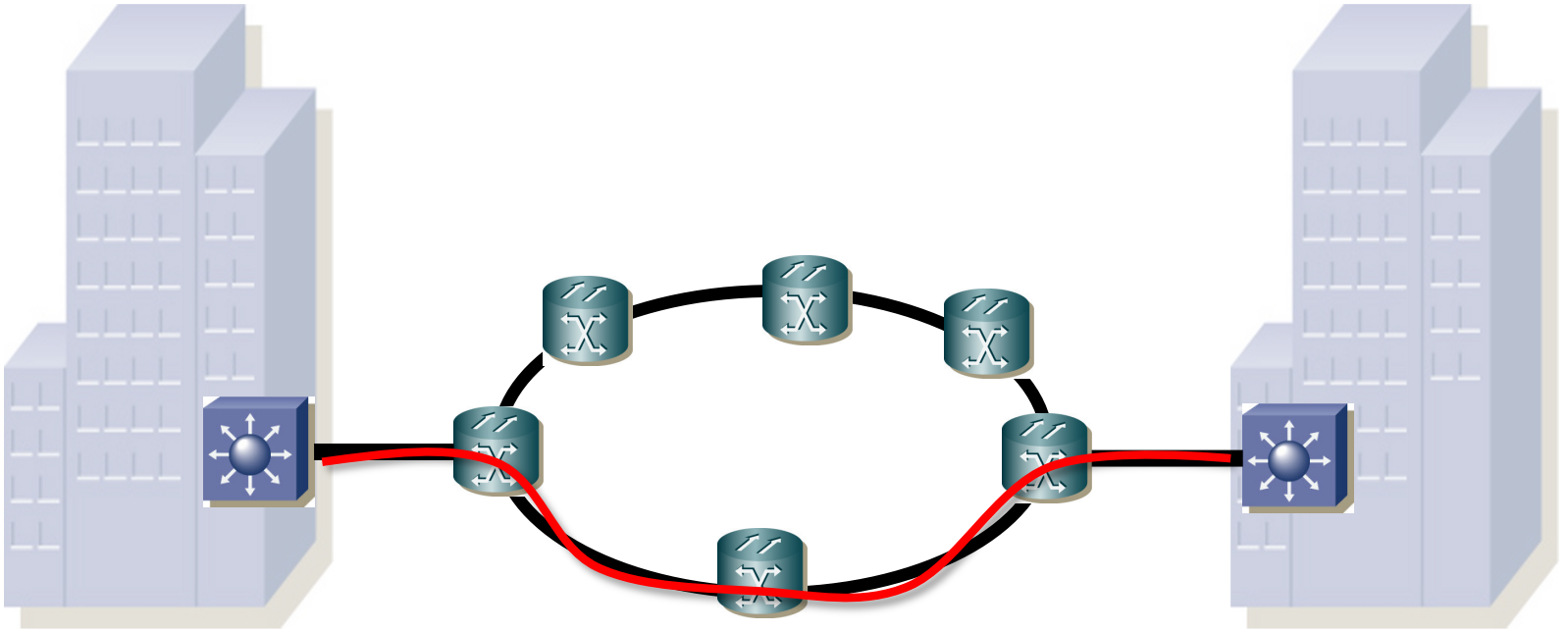
¿ Interconexión L2 o L3 ?

- Habitualmente la interconexión se recomienda en capa 3
- Eso limita los problemas de capa 2 a cada DC
- Sin embargo muchas aplicaciones con funcionalidades de clustering requieren adyacencia en capa 2
  - Heartbeats o información de estado que envían multicast/broadcast
  - Nodos que comparten dirección IP y dirección MAC
- La movilidad de servidores (físicos o virtuales) requiere mantener la pertenencia a la misma VLAN
- O el crecimiento nos puede llevar a otro edificio
- Es decir, podemos necesitar extender las VLANs entre DCs
- Todo esto aplica tanto a interconexión de CPDs como de sedes remotas



Interconexión por fibra

- Se puede emplear *fibra oscura*
- Puede transportar múltiples wavelengths (CWDM, DWDM)
- Solución habitual para el almacenamiento (“SAN extension”)
- O se podría transportar una o varias wavelengths por una red de conmutación óptica
- Esta red puede dar protección
- La distancia sigue limitada pues da continuidad óptica (no hay OEO)
- Y es probable que queramos redundancia en el acceso a ella



Múltiples *Sites*

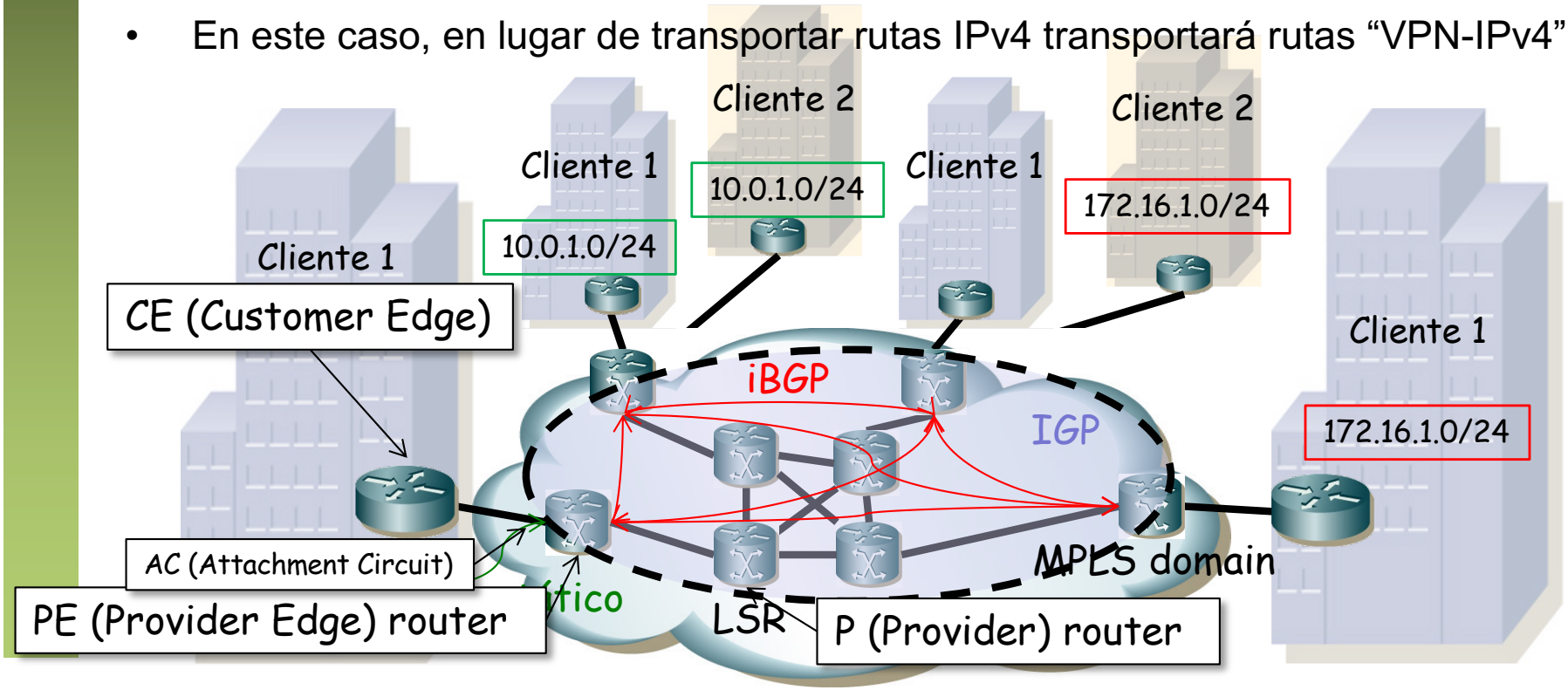
- En vez de circuitos ópticos podrían ser SDH, ATM, MPLS, etc
- Independientemente de cómo se resuelva el transporte WAN
- Ignorando el transporte WAN tenemos circuitos p2p entre las sedes
- Pueden formar diversas topologías (*hup&spoke, mesh, etc)*
- Cualquiera de los transportes (f.o., SDH, ATM, MPLS) permite transportar Ethernet o IP
- Si crea bucles en Ethernet habrá que resolverlos (STP, EPB, TRILL)
- Serían L2 VPNs o L1 VPNs
- Hoy en día es habitual la solución de transporte WAN MPLS
- En lugar de wavelengths o PVCs ATM tenemos LSPs
- Recordemos que podemos encapsular Ethernet sobre MPLS (EoMPLS)
  - RFC 4448 “Encapsulation Methods for Transport of Ethernet over MPLS Networks”
- De hecho se suele decir que tenemos “AToM” o “Any Transport over MPLS”
- Los equipos de usuario van a poder ser capa 2 o capa 3

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

Layer 3 MPLS VPNs

Layer 3 VPNs

- RFC 4364 “BGP/MPLS IP Virtual Private Networks (VPNs)” (Cisco Systems y Juniper Networks, 2006)
  - *“This document describes a method by which a Service Provider may use an IP backbone to provide IP Virtual Private Networks (VPNs) for its customers.”*
- VPN para el transporte de paquetes IP entre sedes (*sites*)
- El backbone del proveedor de servicio es una red IP MPLS
- RFC 4760 “Multiprotocol Extensions for BGP-4” (Cisco, Sanoa, Juniper, 2007)
- Extensiones a BGP-4 para poder transportar información de otros protocolos de nivel de red: IPv6, IPX, L3VPN, etc
- En este caso, en lugar de transportar rutas IPv4 transportará rutas “VPN-IPv4”



L3VPN: Routing

- Los CEs anuncian sus rutas a los PEs (con un IGP o rutas estáticas)
- Los PEs emplean MP-BGP para intercambiarse esas rutas (iBGP)
- El PE la distribuye al CE del mismo cliente (de la misma VPN)
- Los P y PE corren un IGP para tener alcanzabilidad interna
- Los CE son routers convencionales, no necesitan ninguna configuración de VPN ni emplean MPLS
- Los CEs no intercambian información de routing entre ellos, no son adyacentes
- La VPN no actúa como un overlay sino una red IP con otro gestor
- Dos VPNs pueden emplear espacios de direcciones IP que se solapen
- Los anuncios VPN-IPv4 (MP-BGP) incluyen un identificador (*Route Distinguisher = RD, 64 bits)* que las diferencia
- Anuncio de la AF (Address Family) VPN-IPv4 es un prefijo de 12 bytes:

  `<Route Distinguisher>:<IP Prefix>`
- Permite también anunciar más de una vez el mismo prefijo IPv4 para el mismo cliente empleando distinto RD (múltiples rutas para el mismo destino)
- RD no tiene significado pero un formato habitual es <ASN>:<id>
- Así cada service provider tiene su espacio de valores RD
- Las sesiones iBGP son entre los PE, así que ...
- Los P no ven las rutas de las VPNs (evita problemas de escalabilidad)
- ¿Cómo se enruta si hay direcciones duplicadas y los routers centrales las ven?

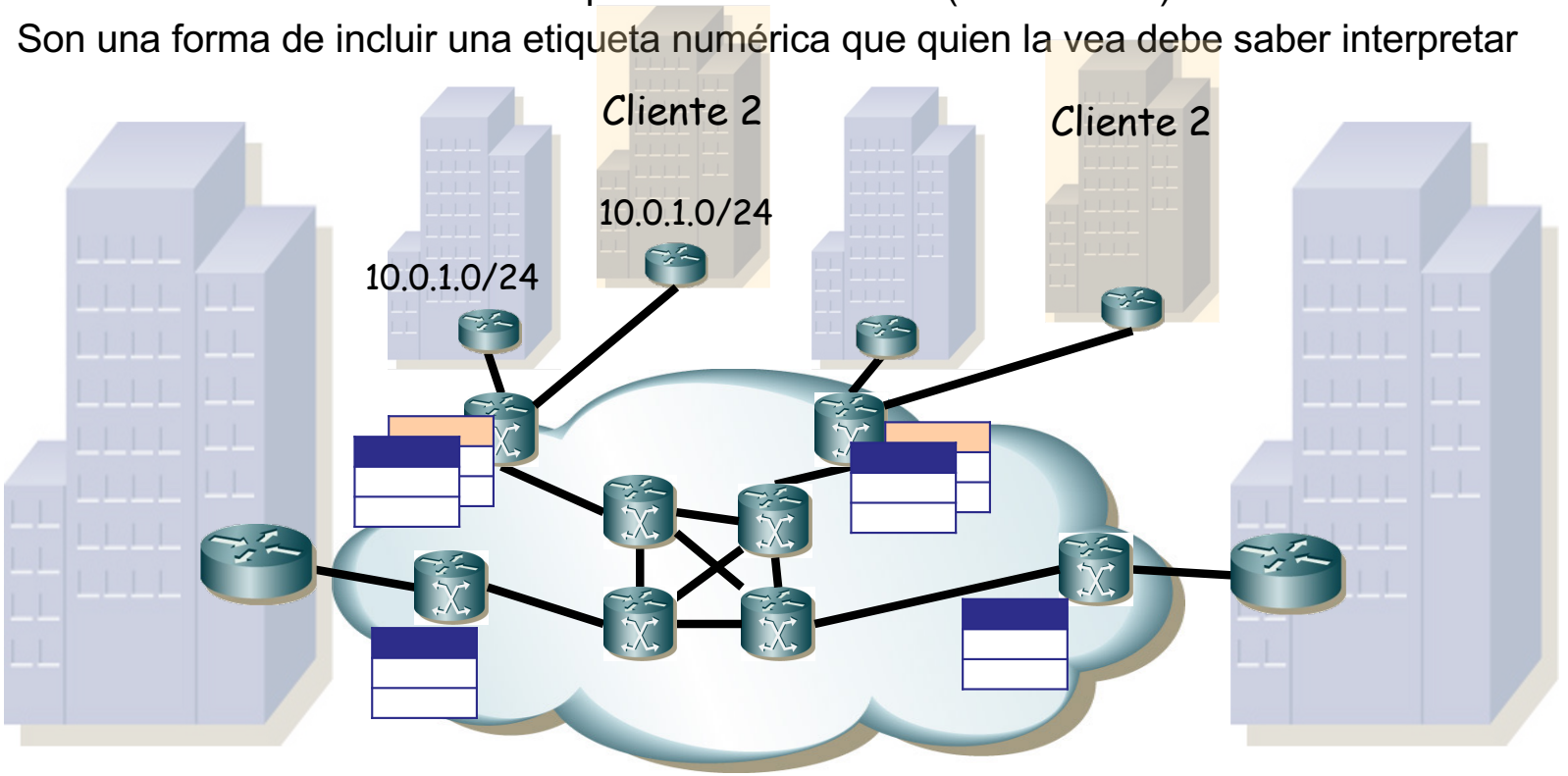
L3VPN: Forwarding

Forwarding

- Cada PE mantiene una tabla de rutas para cada cliente y además una tabla por defecto
- VPN o *VPN/Virtual Routing and Forwarding tables (VRFs)*
- Al recibir un paquete IP de un cliente consulta la VRF correspondiente

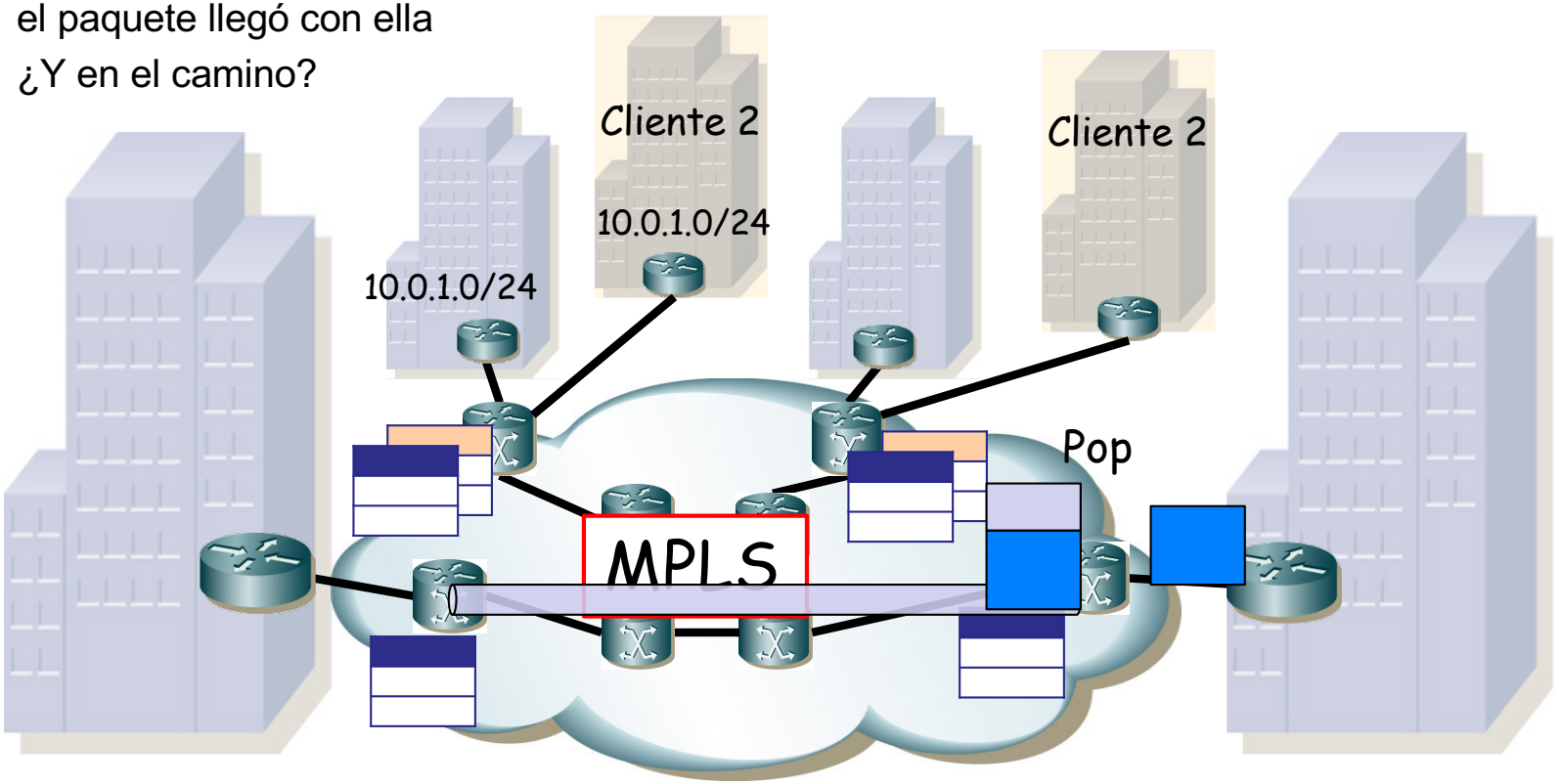
Routing

- Los anuncios MP-BGP de la AF VPN-IPv4 incluyen uno o más *"Route Targets"* (RT)
- Una VRF importa las rutas que traen los RT que desee
- El RT se transporta como una *extended community* (RFC 4360)
- Las comunidades son atributos opcionales transitivos (32 ó 64 bits)
- Son una forma de incluir una etiqueta numérica que quien la vea debe saber interpretar



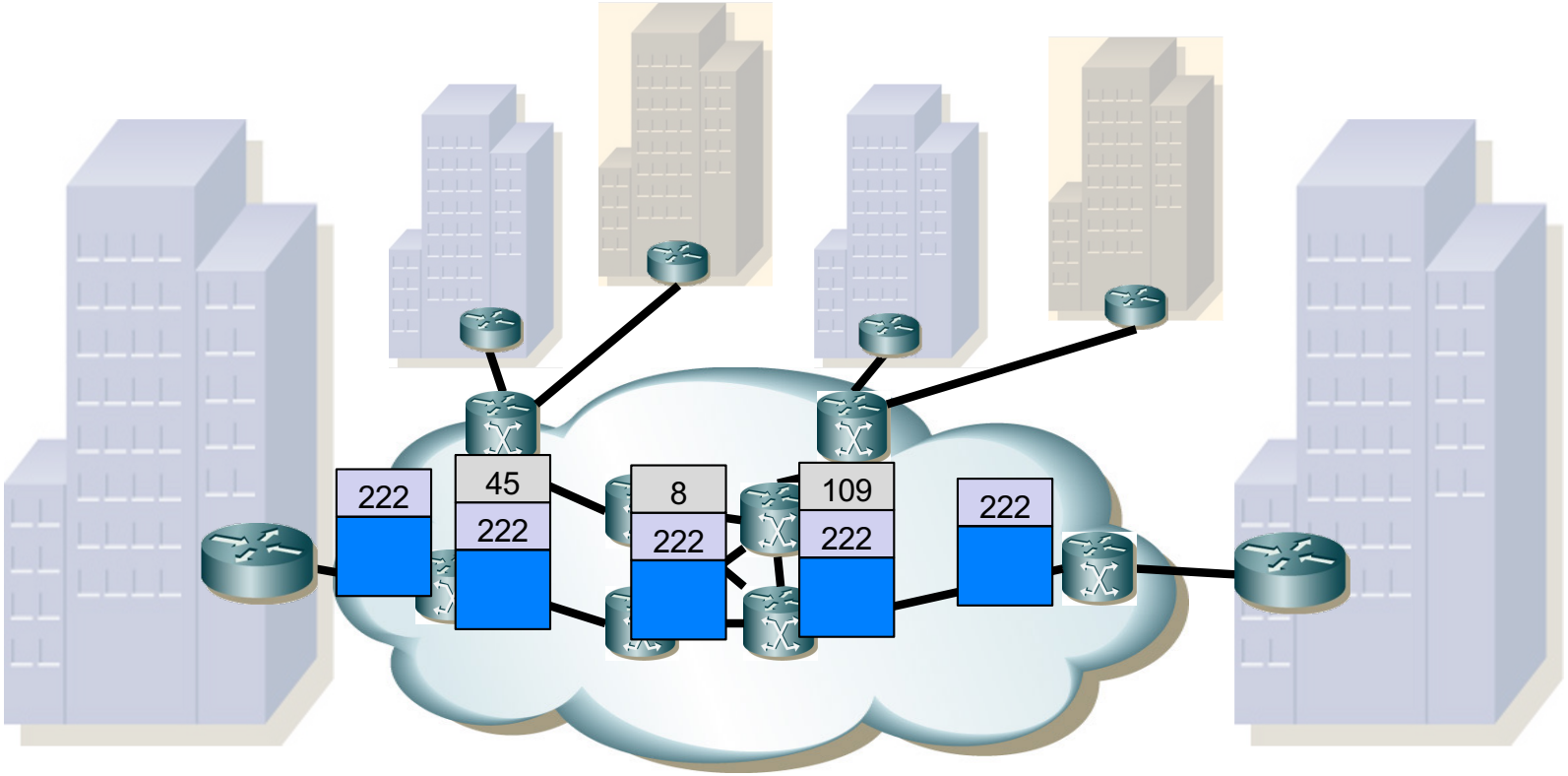
L3VPN: Routing + Forwarding

- RD:prefijo y RT son plano de control
- El anuncio MP-BGP incluye una etiqueta MPLS
- Esta etiqueta es para el plano de datos; asociada a la ruta
- Paquetes de la VRF que siguen esa ruta se introducen en LSP con esa etiqueta (*VPN route label)*
- Al salir el paquete en el PE de egreso se sabe a qué VRF pertenece en base al LSP
- No puede usar la dirección IP destino ya que pueden estar duplicadas
- El paquete MPLS viene con la etiqueta que puso el de ingreso
- Es decir, PE de egreso hizo el anuncio de la ruta con la etiqueta, PE de ingreso la añadió y el paquete llegó con ella
- ¿Y en el camino?



L3VPN: MPLS

- Un LSP entre cada pareja de PEs
- Mala escalabilidad para los P routers (muchos LSPs los atraviesan)
- Se crean LSPs entre los PEs (“Tunnel LSPs”)
- Los P routers reenvían en función de esa etiqueta externa
- Un full-mesh entre los PEs que compartan VRF

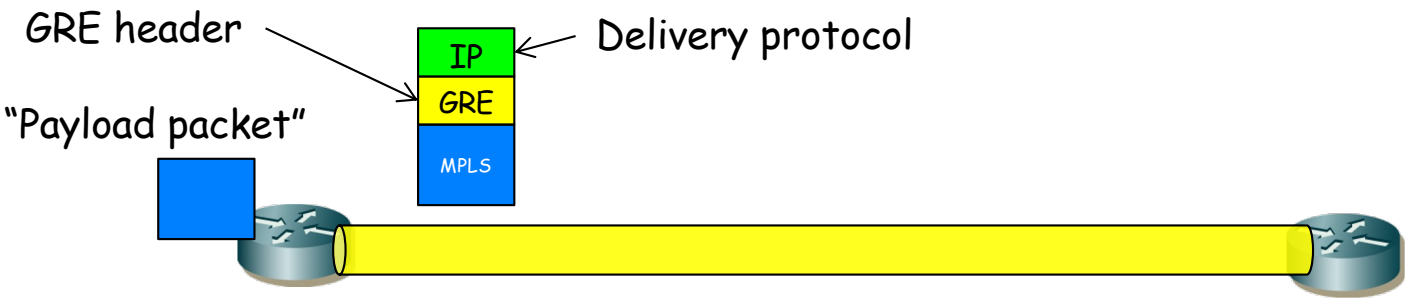


Recapitulando

- Un LSP entre cada pareja de PEs que hacen de túnel para los LSPs de cada cliente
- Cada PE tiene una VRF para cada cliente
- Dado el interfaz (el AC) de entrada del paquete distingue al cliente y su VRF
- La ruta en la VRF indica la etiqueta MPLS a añadir y el puerto de salida
- Aprendió esa ruta y su etiqueta asociada mediante MP-BGP
- Asoció esa ruta a esa VRF en base al RT
- Distingue en MP-BGP esa ruta de otra que tenga la misma dirección IP destino con el RD
- Esa etiqueta corresponde al LSP del usuario
- El puerto de salida es el Tunel LSP entre los dos PEs
- Se le añade otra etiqueta que identifica al Tunel LSP entre esos dos Pes
- Esa etiqueta corresponde al LSP del usuario
- El puerto de salida es el LSP entre los dos PEs
- Se le añade otra etiqueta que identifica al LSP entre esos dos PEs
- Los P routers reenvían en función de esa etiqueta externa
- Los P routers no ”ven” la etiqueta interna y por lo tanto no ven los LSPs de usuarios, solo la malla entre los PEs
- En el PE de egreso la etiqueta del LSP permite identificar la VRF y con ella la tabla de rutas

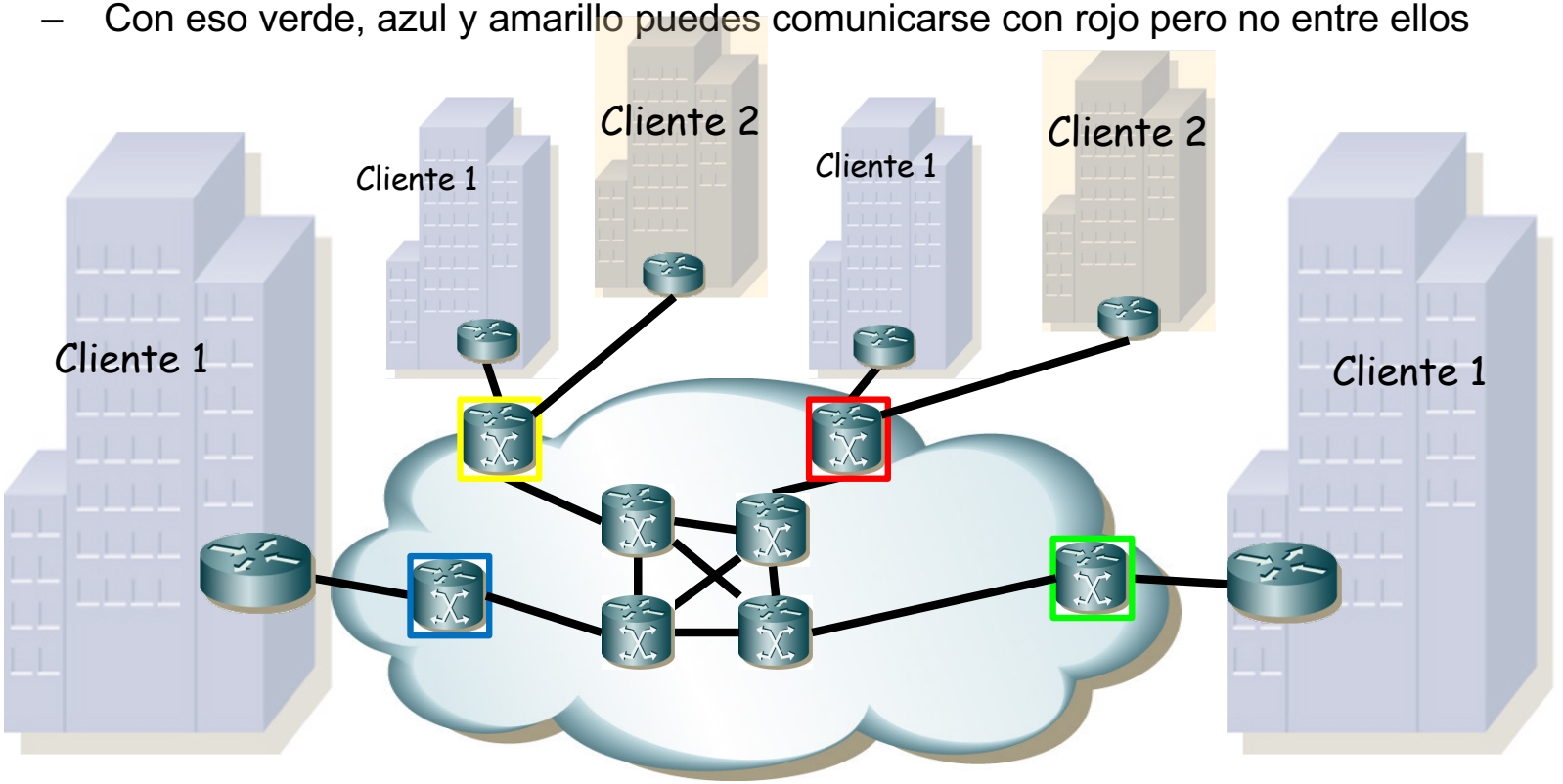
MPLS in GRE in IP

- Los túneles entre los PE pueden ser túneles GRE o simple IPoIP (RFC 4797)
- Entonces la red de transporte ya no necesita ser MPLS, es simple IP
- RFC 4023 “Encapsulating MPLS in IP or Generic Ruting Encapsulation (GRE)” (Motorola, Juniper, Cisco, 2005)
- El “*delivery protocol*” podría ser IP (protocol = 47 = GRE)
- El “*payload packet*” podría ser MPLS (Ethertype 0x8847 para unicast y ese mismo ó 0x8848 para multicast, RFC 5332)
- EoMPLSoGRE = Ethernet over MPLS over GRE
- Al transportarse sobre IP puede emplear IPSec
- RFC 4023 contempla también que MPLS se transporte directamente sobre IP, lo cual es más eficiente (sin GRE, protocolo 137 sobre IP)
- Puede haber motivos para tener GRE (exista el túnel con anterioridad, la implementación del equipo lo requiera en su fastpath, etc)



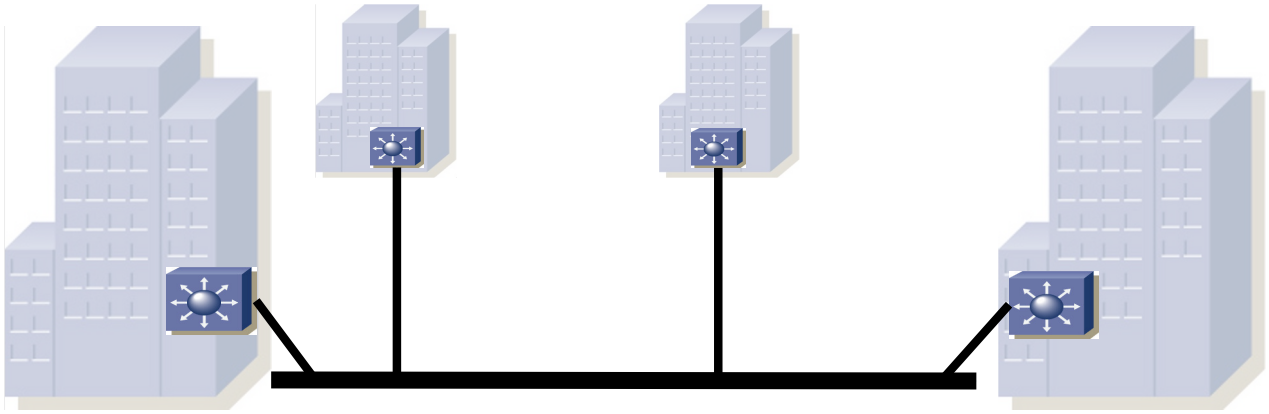
RD y RT

- RD permite anuncios del mismo prefijo que se puedan diferenciar
- RT permite importar en una VRF los anuncios con ciertos RTs
- En el anuncio se pone el RT que emplearemos para importar en otras VRFs
- Unos PEs pueden importar los anuncios de un solo RT y otros los de varios
- Ejemplo:
  - Para el cliente 1 cada PE podría hacer sus anuncios con un RT diferente
  - Podríamos decirle a los PEs amarillo, azul y verde que importen del RT rojo
  - Y decirle al PE rojo que importe de los RTs amarillo, azul y verde
  - Con eso verde, azul y amarillo puedes comunicarse con rojo pero no entre ellos



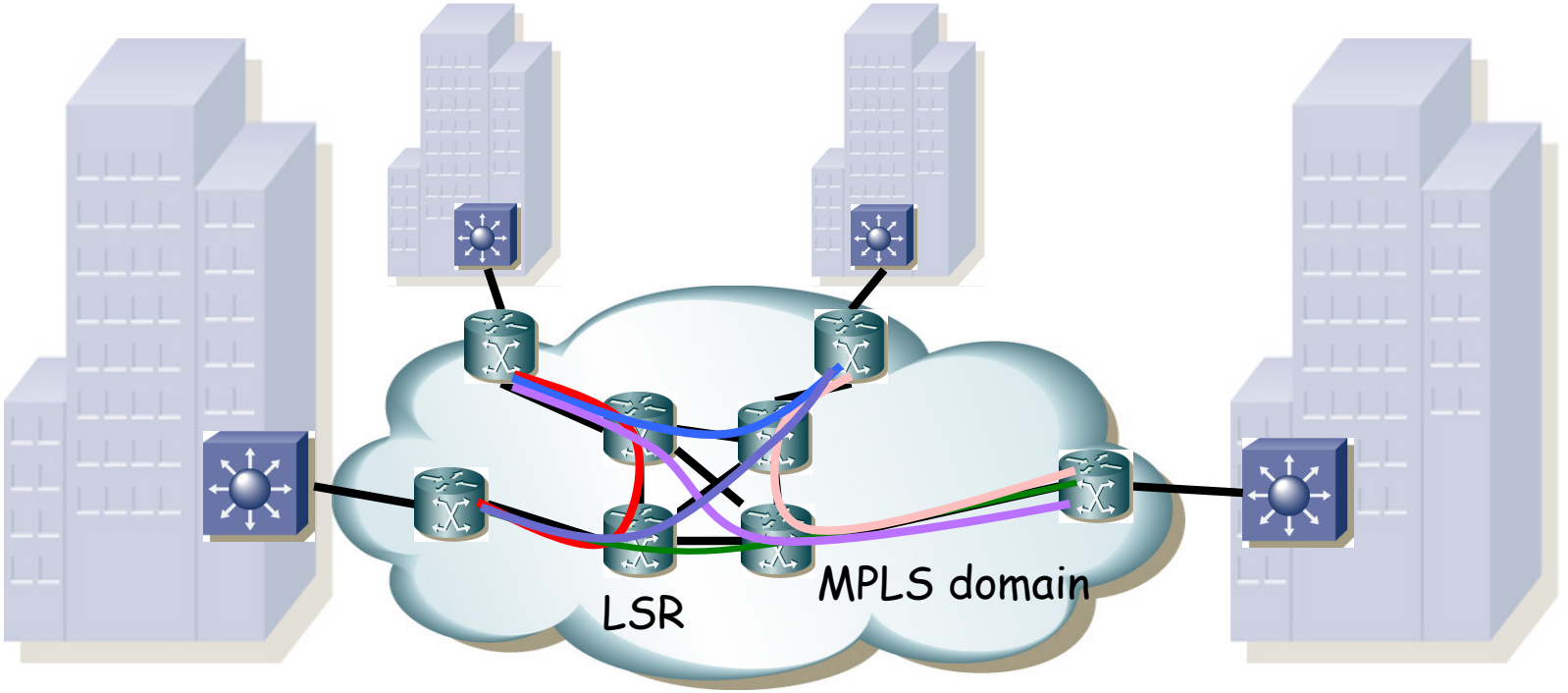
MPLS y VPLS

- *"Virtual Private LAN Service",* una VPN layer 2 (RFC 4664, Acreo y Cisco, 2006)
- Interconecta múltiples *sites* en un solo dominio puenteado
- Los extremos se comportan como si estuvieran en una LAN (*E-LAN Service)*
- Transporta Ethernet; sobre ella puede transportar IP o cualquier otro protocolo
- Los equipos de usuario (Customer Edge) pueden ser switches o routers
- El dominio MPLS puede transportar las tramas MPLS directamente o sobre IP u otra tecnología (GRE, L2TP, IPsec)
- La red puede dar servicio VPLS a más de un cliente
- El PE hace aprendizaje de direcciones MAC y replicación de tramas de forma independiente para cada cliente
- No interfiere el servicio de un usuario al otro (pueden por ejemplo emplear el mismo direccionamiento IP)
- Los equipos frontera establecen entre ellos los LSPs necesarios para el servicio multiacceso



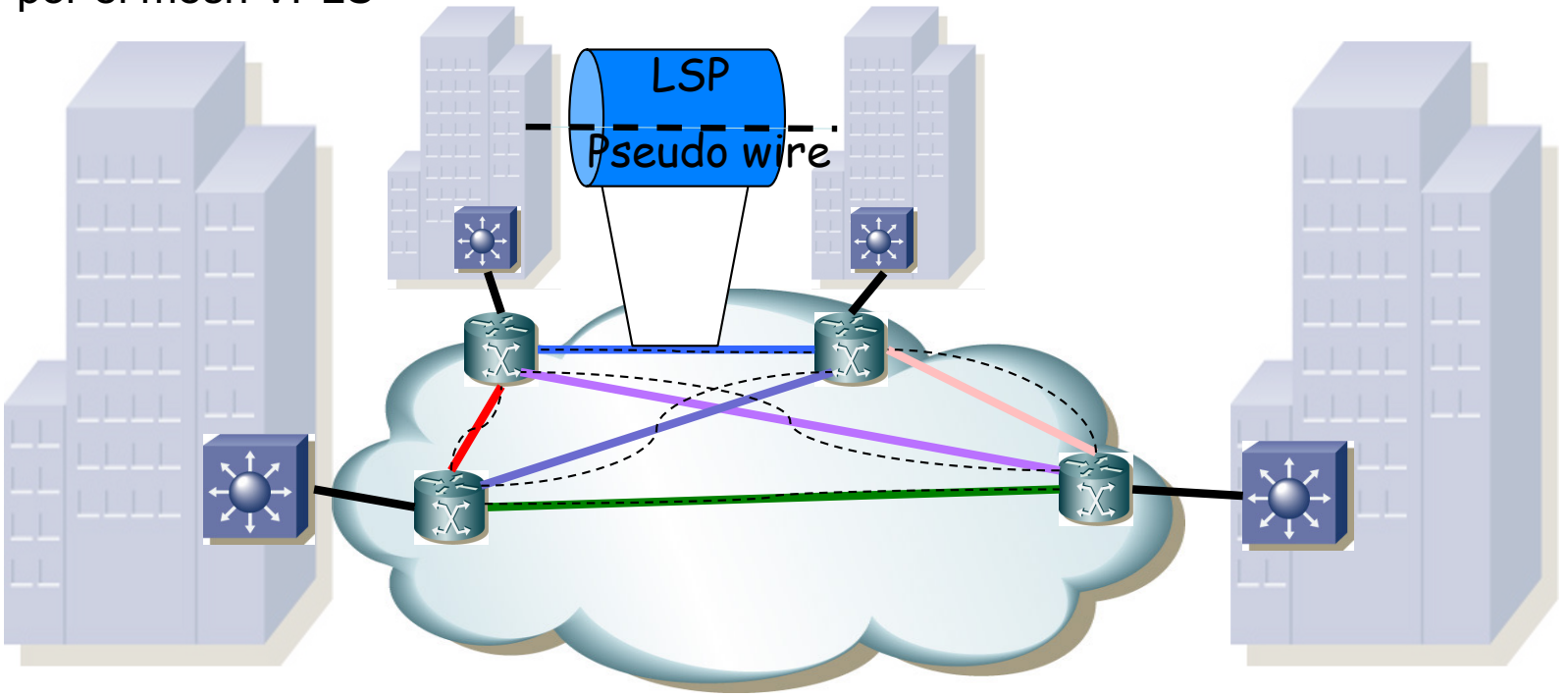
MPLS y VPLS

- Los PEs establecen un *full-mesh*
- Para ello disponen de RSVP-TE (o LDP)
- Esos LSPs son globales al servicio VPLS, no particulares para cada cliente
- Por cada instancia VPLS (cada cliente) se establece un full mesh de *pseudo-wires* (PWs) entre los PEs (en el fondo la segunda etiqueta MPLS)
- RFC 3985 “Pseudo Wire Emulation Edge-to-Edge (PWE3) Architecture” (Cisco Systems, Overture Networks, 2005)
- Un PW emula un circuito, por ejemplo para transportar un E1 o un PVC ATM
- También puede transportar Ethernet, AAL5, SDH, etc



VPLS y PWE

- El full-mesh de PWs hace que los PE puedan enviarse directamente los unos a los otros
- No necesitan hacer reenvío y no hace falta resolver posibles bucles
- Simplemente se implementa una solución que se llama de "*split horizon*":
  - Un PE no debe reenviar tráfico de un PW a otro en el mismo mesh VPLS
- El aprendizaje de direcciones MAC se hace en los PEs en el plano de datos (con la llegada de tramas Ethernet)
- Sí puede haber ciclos, pero creados por el usuario para obtener redundancia
- En ese caso podrá emplear STP y las BPDUs se transportarían normalmente por el mesh VPLS

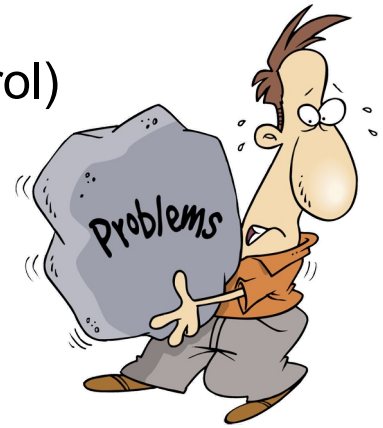


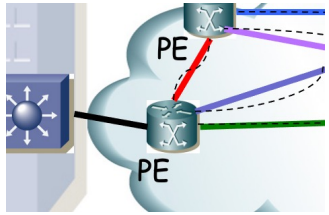
VPLS Control Plane

- Alternativas para el establecimiento de los pseudo-wires:
  - RFC 4761 “Virtual Private LAN Service (VPLS) Using BGP or Auto-Discovery and Signaling”
  - RFC 4762 “Virtual Private LAN Service (VPLS) Using Label Distribution Protocol (LDP) Signaling”
- Repito: el aprendizaje de direcciones MAC se hace en el plano de datos, es decir, con la dirección MAC origen de la trama recibida (*flood and learn*)

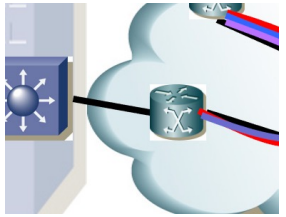
Problemas en VPLS

- Se deben establecer Nx(N-1)/2 pseudo-wires
- Problema de escalabilidad (cantidad de tráfico de control)
- Replicación de paquetes que sufren inundación:
  - Se lleva a cabo en el PE de entrada
  - Se dirigen punto-a-punto a cada otro PE del servicio
  - Mayor trabajo en el PE
  - Más uso de capacidad
  - Mayor retardo (si hay que enviar N veces la trama por N PWs que se implementan sobre el mismo LSP irán en serie)





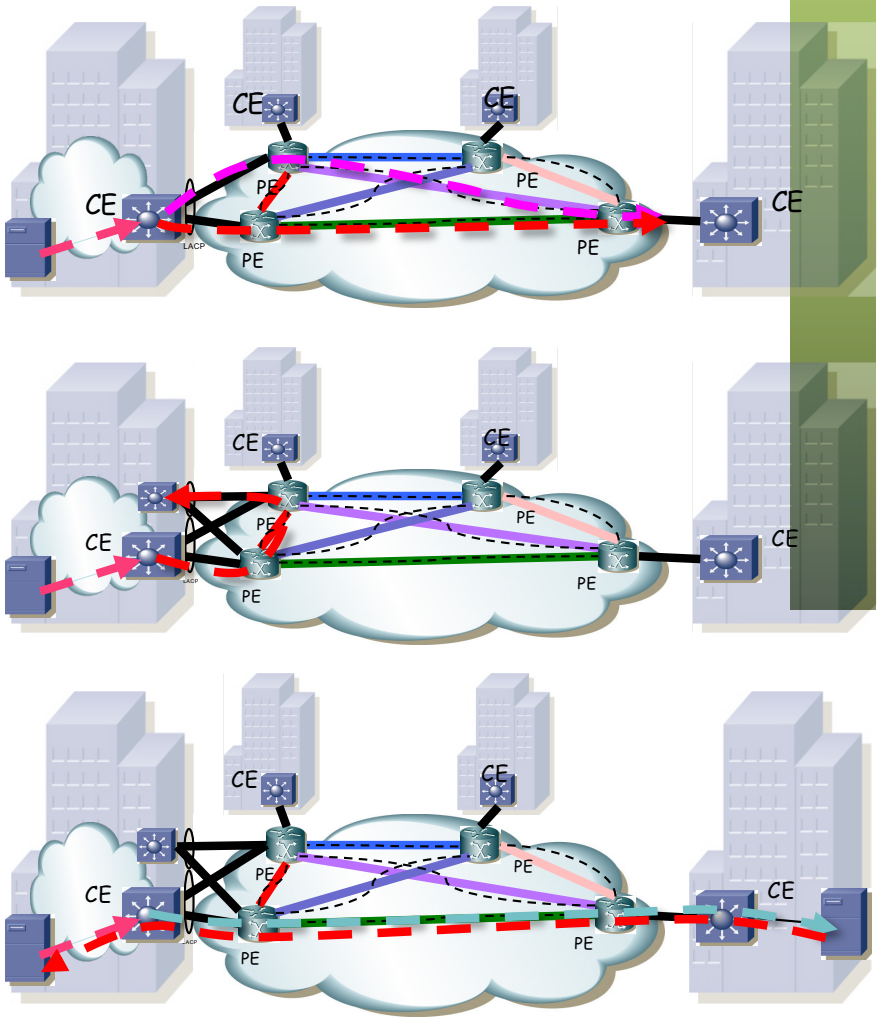
3 PWEs pero un solo interfaz físico



- Si se añade un acceso del cliente, a un PE diferente, se deben crear los PWs, lo cual implica reconfigurar los demás PEs
- Para despliegues pequeños
- Mejoras:
  - H-VPLS (Hierarchical VPLS)
  - Hierarchical BGP VPLS

Limitaciones de VPLS

- Solo soporta *Single-Active Redundancy Mode* (no *all-active*)
- Ejemplo:
  - CE un LAG hacia dos PEs y reparte tráfico entre ellos
  - Al llegar a otro PE llega la misma dirección MAC origen por dos PWs, saltando la MAC aprendida de uno a otro
- Ejemplo:
  - Dos CEs con LAGs (redundancia en el CE)
  - BUM es enviado por PE a todos los demás PEs y puede volver por el otro CE
- Ejemplo:
  - No hay load balancing en el sentido de respuesta si el hash lleva a emplear un solo PE
  - El otro extremo solo aprende la dirección MAC por un PW
  - En cualquier caso no la puede aprender por dos PWs
- Active-Active solo mediante soluciones propietarias (vPC, VSS, etc)



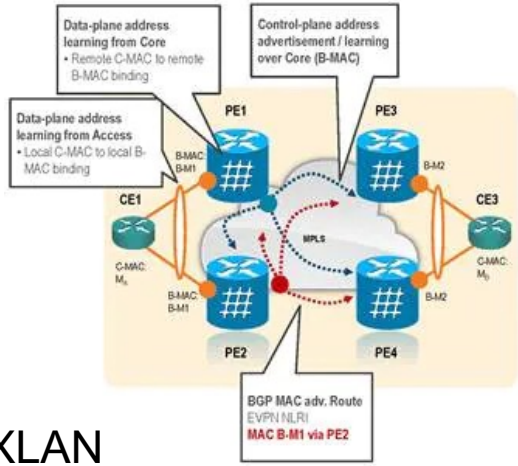
MPLS-EVPN

- RFC 7209 “Requirements for Ethernet VPN (EVPN)”, Cisco, Arktan, AT&T, Verizon, Alcatel-Lucent, Bloomberg (2014)
- RFC 7432 “BGP MPLS-Based Ethernet VPN”, Cisco, Arktan, Verizon, Bloomberg, AT&T, Juniper, Alcatel-Lucent (2015)
- Emplea un plano de control **BGP,** como una L3VPN
- Aprendizaje de direcciones MAC en el plano de control en lugar de en el plano de datos
- Ofrece una VPN capa 2, como VPLS
- Permite redundancia en el acceso (*all-active) y b*alanceo de carga por flujo
- Los PEs pueden estar conectados mediante LSPs o túneles (IP/GRE)
- Abandona el aprendizaje en el plano de datos (no *flood and learn*)
- BGP distribuye Ethernet MACs (opcional el par MAC-IP)
- PEs anuncian direcciones MAC aprendidas del CE, junto con una etiqueta MPLS, al resto de PEs mediante MP-BGP
- Queda abierto cómo aprende el PE del CE (puede ser plano de datos)
- Igual que las L3VPNs emplea *route distinguishers* y *route targets*
- Su uso principal es en DCI

Otros tipos de EVPN

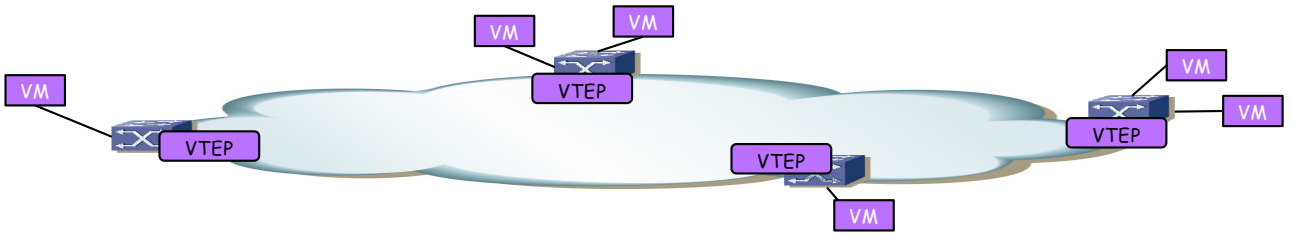
PBB-EVPN (RFC 7623) combina EVPN con plano de datos PBB (802.1ah)

- El core es MPLS
- La frontera son PEs de PBB que aprenden en el plano de datos las C-MACs asociadas a las B-MACs
- BGP anuncia las B-MACs
- Mejor escalabilidad en la RIB al no aprender las C-MACs



VXLAN-EVPN

- Plano de datos VXLAN (underlay IP)
- VTEPs corren proceso MP-BGP
- Anuncian las direcciones MAC de los hosts de la VXLAN
- Puede hacer *ARP suppression*
  - VTEP hace *ARP snooping* e inspecciona ARP request/response
  - VTEP aprende relación MAC/IP de cliente
  - VTEP anuncia esa relación mediante MP-BGP al resto de VTEPs
  - A partir de aquí, cualquier VTEP al hacer *ARP snooping* y ver una pregunta respecto a esa IP contesta él con la MAC aprendida



https://www.cisco.com/c/en/us/products/collateral/routers/asr-9000-series-aggregation-services-routers/whitepaper_c11-731864.html

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

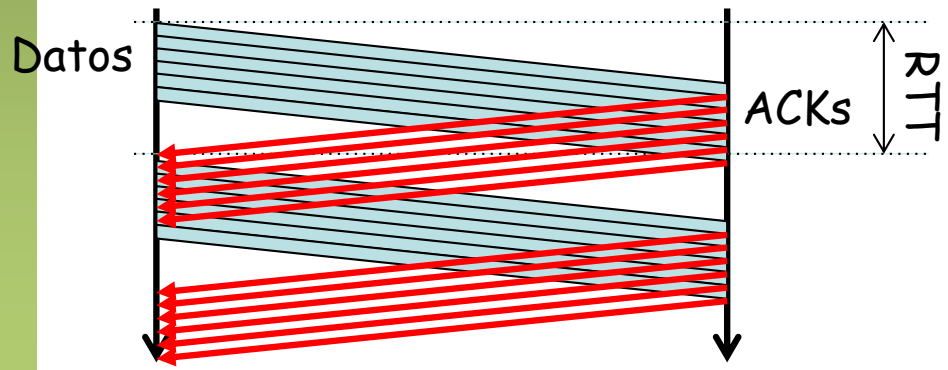
WAN optimization

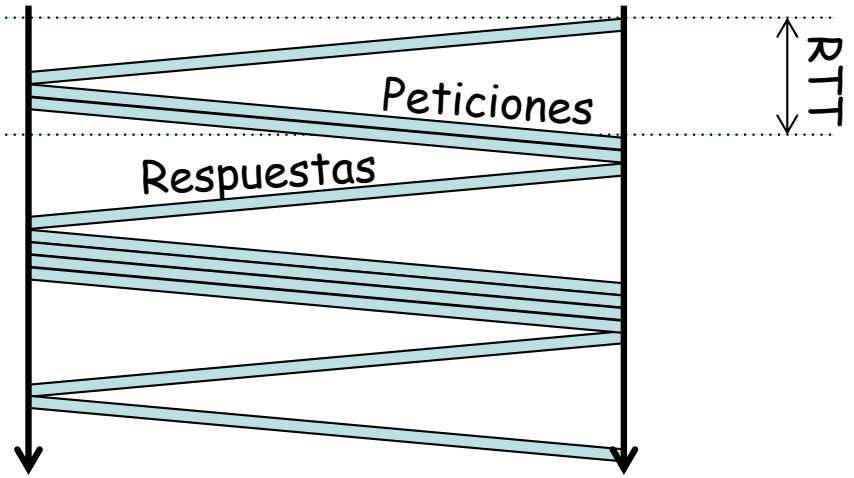
Rendimiento en la WAN

- Muchos protocolos y aplicaciones se han desarrollado para el entorno LAN
- Al emplearlos en el entorno WAN se encuentran con limitaciones debidas a bandwidth, latencia, pérdidas, servidor cargado, etc
- En los 90s accesos WAN de bajas velocidades (<512Kbps) y caros
- Una oficina remota puede no justificar el coste o simplemente no haber disponibilidad en esa región
- Los servidores tenían que estar cerca de los usuarios
- Eso quiere decir gran cantidad de servidores, más caro
- Más complicado gestionarlos, securizarlos, hacer backups, actualizaciones, etc
- Administración y mantenimiento más complejo y caro
- Hoy en día accesos de baja o alta velocidad según la localización o las necesidades
- DSL, cable modem, fibra, metro Ethernet, UMTS, satélite, etc
- Por el enlace WAN: acceso a ficheros, backups, e-mail, acceso a Internet, streaming, impresión, aplicaciones de gestión, aplicaciones empresariales, autentificación, sesión remota, etc.
- Intentamos centralizar los servicios
- Han aparecido equipos que buscan acelerar el comportamiento de las aplicaciones en base a modificar los flujos de aplicación
- Evidentemente son muy dependientes del tipo de aplicaciones y su uso

Latencia (retardo)

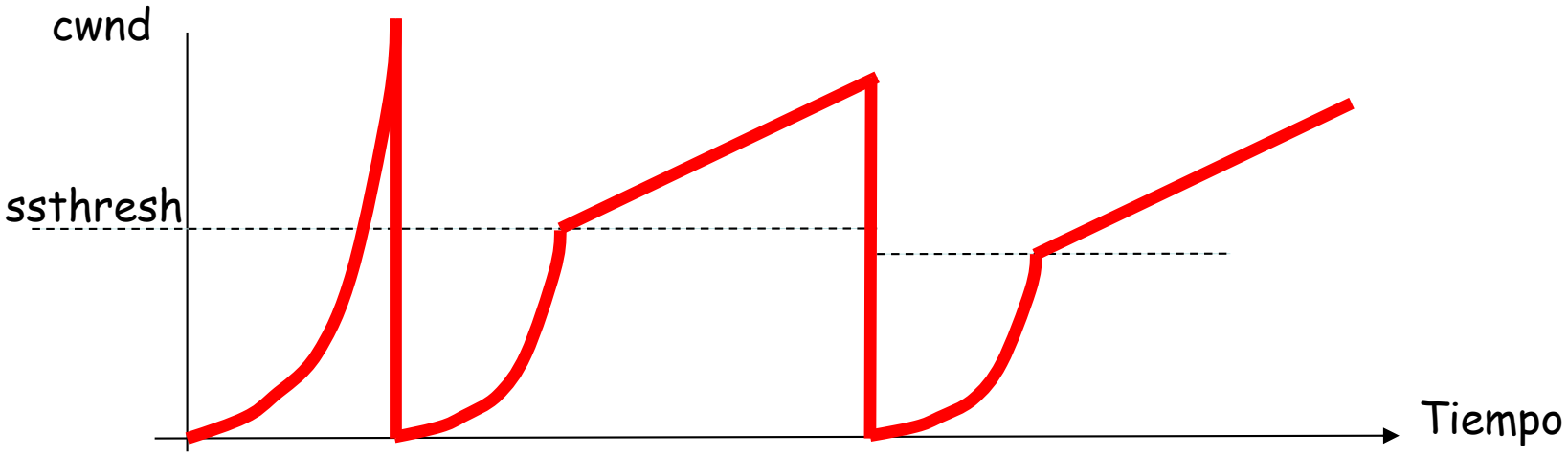
- El RTT tiene un impacto en la velocidad de transferencia y en la interactividad
- En protocolos de ventana deslizante estamos limitados por el tamaño de la ventana entre el RTT
- Esto no es solo TCP sino también protocolos de nivel de aplicación (por ejemplo SMB)
- Añadir capacidad a los enlaces no mejora el throughput (limitado por ventana)
- Aplicaciones muy *chatty* consumen muchos RTTs con pocos datos
- No están limitadas por la ventana del protocolo sino porque necesita la respuesta a una petición antes de poder hacer la siguiente





Pérdidas (TCP)

- Si la ventana de control de flujo es suficientemente grande
- Y la transferencia dura suficiente
- TCP aumenta cwnd hasta congestionar el camino
- Se producen pérdidas y baja agresivamente la tasa de envío
- Y repite (más lento)

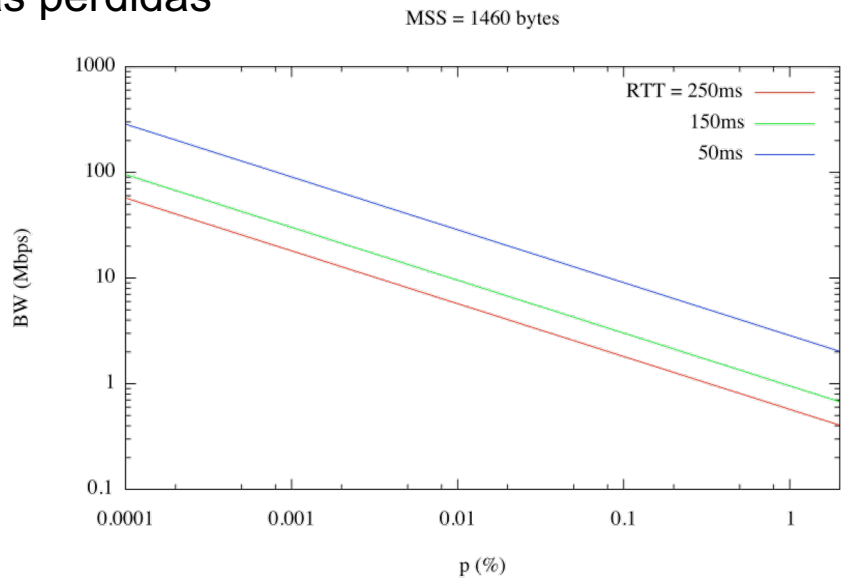


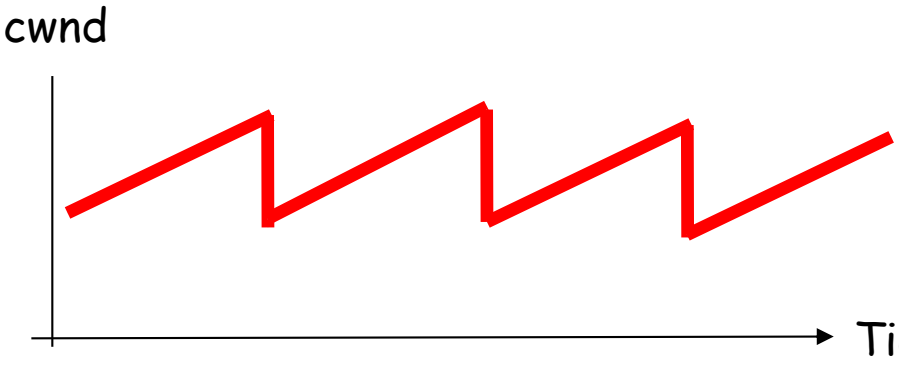
Pérdidas (TCP)

- Si la ventana de control de flujo es suficientemente grande
- Y la transferencia dura suficiente
- TCP aumenta cwnd hasta congestionar el camino
- Se producen pérdidas y baja agresivamente la tasa de envío
- Y repite (más lento)
- Aproximación al throughput promedio para conexiones largas (TCP Reno) con una tasa *p* constante y pequeña de pérdidas
- TCP es muy sensible (agresivo) ante las pérdidas

$$BW = \frac{MSS}{RTT}\sqrt{\frac{3}{2p}}$$

M. Mathis
(ACM SIGCOMM'97)





TCP Reno en
congestion avoidance

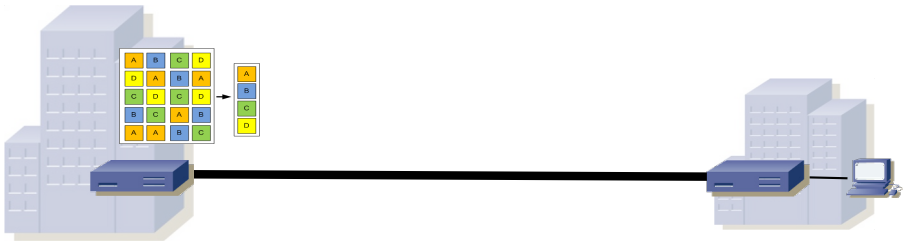
Técnicas de WAN opt.

Compresión

- Compresión sin pérdidas de los datos a enviar
- Por ejemplo, si el navegador lo soporta, el servidor web puede enviar comprimido
- En caso de no aceptarlo el cliente, podría ser anunciada la opción por un proxy
- Según las aplicaciones, puede ser útil en ambos sentidos

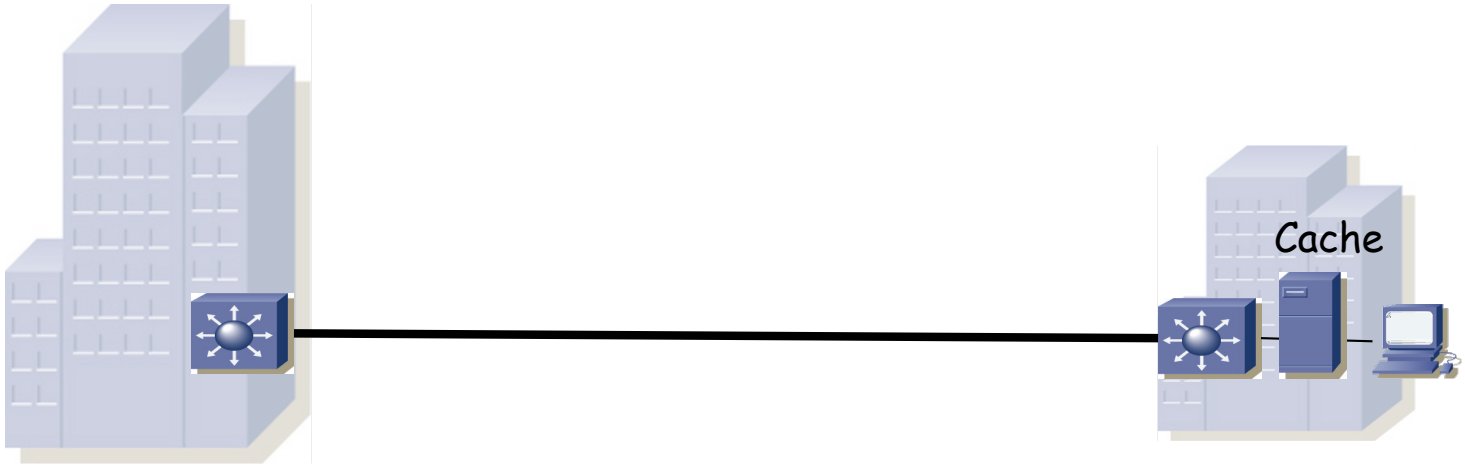
Data deduplication

- Detectar patrones que se han enviado antes
- No enviarlos de nuevo sino un identificador
- El otro extremo recrea el patrón
- Puede ser a nivel de no volver a enviar el mismo fichero
- O detectar cambios en trozos en el mismo y solo enviar los cambios
- Puede ser entre diferentes aplicaciones: por ejemplo descargar un fichero para luego adjuntarlo a un e-mail
- No es sencillo por los cambios en los protocolos de transporte y aplicación
- Los firewalls no son útiles con el tráfico desduplicado



Latencia: Caches

- Sencillas para protocolos como HTTP que están diseñados con ellas en mente
- Factibles pero más complejas para protocolos de compartición de ficheros (por ejemplo SMB)
- Caches para diferentes aplicaciones pueden acabar manteniendo los mismos objetos, ejemplo:
  - Usuario descarga un fichero de un email
  - Lo carga a un directorio compartido (SMB)
  - Lo sube a una web (HTTP)



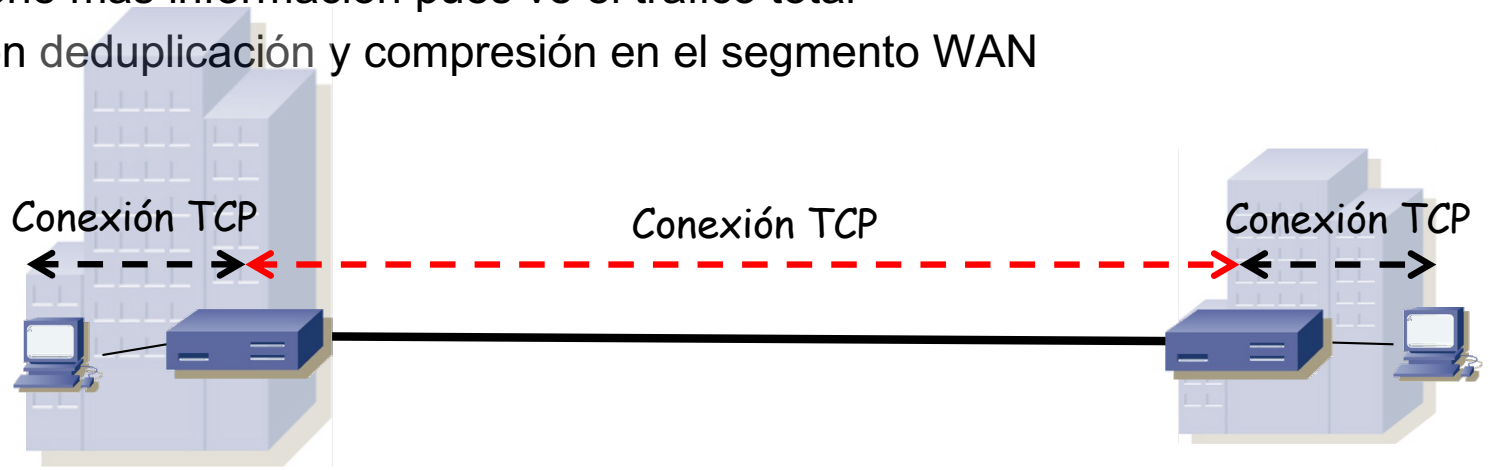
Aceleración

Application Acceleration

- Conoce el protocolo de aplicación
- Se anticipa a las acciones del usuario
- Por ejemplo puede hacer un *read-ahead* cuando la aplicación está leyendo un documento
- O puede conocer las secuencias típicas de mensajes e intentar optimizarlas

TCP acceleration

- Equipos con versiones antiguas o no optimizadas de TCP, las ecualiza
- El optimizador puede modificar el tráfico implementando nuevos mecanismos de control de congestión, mayores valores de ventana (*virtual window scaling*), etc
- O por ejemplo rompiendo la conexión en tres
- Tiene más información pues ve el tráfico total
- Con deduplicación y compresión en el segmento WAN

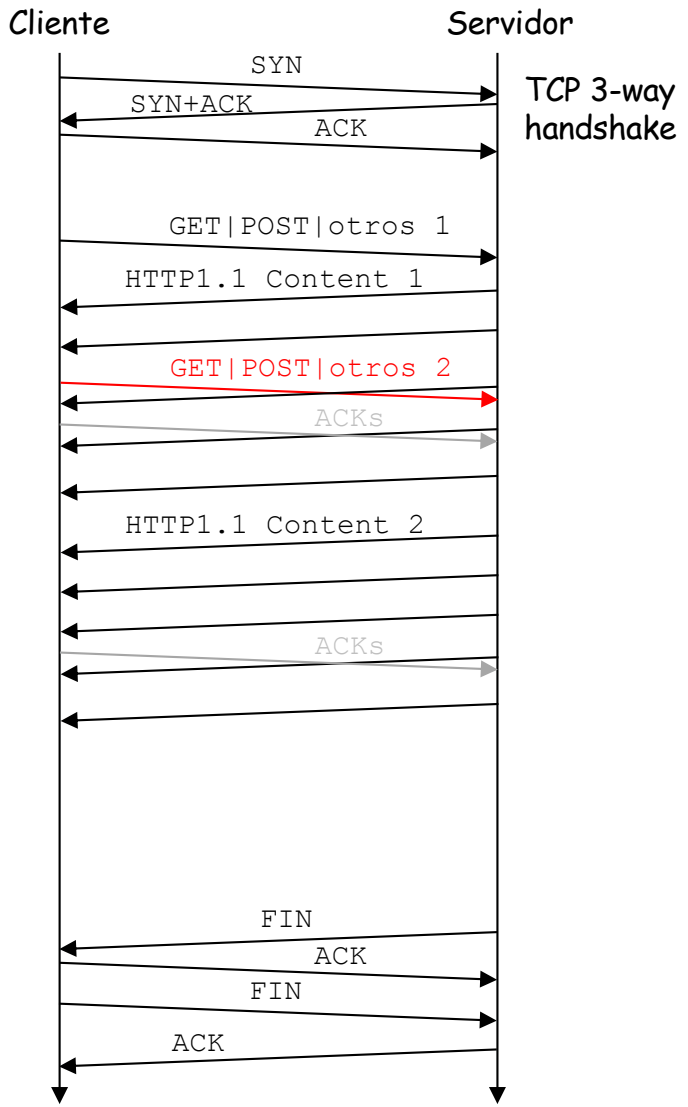


Nuevos protocolos

HTTP 1.1

- Conexiones persistentes
- Pipelining: respuestas en el mismo orden que las peticiones
- Posible entrega de rangos de bytes

- RFC original 2616 (junio 1999)
- RFC 9112: HTTP/1.1 (junio 2022) deja obsoleta la RFC 7230 (y ésta la 2616)

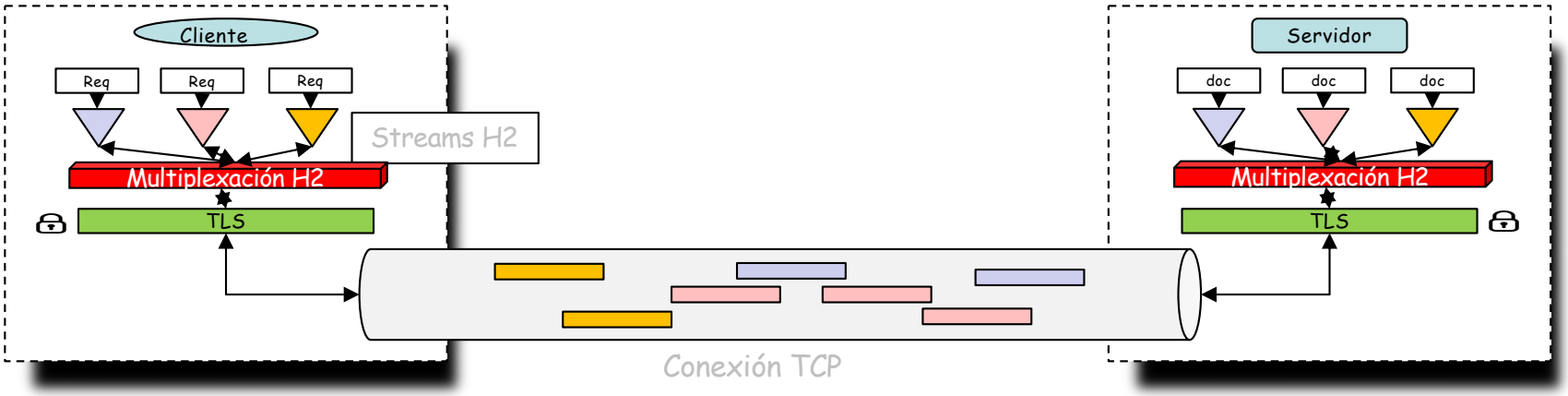


HTTP1.1 - Problemas

- “Hacks” de navegadores y desarrolladores
  - Conexiones en paralelo (limitadas a 5-6 para un servidor)
  - *Domain sharding* (recursos en diferentes dominios)
    - Ambas crean mayor número de sockets, más conexiones en NATs, más DNS
  - Unión de ficheros pequeños (CSS, Javascript, imágenes)
  - Inline de ficheros con el HTML
    - Concatenación e inlining rompen la modularidad y entorpecen las caches
- Optimizado para grandes transferencias, pero una web contiene muchos recursos pequeños
- Pipelining no suele emplearse (problemas con proxies y algunos servidores)
- Aún con pipelining tiene **HOL blocking** (hasta completar respuesta grande no se atiende a otra del pipeline)
- Para mejorar el tiempo de carga hay que reducir el RTT (CDNs), pero esto tiene un límite (“c”)
- Reducir RTT reduciendo *chattiness*
- *TLS* añade al menos 1 RTT
  - Identificación de capacidades (ciphers, versiones)
  - Intercambio de certificados y de información para la construcción de claves

HTTP/2 (H2)

- RFC 9113 “HTTP/2” (Mozilla, Apple, 2022, RFC 7540 de 2015)
- Binario
  - Ya no es texto, más compacto, más eficiente al procesarlo
  - Se pueden comprimir las cabeceras
- Multiplexación de peticiones y respuestas
  - No es necesario completar una respuesta para empezar a enviar otra
  - Trabaja con *streams* dentro de una sola conexión
  - El navegador puede asignarles prioridades en las peticiones
  - Esto elimina el HOL blocking y la necesidad de conexiones en paralelo
  - Frames (nombre en H2) de los diferentes streams HTTP/2 en mensajes *Application Data* de TLS enviados por la conexión TCP
  - Frame HEADERS contienen la cabecera HTTP
  - Frame DATA con el contenido de la respuesta (o del POST)



HTTP/2 (H2)

- Server Push
  - El servidor puede entregar recursos al cliente aunque no los haya pedido
  - Ahorra la petición (esto se estaba haciendo con el *inlining*)
- Flow Control
  - Para cada *stream*
  - HTTP/2 no fuerza a un algoritmo para el control de flujo, solo lo soporta
- Negociación
  - Mediante mensajes se puede subir una conexión de HTTP1.1 a HTTP/2
  - Obsoleto desde RFC 9113 (2022)
  - Hoy en día solo se está usando HTTP/2 sobre TLS
  - Negociación mediante el ALPN en TLS

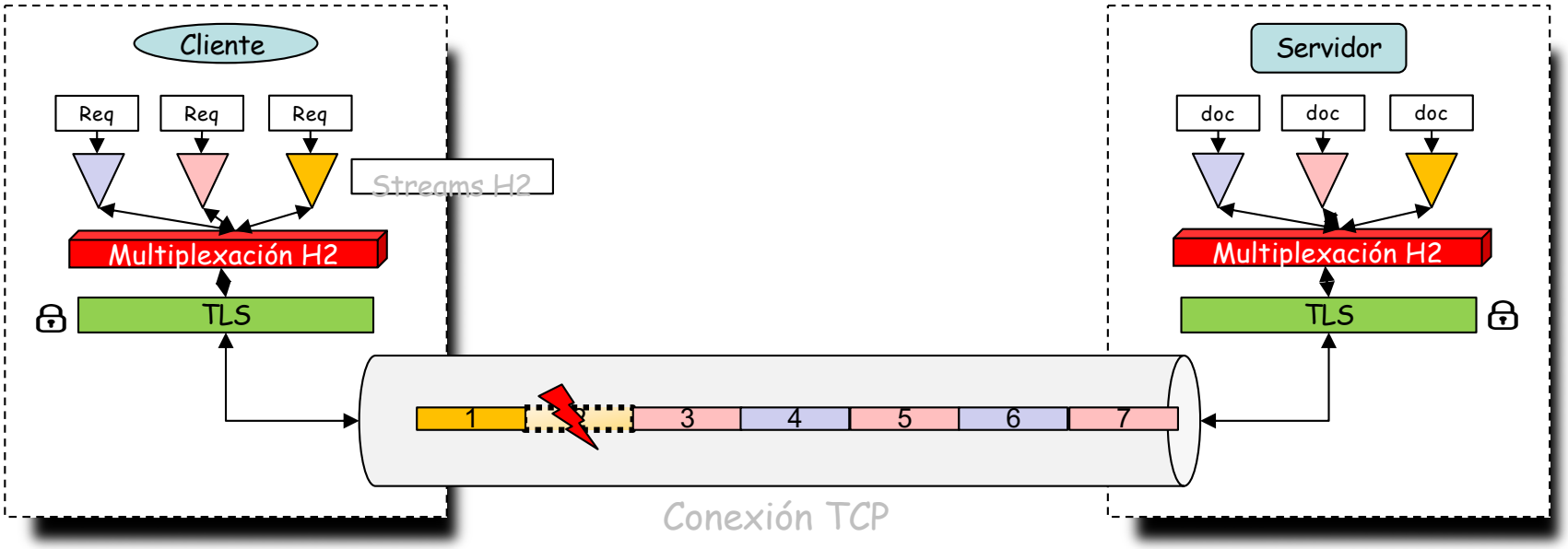
HTTP1.1 – HTTP/2

- Conexiones en paralelo
  - Consume más recursos en cliente, servidor, NATs, firewalls, etc
  - Acelera descarga por evitar HoL blocking
  - En H2 la multiplexación de streams resuelve el HoL blocking
- *Domain sharding* (recursos en diferentes dominios)
  - Era para aumentar las conexiones en paralelo pero ahora están los streams
- Unión de ficheros pequeños (CSS, Javascript, imágenes)
  - Evitaba múltiples peticiones con problema HoL que ya está resuelto en H2
- Inline de ficheros con el HTML (imágenes)
  - Evitaba tener que pedir ese recurso
  - Ahora se puede “empujar” (*push*) desde el servidor

- Otras mejoras:
  - Binario
  - Compresión de cabeceras
  - Control de flujo por stream
  - Prioridades

HTTP/2 (H2)

- Emplea una conexión TCP, con 2 streams (uno en cada sentido)
- Cada stream es un flujo bytes ordenados
- Ante una pérdida
  - No se entrega a la aplicación el resto de datos hasta “rellenar el hueco”
  - La pérdida ha podido afectar solo a paquetes de un stream h2
  - Pero afecta a todo el stream TCP y por lo tanto a todos los streams h2
  - Además detiene todo el flujo (retransmisiones y control de congestión)
  - Resuelto HOL blocking en nivel de aplicación pero no de transporte



RTTs

- 1 RTT establecimiento de la conexión TCP
  - Aunque tenemos TCP Fast Open (RFC 7413)
    - Permite entregar a la aplicación datos que llegan con el SYN
    - Pero tiene escaso despliegue (modifica TCP y problemas con middleboxes)
- 2 RTTs para establecer la sesión TLS
  - Aunque tenemos
    - Sesiones, tickets
    - 0-RTT con TLS 1.3 (Early data)
- La aplicación no puede sacar provecho a múltiples interfaces en el host

¿Soluciones?

- Cambiar el nivel de transporte
- RFC 4960 Stream Control Transmission Protocol
- Diseñado para el transporte de la señalización de la red telefónica
- Ofrece:
  - Transporte fiable (acknowledged error-free non-duplicated)
  - Multiplexación de sub-streams
  - Transporte ordenado dentro de cada stream
  - Entrega mensajes en lugar de un byte stream
  - Segmentación
  - Multi-homing
  - Congestion avoidance
  - Flow control
- Problema con SCTP
  - Implementarlo en los sistemas operativos de los hosts
  - Implementarlo en los NATs
  - Diferente API

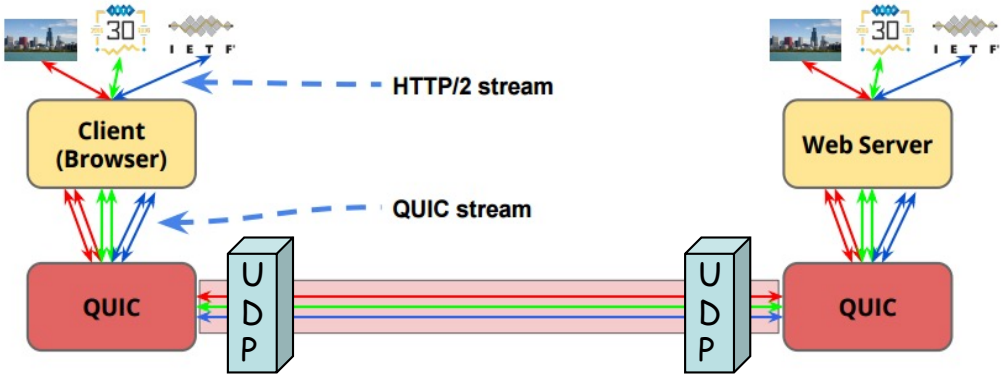
upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

Redes de Nueva Generación
Área de Ingeniería Telemática

HTTP/3

HTTP/3

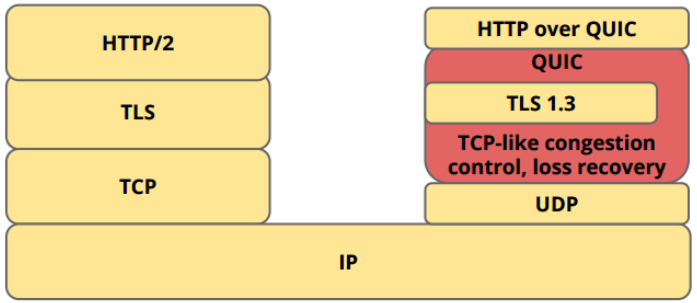
- QUIC (Quick UDP Internet Connections)
  - Inicialmente protocolo experimental de Google (2014)
  - RFC 9000 “QUIC: A UDP-Based Multiplexed and Secure Transport”
- RFC 9114 “HTTP/3” (Akamai, Junio 2022)
- *“This document describes a mapping of HTTP semantics over QUIC.”*
- HTTP/2 + QUIC = HTTP/3
- Implementado sobre UDP
  - Para poder atravesar NATs y otros middleboxes
  - En la aplicación en lugar de ser parte del kernel (despliegue más rápido)
  - En algunas redes puede estar filtrado (fallback a TCP+TLS)
  - Timers en middleboxes menores que para TCP
- Sub-streams (elimina HOL blocking) fiables, ordenados



https://youtu.be/ZF6-KmHz1sU
https://www.ietf.org/proceedings/98/slides/slides-98-edu-sessf-quic-tutorial-00.pdf

Características de QUIC

- Puede establecer una conexión QUIC en 0 RTTs
  - Si ha establecida una conexión previa que ofrezca las credenciales
  - Manda datos con el paquete para establecer la conexión
  - Si no ha establecido antes una conexión entonces sí gasta 1 RTT
  - 2 RTTs si tiene que negociar versión
- Mejora la recuperación ante pérdidas
  - No hay ambigüedad en las retransmisiones
  - Da timestamp de llegada de datos en el ACK
  - Permite más rangos confirmados en SACKs



- Más flexible control de congestión
- QUIC del IETF ligeramente diferente a la versión de Google
- TLS 1.3 ofrece el handshake en 0 RTTs
- La encriptación oculta QUIC a los equipos de red y lo limita a los extremos
  - Middleboxes no podrán basarse en ello (proxy transparente)
  - Impide la solidificación del protocolo debido a middleboxes
  - QUIC versión 2 para evitar la ossification (en curso)
  - Puede ser un problema para ISPs que quieran inspeccionar cabeceras

QUIC hoy

- RFC 9001 “Using TLS to Secure QUIC”
- RFC 9002 “QUIC Loss Detection and Congestion Control”
- RFC 9114 “HTTP/3” (junio 2022)
- RFC 9221 “An Unreliable Datagram Extension to QUIC”
- RFC 9250 “DNS over Dedicated QUIC Connections”
- RFC 9369 “QUIC Version 2” (dic 2023)
- Drafts
  - “Multipath Extension for QUIC”
    - draft-ietf-quic-multipath-06, Oct. 2023 (Alibaba, Uber, Univ. Of Mons, UCLouvain, Tessares, Private Octopus, Ericsson)
  - “RTP over QUIC”
    - draft-ietf-avtcore-rtp-over-quic-08, Feb 2024 (Tech. Univ. Munich, Tencent America LLC)
  - “BGP over QUIC”
    - draft-retana-idr-bgp-quic-03, Octubre 2023 (Futurewei, Juniper, Huawei, Nvidia)

https://datatracker.ietf.org/wg/quic/
https://httpwg.org/specs/

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

MPTCP

Problemas de partida

- La separación entre TCP e IP no es completa
- Una conexión TCP viene asociada a la 5-tupla, lo cual implica estar asociada a las direcciones IP
- Una conexión TCP no puede mantenerse ante el cambio de las direcciones de nivel de red
- Soluciones en capa 3
  - Mobile IP (RFC 5944), HIP (Host Identity Protocol, RFC 4423), Shim6 (Site Multihoming by IPv6 Intermediation, RFC 5533)
  - Ocultan a TCP el cambio de dirección, con lo que ocultan el cambio de camino al control de congestión
- SCTP (Stream Control Transmission Protocol, RFC 4960)
  - Protocolo de nivel de transporte que soporta múltiples direcciones IP por conexión de transporte
  - Despliegue imposible por falta de soporte en NATs (SCTP over UDP?)
  - API diferente para las aplicaciones
- RFC 8684 “TCP Extensions for Multipath Operation with Multiple Addresses” (Pexip, U.Pol. Bucharest, UCL, U.Cath. Louvain, Apple, Marzo 2020), MultiPath TCP (MPTCP)

Middleboxes

- Descartan paquetes de otros protocolos de transporte
- Modifican cabeceras IP y TCP (NAT, proxy transparente)
- Modifican ventana de control de flujo (control de BW, escalado)
- Modifican números de secuencia (ej: ISN aleatorio)
- Eliminan opciones que no conocen
- Abortan conexiones con opciones que no conocen
- Descartan paquetes si no han visto el inicio de la conexión
- Hacen coalescencia o segmentación de paquetes
- Modifican el flujo de datos (ALGs)

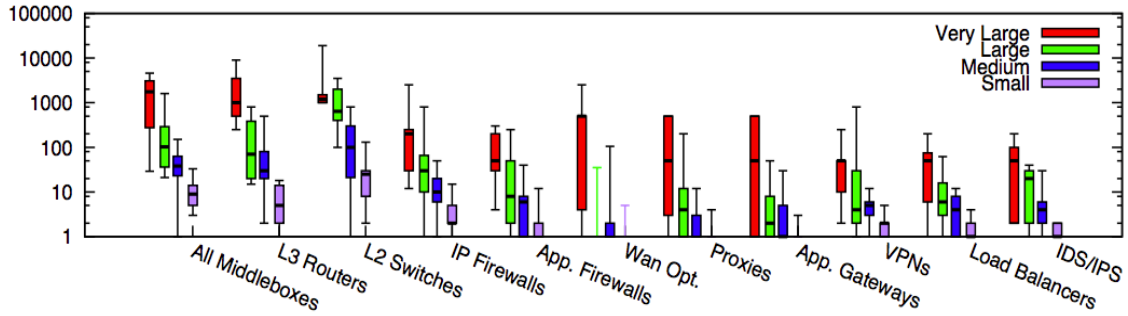


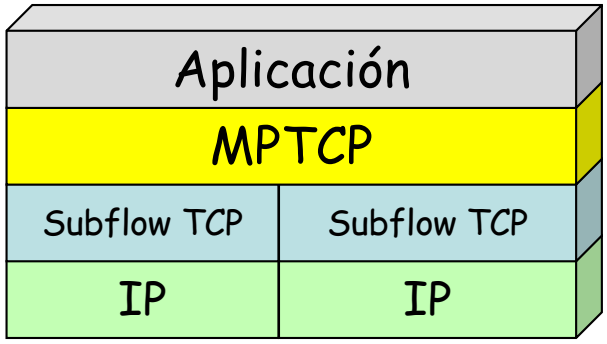
Figure 1: Box plot of middlebox deployments for small (fewer than 1k hosts), medium (1k-10k hosts), large (10k-100k hosts), and very large (more than 100k hosts) enterprise networks. Y-axis is in log scale.

Sherry, Justine, et al. "*Making middleboxes someone else's problem: Network processing as a cloud service.*" Proceedings of the ACM SIGCOMM 2012 conference. ACM, 2012.

- Hay más middleboxes que routers: Firewalls, balanceadores, VPN concentrator, SSL terminador, IP telephony router ...

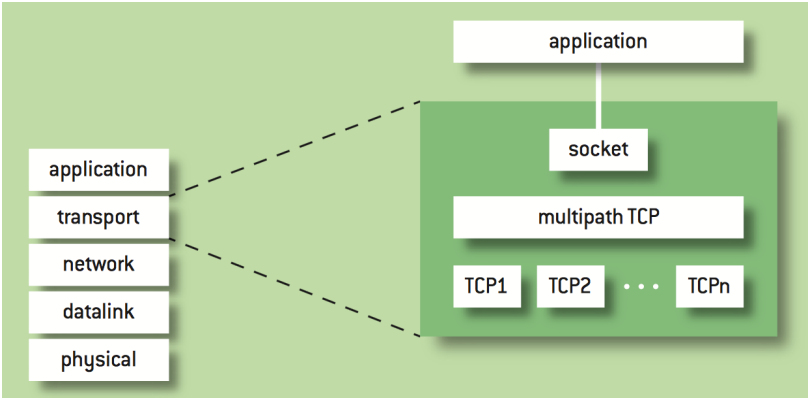
Objetivos de MPTCP

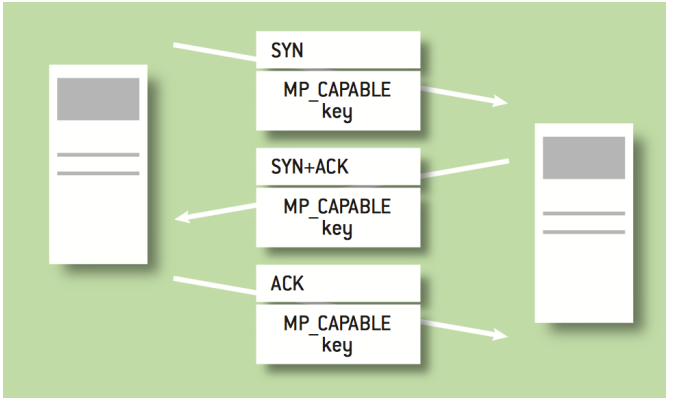
- Emplear múltiples caminos en paralelo para una misma conexión
  - WiFi y Ethernet
  - Múltiples interfaces Ethernet en servidores
  - Múltiples caminos en el datacenter
  - Failover
- Emplearlos tan bien como TCP, siendo TCP-friendly
  - Que no ahogue a otros flujos TCP
- Usable como TCP tradicional (API)
- Si TCP funciona en un camino entonces habilitar MPTCP no debe impedir la comunicación



Arquitectura

- MPTCP crea subflujos que son conexiones TCP
- Los hosts extremos mantienen estado que une esos flujos
- MPTCP actúa como una capa intermedia entre la aplicación y el nivel de transporte
- La señalización adicional se logra mediante opciones TCP
- Su mayor problema es ser resistente ante los diversos comportamientos de middleboxes
- Negociación mediante una opción TCP nueva (*MP_CAPABLE*) anunciada en el establecimiento de la conexión TCP inicial
- Incluyen una clave
- Middleboxes pueden eliminar la opción (entonces solo TCP)
- Middleboxes pueden descartar los paquetes por no reconocer la opción (entonces reintentar sin ella y solo TCP)

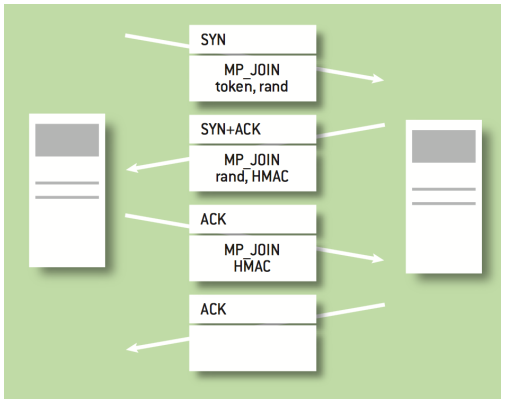




C. Paasch y O. Bonaventure, “Multipath TCP”, ACM Queue, vol. 12, n.2, Marzo 2014

Subflujos

- Una nueva conexión se añade como subflujo a la primera
- Puede estar iniciada desde otro interfaz
- No se pueden identificar las conexiones mediante la 4-tupla por la posible presencia de NATs
- Se identifica la conexión MPTCP a la que unirse mediante la opción MP_JOIN con un token generado a partir de la clave
- Se puede enviar los datos de la aplicación por cualquiera de los subflujos
- Para hacer una entrega en orden hace falta numerar los datos independientemente del subflujo
- ¿Números de secuencia de segmentos TCP sean del flujo MPTCP?
  - Habría huecos en la secuencia de un subflujo: algunos middleboxes (DPI) no lo soportan (tal vez aborten la conexión con un RST generado por ellos)
  - Hay middleboxes que cambian el ISN en las conexiones TCP
  - Hay middleboxes que dividen los segmentos (ej: NICs con TSO)



C. Paasch y O. Bonaventure, “Multipath TCP”, ACM Queue, vol. 12, n.2, Marzo 2014

Subflujos

- Se mantienen los números de secuencia independientes para cada subflujo
- Secuencia global de MPCTP de 64 bits
- Relación con secuencia global va en una opción TCP (DSS)
- En la opción va un offset respecto al inicio del subflujo
- Un checksum permite detectar que un middlebox haya introducido cambios en los datos (ALG) y se termina el subflujo
- En cada subflujo hay confirmaciones y retransmisiones mediante TCP (SACK, fast retransmit, RTO, etc)
- Se debe retx por el subflujo donde se produjo la pérdida
- Se puede enviar la retransmisión también por el otro subflujo
- Confirmación de la secuencia global (ACK acumulado) mediante la opción DSS (Data Sequence Signal)
- Los datos enviados no pueden liberarse hasta ser confirmados tanto en el DSS como por los ACK del subflujo
- ACK global sirve como inicio de la ventana de control de flujo
- La ventana de control de flujo es compartida entre los subflujos
- Es decir, la ventana anunciada en un subflujo es la global
- Emisor emplea la mayor de las anunciadas por los subflujos (un middlebox podría estar cambiando una de ellas)

Control de congestión

- Cada subflujo TCP tiene su cwnd
- Se debe acoplar el control de flujo de los subflujos o si no habrá reparto injusto con TCP tradicional
- Ahora un flujo de la aplicación son varios de nivel de transporte y el reparto en los cuellos de botella suele ser por esos flujos
- Se desea que si múltiples subflujos pasan por el mismo cuello de botella empleen tanta capacidad como un solo flujo TCP
- En la RFC queda abierto
- Una propuesta: RFC 6356 (Experimental) “Coupled Congestion Control for Multipath Transport Protocols”

DNS

DNS over TLS (DoT)

- RFC 7858: “Specification for DNS over Transport Layer Security (TLS)”
- USC/ISI, ICANN, Mayo 2016
- TCP, puerto 853

DNS over QUIC (DoQ)

- RFC 9250: “DNS over Dedicated QUIC Connections”
- Private Octopus, Sinodun IT, Salesforce
- Encriptación como DoT pero latencia como DNS over UDP
- UDP port 853

DNS over HTTPS (DoH)

- RFC 8484: DNS Queries over HTTPS
- ICANN, Mozilla, Octubre 2018
- Petición DNS va en petición HTTP (GET o POST)
- El servidor DoH define el URI y el template para las peticiones
- Respuesta de tipo application/dns-message

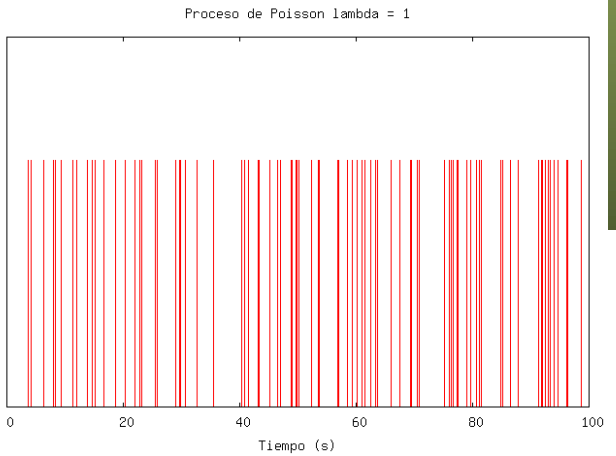
upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa
Redes de Nueva Generación
Área de Ingeniería Telemática

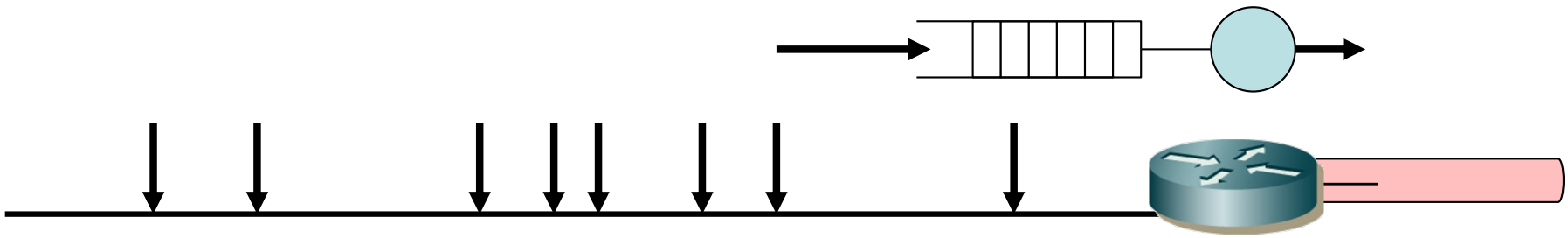
Tráfico de datos

Poisson para datos

- En telefonía modelo de llegadas de Poisson
- En datos las llegadas no serían de llamadas sino de paquetes
- La duración es de los paquetes (tiempo de transmisión)
- Solo hay un canal de salida, el enlace de salida
- Ahora no podemos emplear un modelo Erlang-B
- No se descarta una nueva llegada ante que el canal de salida esté ocupado (no *blocked calls cleared*)
- Se encolan; teoría de colas

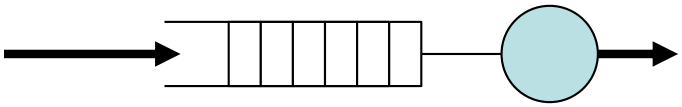
$$P[N = k] = \frac{(\lambda \Delta t)^k}{k!} e^{-\lambda \Delta t}$$





Poisson para datos

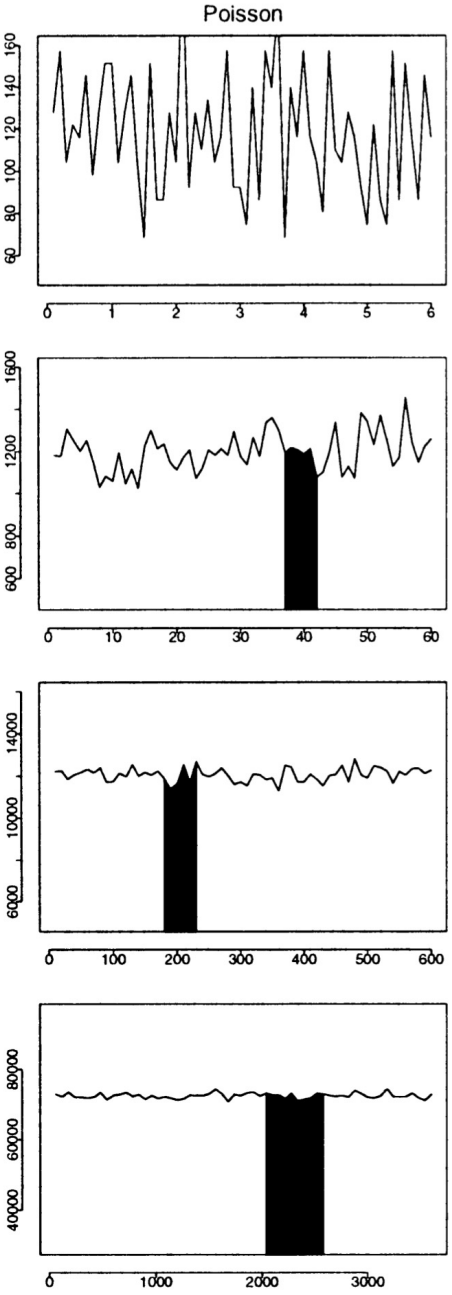
- ¿Podemos resolver sistemas de colas con llegadas de Poisson?
- Calcular probabilidades de pérdidas (ante cola llena)
- Calcular distribuciones de tiempo de espera en cola
- Sí, se puede
- ¿Es útil? No
- ¿Por qué?
- Porque el tráfico no es de Poisson



¿Qué tiene de bueno Poisson?

1. Matemáticamente tratable
   - M/M/1 y familia
   - Si fuera de Poisson podríamos dar factores de utilización del 99% con retardos en cola por debajo de 1ms
2. Se suaviza al agregar
   - Dimensionar ligeramente sobre la media

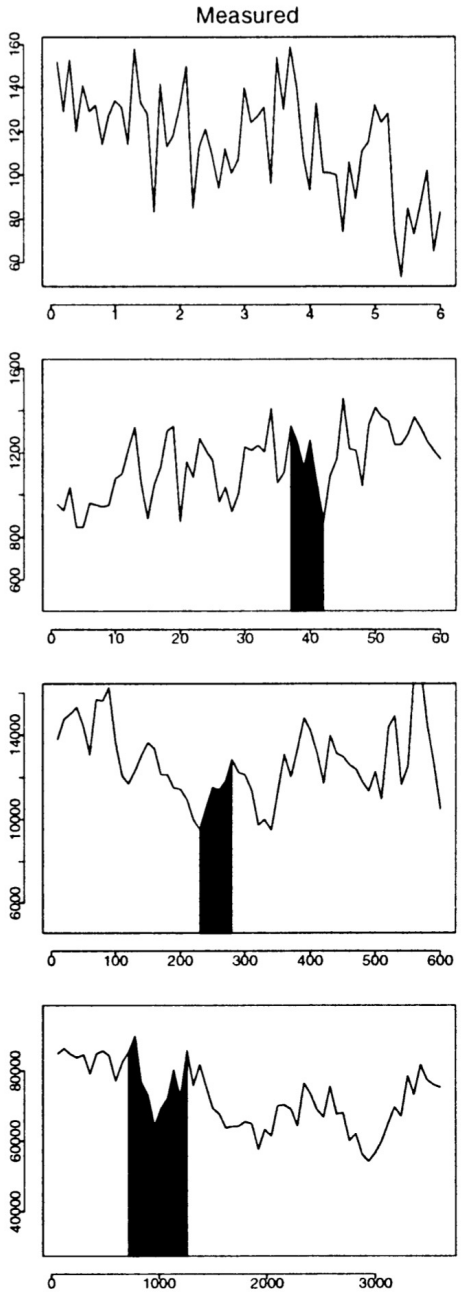
- ¿Podemos resolver sistemas de colas con llegadas de Poisson?
  - Calcular probabilidades de pérdidas (ante cola llena)
  - Calcular distribuciones de tiempo de espera en cola
- Sí, se puede pero no es útil porque el tráfico de datos no es poissoniano



W. Willinger and V. Paxson, "Where Mathematics Meets the Internet", Notices of the AMS, Vol 45, No.8, P961-970, 1998

¿Cómo es el tráfico?

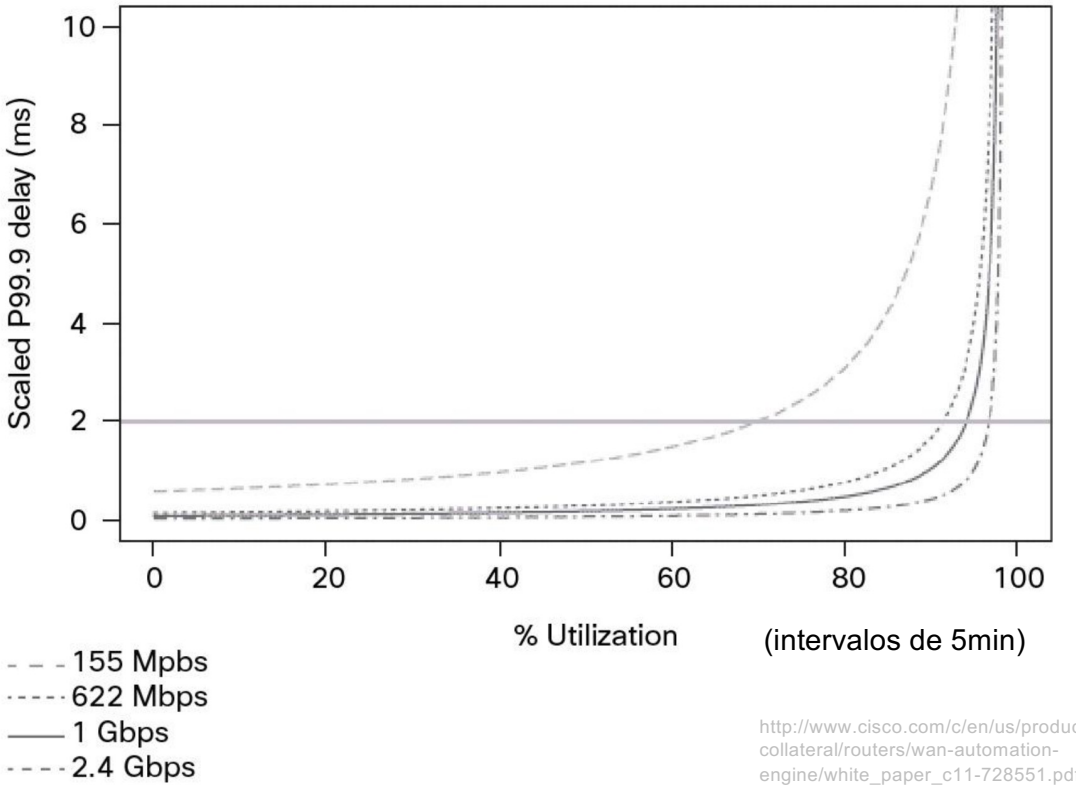
- Mantiene ráfagas (*burstiness*)
- En todas las escalas de agregación
- Con tráfico de Poisson teníamos que considerar muy pequeña variación con alta agregación
- Ahora con el nivel de agregación la variación no se reduce tan rápido
- Para mantener un retardo bajo debemos trabajar en factores de utilización moderados (¡nada del 99%!)
- Modelos auto-similares, fractales
- Discutidos, lo que importan son las escalas bajas, modelos multi-fractales (¡!)
- Aproximaciones numéricas por simulación



W. Willinger and V. Paxson, "Where Mathematics Meets the Internet", Notices of the AMS, Vol 45, No.8, P961-970, 1998

Ejemplo de aproximación

- Thomas Telkamp, “Traffic Characteristics and Network Planning”, presentación en el NANOG 26, 2002
- Intenta mejorar reglas como “deberías aumentar la capacidad del enlace si su utilización llega al 50%”
- Se obtienen (para el mismo factor de utilización) menores retardos cuanto mayor sea la capacidad del enlace (mayor grado de mux.)
- Es decir, aunque no sea tráfico de Poisson sí hay ganancia al agregar, reflejada en menor *overprovisioning* necesario

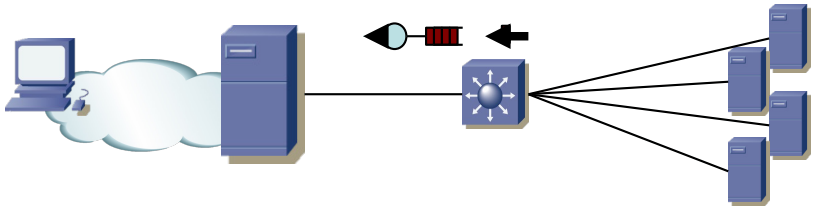


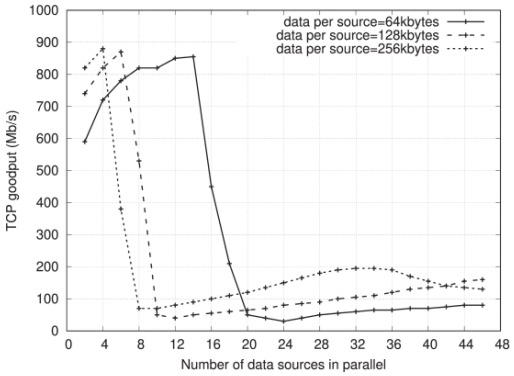
(intervalos de 5min)

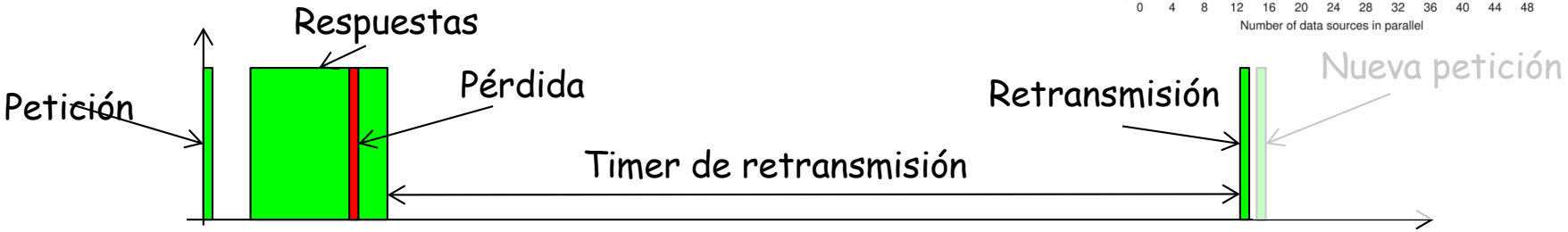
http://www.cisco.com/c/en/us/products/collateral/routers/wan-automation-engine/white_paper_c11-728551.pdf

Incast: Fenómeno

- Aplicaciones con esquema *Partition+Aggregate* y *barrier synchronized*
- Ejemplo:
  - Cliente hace una petición simultáneamente a un gran número de servidores
  - Suelen ser múltiples conexiones TCP
  - Los servidores contestan prácticamente a la vez
  - El tráfico puede saturar el buffer del puerto hacia el cliente y produce pérdidas
- Las conexiones que sufren pérdidas, si no tienen más datos para enviar, deben recuperarlas mediante timeout
- El timeout suele estar en el rango de 200-300ms
- El cliente no hace nuevas peticiones hasta completar la respuesta de la anterior
- Pero la fase de transferencia ha podido durar unos pocos ms
- El cliente puede poner un deadline y construir sus respuesta con información incompleta







Incast: Causas

- Ejemplo: Cisco Nexus 3048
- RTO con valor mínimo en 200-300ms
- De hecho el mínimo en la RFC es 1s
- ¿Por qué? TCP está diseñado para la Internet
- ¿RTTs en el datacenter? Unos 100µs (buffer vacío)
- ¿Cómo? 1518bytes a 1Gbps son 12µs, eso 4 veces (1 solo salto por conmutador) son ya unos 50µs
- ¿Y con buffer lleno? Depende de la “profundidad” del buffer
  - Ejemplo: 100Kbytes por puerto
  - Suponiendo solo el puerto hacia el cliente saturado
  - RTT de menos de 1ms (100 x 1024 x 8 / 1000000 = 0.8192 ms)

Incast: Soluciones

- Aumentar el buffer del conmutador
  - Mayor coste
  - Ojo, ¿quieres 9GBytes en un puerto? Si se llenan dan un retardo en cola (con puerto de 10Gbps) de… ¡ 7 segs !
- Mejorar la gestión de los buffers
  - No suelen estar congestionados todos los puertos
  - Repartos dinámicos de memoria
- Reducir el RTO mínimo
  - Requiere timers de alta resolución
  - En entorno WAN puede llevar a retransmisiones espúreas
- Reaccionar ante la congestión en el nivel de enlace (ej: QCN)
- Reducir el tamaño de las respuestas
  - A nivel de aplicación (también introducir jitter entre ellas)
  - Estos flujos suelen convivir con *long-lived flows*
- Modificar el control de congestión de TCP
  - DCTCP (Data Center TCP): Requiere ECN
  - ICTCP (Incast congestion Control for TCP)
  - Otros: IA-TCP, D^2TCP, TCP-FITDC, TDCTCP, etc.

Incast: Afectados

- Aplicaciones para las que sucede
  - MapReduce
  - Cluster storage
  - Web search (basado en partition+aggregate)
  - Composición de contenido en redes sociales
- En cualquier caso, depende del escenario que se produzca el fenómeno o no
  - Número de emisores
  - Tamaño de bloque que envían
  - Sincronización en el envío
  - Tamaño y gestión del buffer del conmutador
  - Etc.

TCP Outcast

- Partition+Aggregate, drop-tail
- Más flujos de servidores alejados que de cercanos, por puertos diferentes
- Congestión en el cuello de botella➔buffer lleno ➔ pérdidas
- Lo más probable es que se de una llegada en ráfaga y se pierda entera
- Esa ráfaga probablemente viene de uno de los puertos de entrada
- O es el mux de paquetes de los dos
- En enlace con muchos flujos la ráfaga de pérdidas se distribuye entre flujos
  - Baja probabilidad de perder toda una ventana y acabar en RTO
- En enlace con pocos flujos se distribuye entre pocos flujos
  - Mayor probabilidad de acabar en un RTO
- El resultado es que flujos con menor RTT obtienen menor goodput (inverso de lo habitual)