Workshop
VXLAN & EVPN

@arielweher
@lacnog

"Muchos problemas se pueden solucionar añadiendo una capa más de complejidad"

(Menos el problema de tener demasiadas capas)

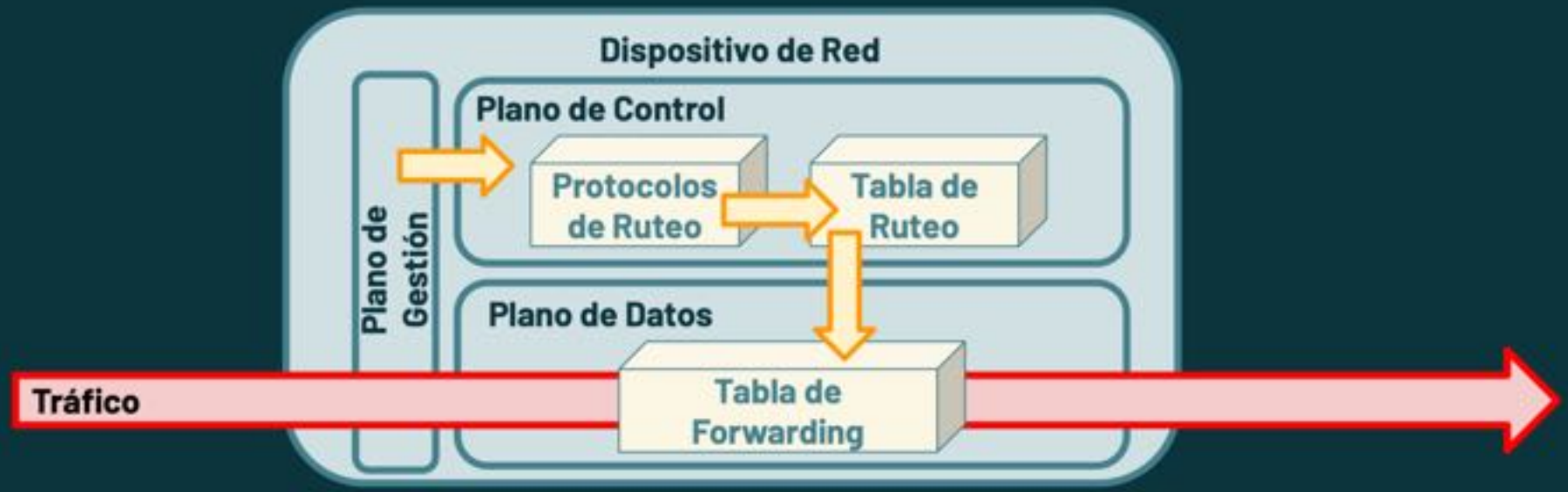
Intro

Algunos conceptos básicos

Data Plane & Control Plane

En networking, un Plano es una concepción abstracta de dónde tienen lugar ciertos procesos.

Los dos planos a los que se hace referencia con mayor frecuencia en las redes son el ***Control Plane*** y el ***Data Plane*** (también conocido como ***Forwarding Plane***).



Control Plane

El plano de control se refiere a todas las funciones y procesos que determinan qué ruta utilizar para enviar un paquete o la trama.

Generalmente se considera que el plano de control es donde un router o switch toma sus decisiones. Está basado en software y utiliza la CPU en lugar de hardware especializado, (ASIC). Por esta razón, en mucha documentación se llama al plano de control el "camino lento".

Data Plane

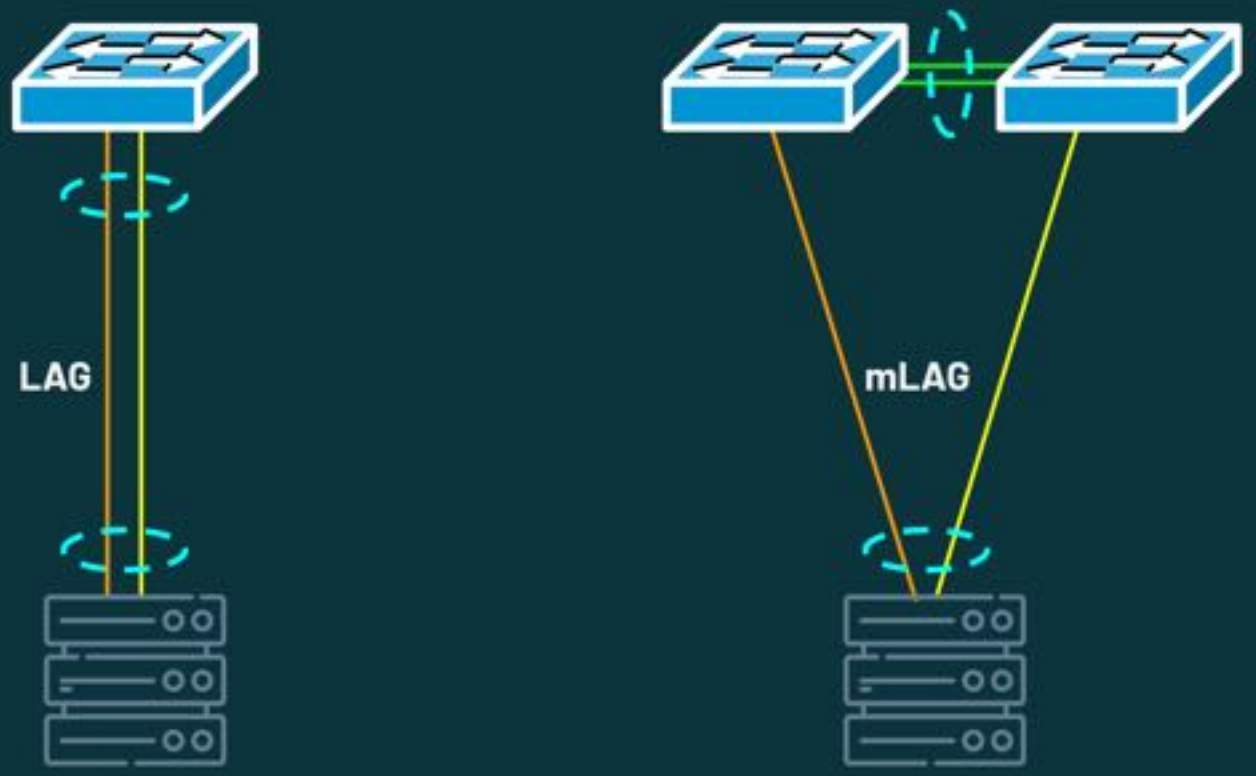
El plano de datos (o plano de forwarding) es la ruta de alta velocidad a través del router/switch. Los paquetes que pasan a través del dispositivo utilizan el plano de datos, a diferencia de los paquetes dirigidos al dispositivo.

El plano de datos debe proporcionar una ruta de alta velocidad y baja latencia. Para lograr esto, gran parte de la implementación del plano de datos está en el hardware. Esto incluye realizar búsquedas/almacenamiento en caché y usar ASIC especiales.

Desafíos de las redes de Data Center

- **Incremento en la comunicación entre dispositivos internos**
  - Los sistemas monolíticos se desarmaron en cientos o miles de microservicios.
  - Según un estudio de Cisco, el 76% del tráfico de un DC es de tipo Este-Oeste.
- **Escalabilidad**
  - Los centros de datos se vuelven cada vez más grandes a medida que la "nube" crece en usuarios.
  - La red debe permitir la comunicación de todos estos equipos sin perder performance mientras que debe mantener un costo eficiente.
- **Resiliencia**
  - Las redes modernas están diseñadas para poder seguir trabajando ante la presencia de fallos.
  - Se debe asumir que los componentes van a fallar, y la operación debe continuar sin afectar la experiencia de uso de los clientes.

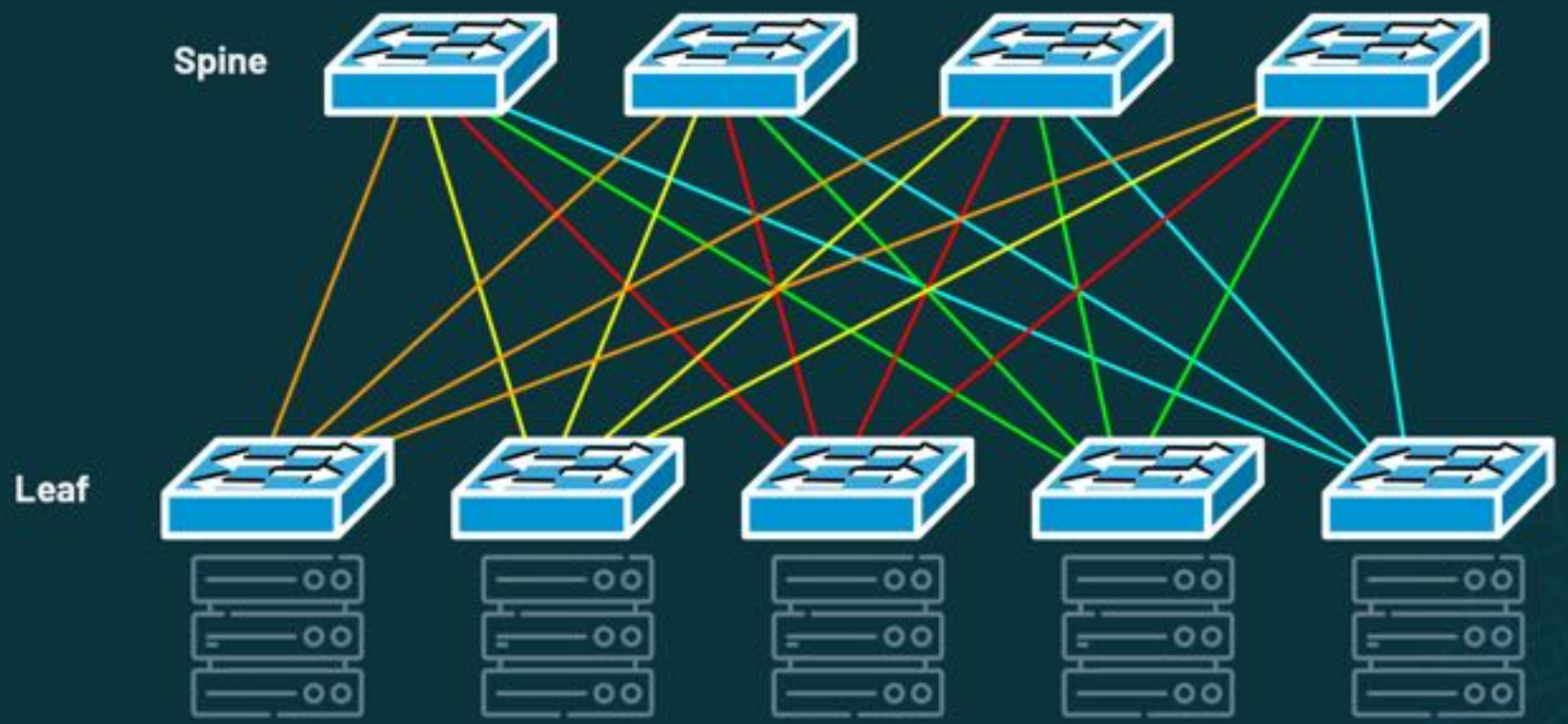
Tecnologías de Acceso: LAG y multi chassis LAG



Redes Clos

- Una red Clos es un tipo de arquitectura de forwarding sin bloqueo que reduce la cantidad de puertos necesarios en una estructura totalmente interconectada.
- La red Clos fue inventada por Edson Erwin en 1938 y formalizada por primera vez por Charles Clos en 1952.
- La ventaja clave de las redes Clos es que el número de puntos de cruce requeridos puede ser mucho menor de lo que sería si todo el sistema de forwarding se implementase con un switch grande.
- Una topología de Clos se compone de capas "Leaf" y "Spine".
- Los servidores están conectados a los switches *Leaf* y cada *Leaf* está generalmente conectado a todos los *Spine*.
- No hay una conexión directa de *Leaf* a *Leaf* o de *Spine* a *Spine*.
- Hay exactamente tres saltos de switch entre cada servidor.
- Los servidores se pueden conectar a varios *Leaf* usando mLAG para obtener redundancia.

Redes Clos



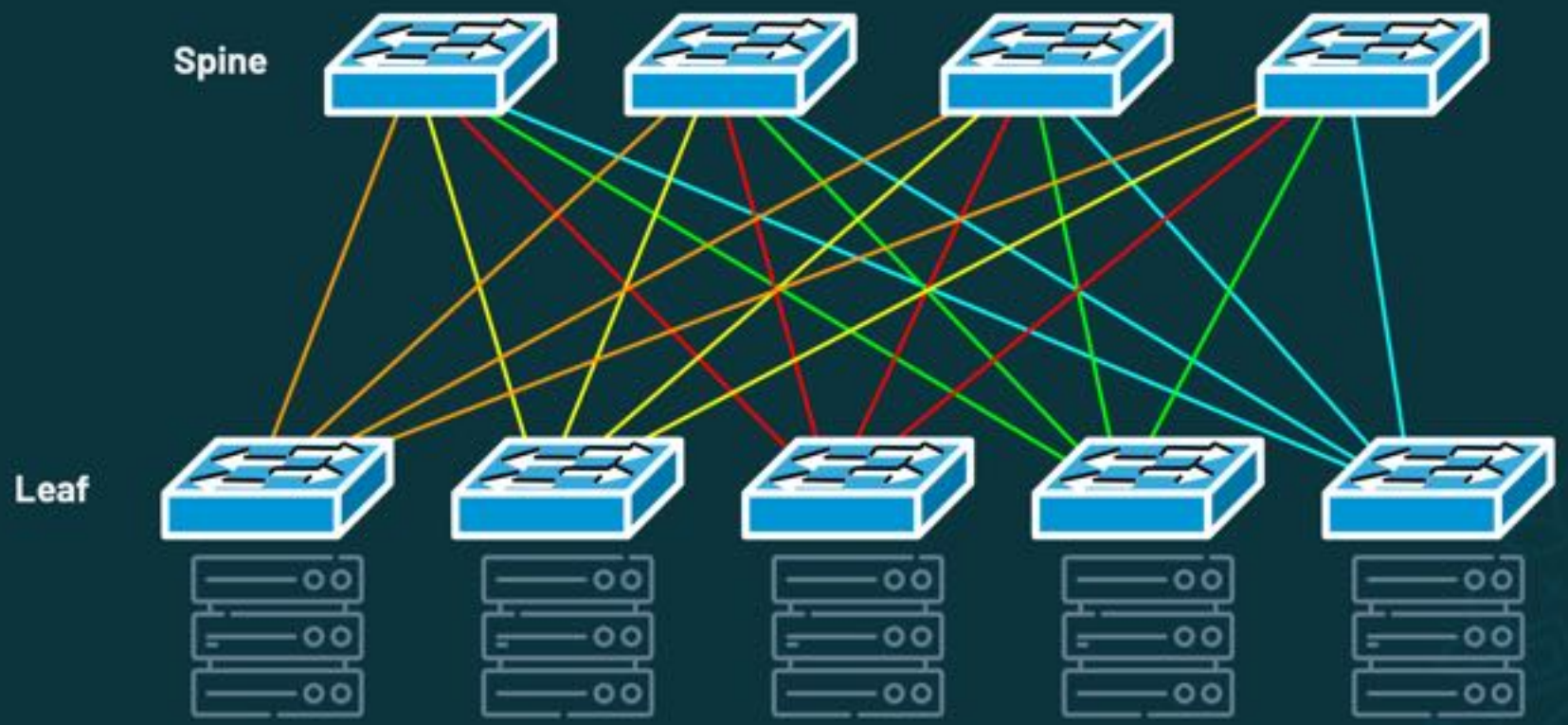
Spanning Tree Protocol (y sus derivados)

STP fue desarrollado en 1985 por Radia Perlman en DEC para permitir rutas redundantes dentro de una topología de Capa 2 y se considera como un hito enorme en la historia de las redes.

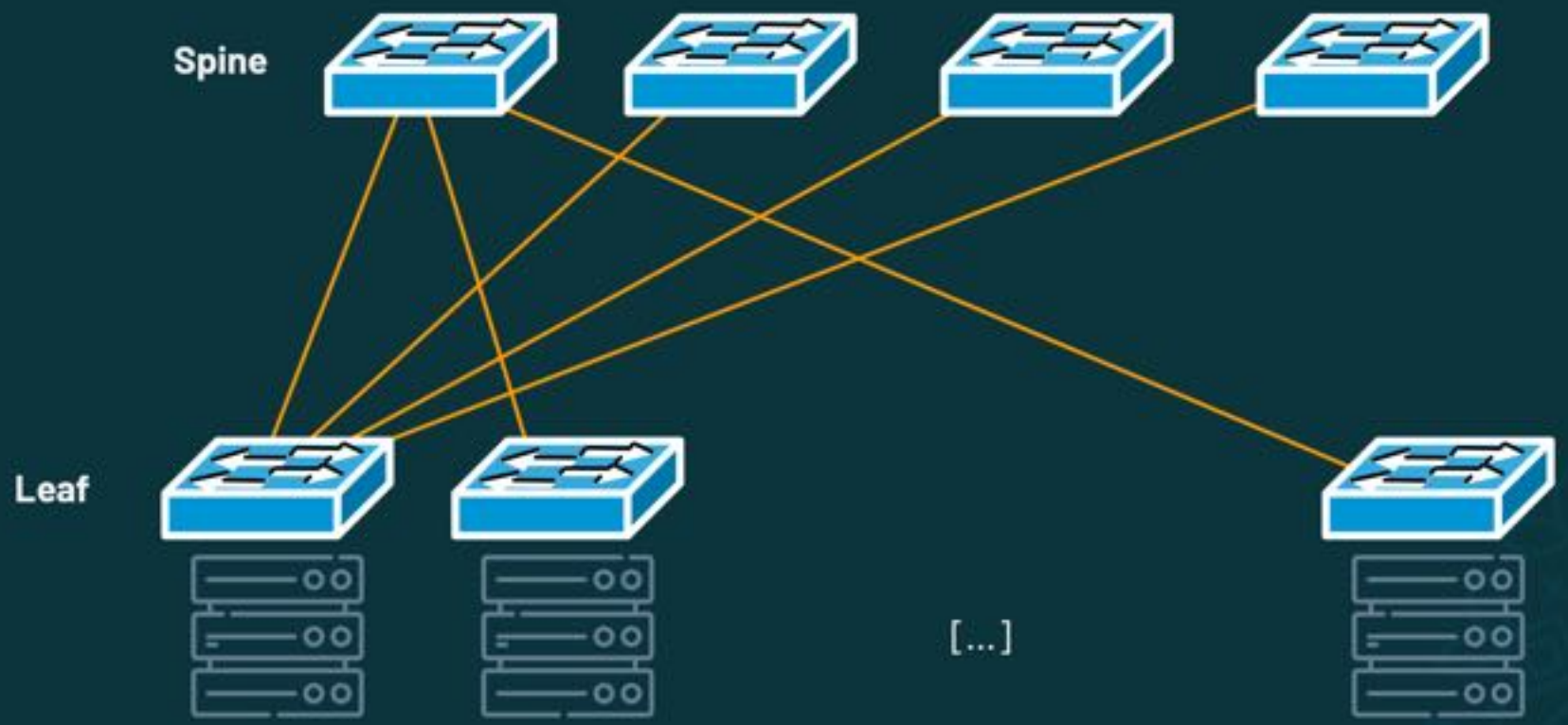
Fué estandarizado por la IEEE 802.1 y desde ese entonces estamos conviviendo con este protocolo y sus derivaciones en casi todos los vendors.

Lamentablemente, STP resuelve un problema pero plantea una limitante importante: Se focaliza en una estructura de árbol desde el punto de vista de su raíz y solamente puede haber un camino activo hacia un dispositivo.

Redes Clos



Redes Clos con STP



Ya es hora de dejar de depender de STP

El cambio de paradigma de forwarding de datos

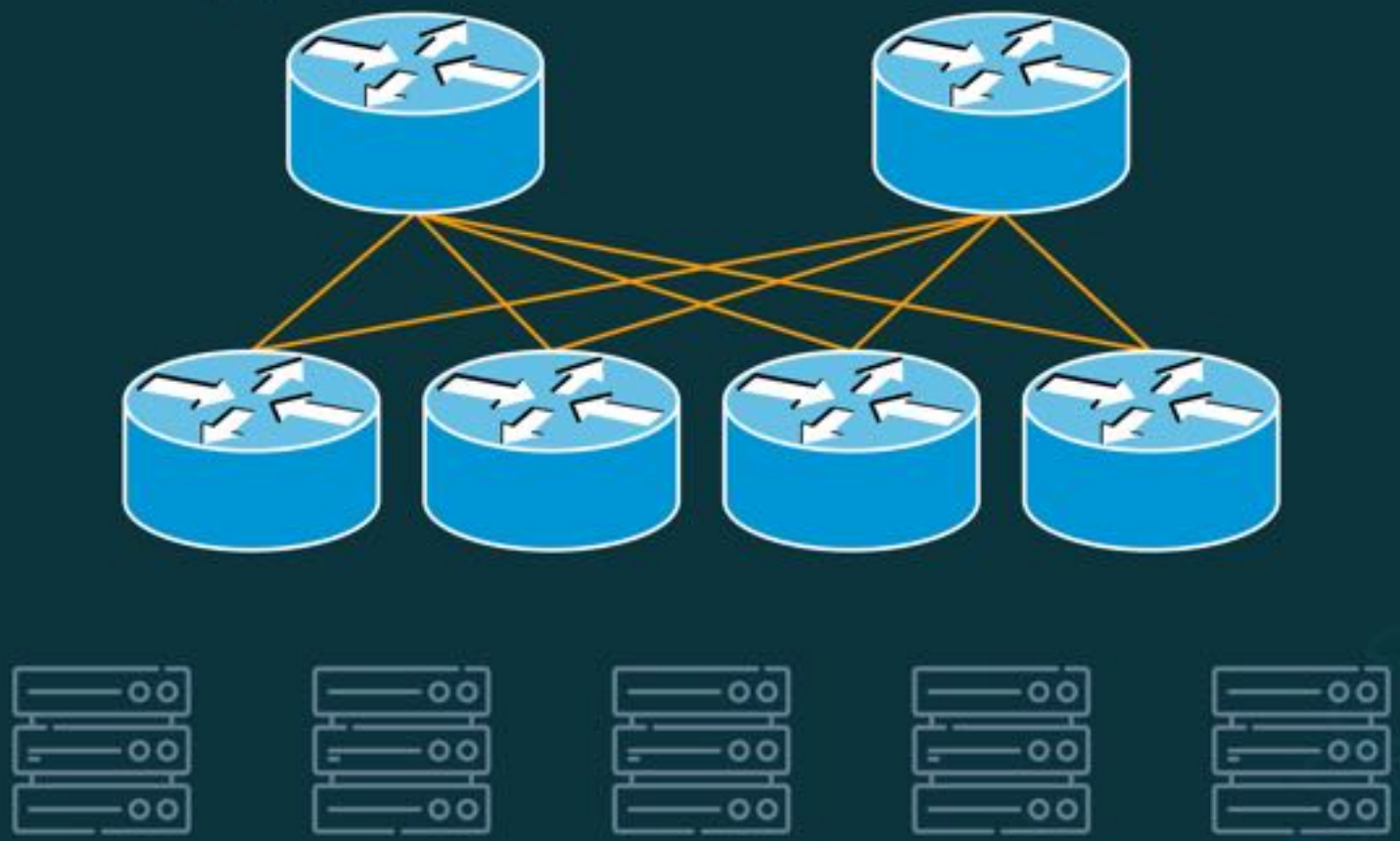
- Este esquema genera una gran matriz de conmutación de tráfico o *"fabric"*.
- Los switches de la red no deben ser exageradamente grandes.
  - Switches más pequeños tienen menos componentes que pueden fallar.
  - Componentes más simples se arreglan de manera simple.
- Los protocolos de prevención de loops bloquean los puertos redundantes.
  - Esta solución no escala.
- La mejor opción es pensar en una red totalmente ruteada y **un overlay L2** por encima.
  - Aprovecha ECMP, se usan todos los enlaces.
  - Se puede tunear para adaptarse rápidamente a los cambios de topología.
- Una parte de la resiliencia y redundancia también pasa a ser responsabilidad de las aplicaciones
  - Al no ser monolíticas, múltiples microservicios se replican aprovechando la conectividad redundante de la red.

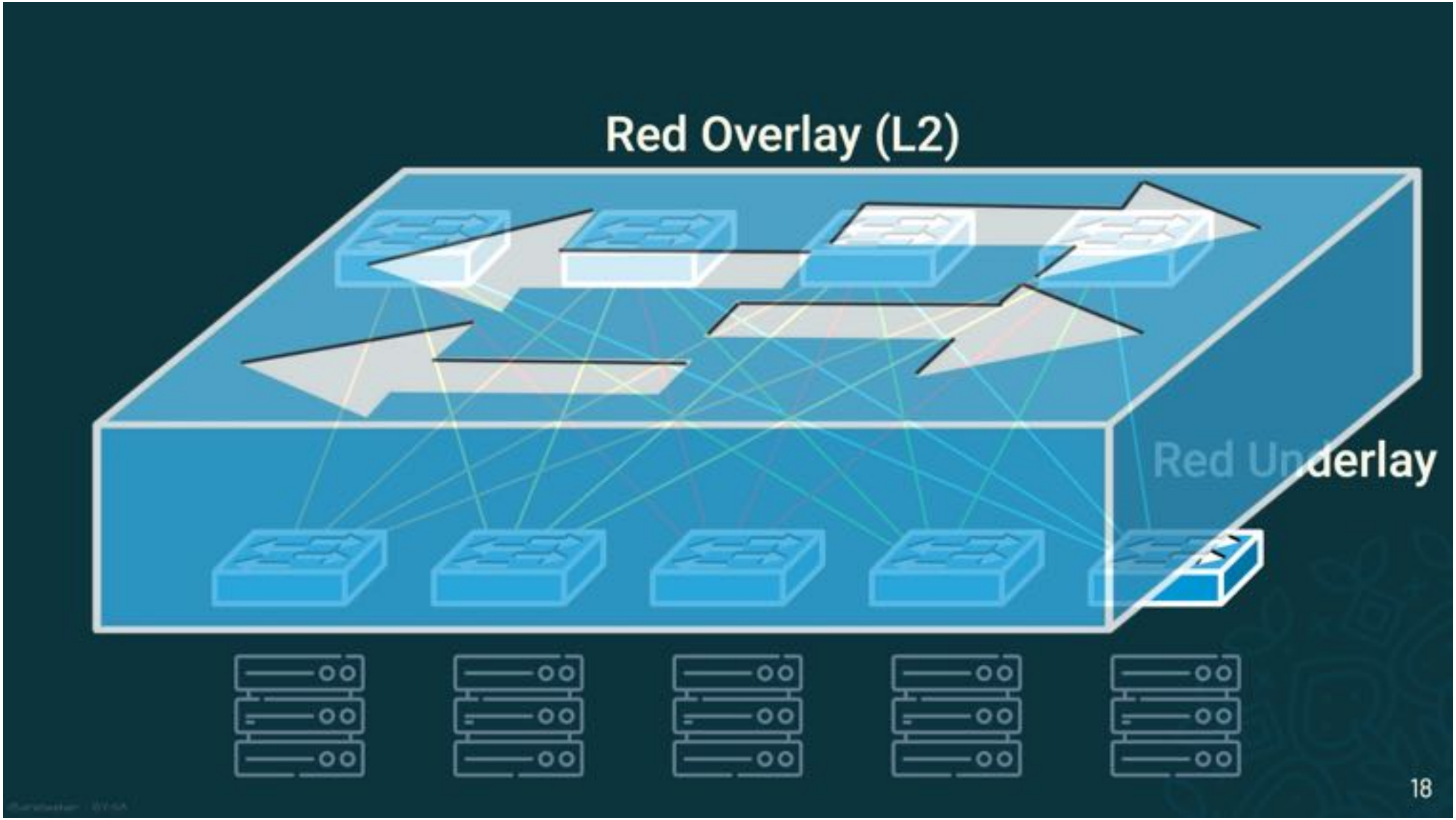
Equal Cost Multi Path (ECMP)

En las redes ruteadas, cuando existen múltiples caminos que pueden llegar a un mismo destino, se debe seleccionar uno o más de ellos para reenviar tráfico. Esta selección se decide exclusivamente en base a las métricas del protocolo de ruteo que se ejecute en la red.

Si hay dos o más rutas con las mismas métricas o características, se puede usar el denominado Equal Cost Multi Path para distribuir el tráfico con el uso de un algoritmo de balanceo de carga para identificar flujos y distribuirlos entre todos los enlaces activos.

Red Underlay (L3)




Red Overlay (L2)
Red Underlay

Underlay

Selección del Protocolo de Ruteo

¿IGP o BGP?

Una vez definido que el fabric opera en Capa 3, debemos elegir el protocolo de ruteo que gobierna los caminos y el balanceo de carga de enlaces. Existen muchas alternativas posibles, cada una con sus propias ventajas y desventajas.

El siguiente análisis incluye la opinión personal del autor en base a su experiencia y casos de uso.

- Ruteo Estático:
    - Inviable.
- Estado Enlace:
    - OSPF:
        - Necesita una instancia distinta para cada protocolo (4|6).
        - Doble gestión = doble trabajo .
        - Propaga cambios en la red a cada rincón de ella, puede generar inestabilidad.
    - IS-IS:
        - Multi protocolo por default.
        - No todos los vendors lo implementan, y los que sí quizás no lo implementan completo.
        - Propaga cambios en la red a cada rincón de ella, puede generar inestabilidad.

¿IGP o BGP?

- BGP:
  - Maduro.
  - Estable.
  - Soporta múltiples AF.
    - IPv4, IPv6, FlowSpec, MPLS, EVPN, etc.
  - ¿Simple?... ¡Simple!.
  - Hace andar internet, permite manejar mucha información.
  - Implementaciones maduras y *completas*, muchas son Open Source.
  - Se puede tunear para un entorno de red interna.
  - Con pequeñas modificaciones, soporta ECMP.
  - Está en contínua evolución.

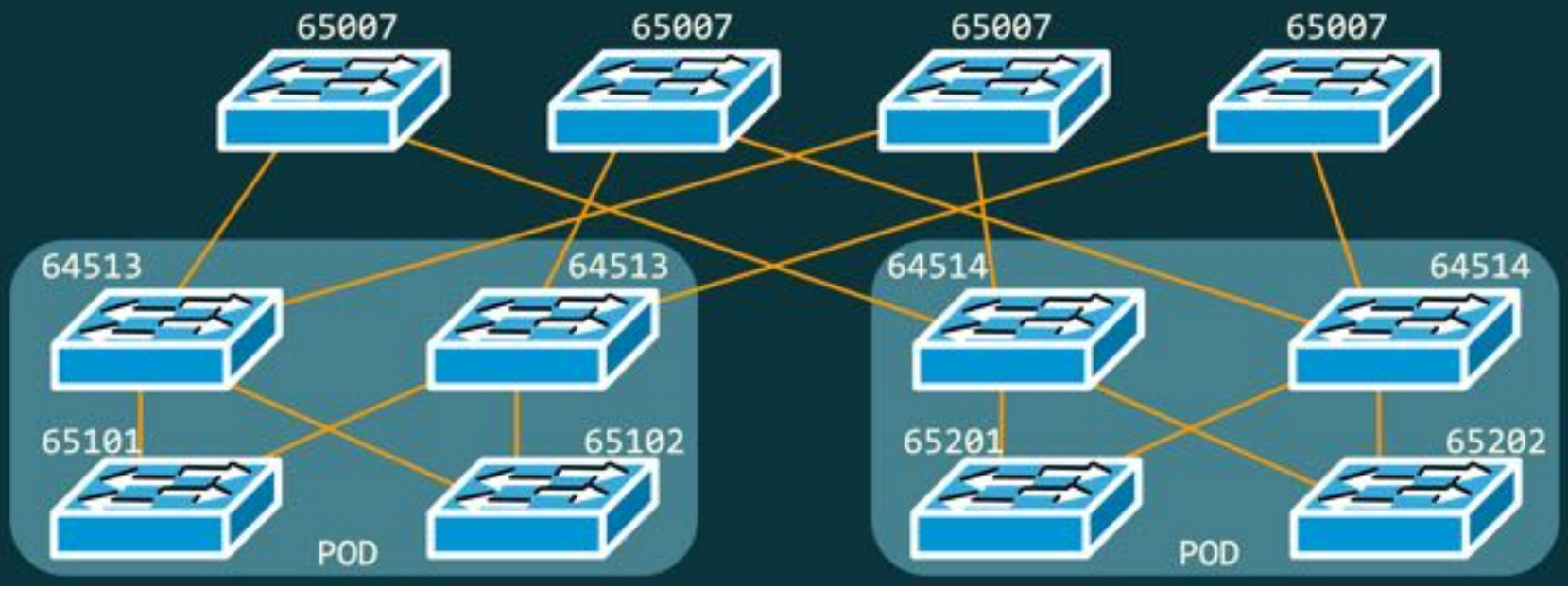
BGP es el protocolo que buscamos

Por favor **NO** pruebe estos ejemplos en producción

1. **RTFMBU** (*Read The Feature Manuals Before Use*)
2. Todos los ejemplos fueron probados en FRRouting v8+
3. Por favor no usar GNU *nano*

¿Usamos eBGP o iBGP?

- **eBGP es más simple que iBGP**
  - Se prefiere implementar eBGP en todos los peerings.
    - ASNs privados (64512-65534), (4200000000-4294967294)
    - Se debe utilizar un esquema de numeración que evite el clásico problema de *"Path-Hunting"*.



¿Qué tunning necesita eBGP?

FRRouting8+

```
router bgp {asn}
 bgp router-id {router-id}
 bgp log-neighbor-changes
 no bgp default ipv4-unicast
 timers bgp 3 9
 neighbor peer-group PG
 neighbor PG remote-as {remote-as}
 neighbor PG advertisement-interval 0
 neighbor PG timers connect 5
 neighbor {ip} peer-group PG
 neighbor {ip} peer-group PG
 address-family ipv4 unicast
  neighbor PG activate
  [...]
  maximum-paths 64
 exit-address-family
```

¿Cada cuánto mandar un KeepAlive? ¿Cuánto tiempo esperar su respuesta?

¿Cuánto esperar para enviar un update?

¿Cuánto esperar para intentar reestablecer el peer?

Habilitar ECMP

¿Qué tunning necesita BGP?

FRRouting8+

- **`bgp bestpath as-path multipath-relax`**
  - Este comando especifica que el proceso de decisión de BGP debe considerar caminos de igual longitud de AS_PATH como candidatos para el cálculo de ECMP.
  - Si no está activo, todo el AS_PATH debe coincidir para el ECMP
- **`bgp bestpath compare-routerid`**
  - Asegura que al comparar rutas que sean iguales en la mayoría de las métricas, incluyendo LOCALPREF, AS_PATH, MED, costo IGP, se desempata basándose en el router-id.
  - Si esta opción está habilitada, se omite la verificación de preferencia de rutas más antiguas.
- **`no bgp ebgp-requires-policy`**
  - RFC8212 estipula que no se intercambian prefijos si no hay un policy definido en el peer.
  - No tiene demasiado sentido tenerlo activo en mi escenario de laboratorio.
  - En el tuyo de producción sí hay que poner filtros, aunque sea en un entorno privado.
    - *"Haz lo que yo digo y no lo que yo permito"*.

Aceptar Peers BGP definidos en un bloque IP

FRRouting8+

- Permite establecer peerings sin necesidad de identificar una IP específica.
- Necesita asociarse a un *peer-group* que le pase el resto de los parámetros.
- No se recomienda usar cuando hay autenticación MD5 para los peers.
  - Salvo que la versión de kernel sea >= 4.14

```
bgp listen range <X.X.X.X/M|X:X::X:X/M> peer-group PG
```

Unnumbered BGP Peers

FRRouting8+

- Permite establecer peerings mediante el uso de la dirección IPv6 de Link Local de una interface.
- También opcionalmente permite establecer esos peers sin necesidad de conocer el AS remoto.

```
neighbor PEER interface remote-as <internal|external|ASN>
```

RFC5549 Advertising IPv4 Network Layer Reachability Information with an IPv6 Next Hop

- Comprende una extensión de las definiciones AFI / SAFI para permitir que la dirección del *NextHop* para NLRI IPv4 o VPN-IPv4 pertenezca al protocolo IPv4 o al IPv6.
- Define la codificación de la información del NextHop para determinar cuál de los protocolos a los que pertenece realmente la dirección y una nueva capacidad BGP que permita a los peers MP-BGP descubrir dinámicamente si pueden intercambiar IPv4 NLRI y VPN-IPv4 NLRI con un NextHop IPv6.
- En criollo... poder enrutar tráfico IPv4 utilizando un NextHop IPv6.
  - A través de una dirección de Link Local.
  - Solo dependiendo del intercambio de RA.
  - NO es NAT.
  - NO es un Túnel.
  - NO es 464464646464646446464XLAT ni ningún otro método de transición.
  - SÍ ES ruteo puro.

Tutorial de Alejandro Acosta (LACNIC) —> https://www.youtube.com/watch?v=ndoKomM7ONo

RFC5549 Advertising IPv4 Network Layer Reachability Information with an IPv6 Next Hop

```
asw@SW1:~$ ip neighbor show dev eth0
169.254.0.1 lladdr 00:50:00:00:03:00 PERMANENT
fe80::250:ff:fe00:300 lladdr 00:50:00:00:03:00 router STALE
[...]
SW1# sh ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup

[...]
B>* 10.254.0.1/32 [20/0] via fe80::250:ff:fe00:300, eth0, weight 1, 00:00:44
B>* 10.255.1.0/30 [20/0] via fe80::250:ff:fe00:300, eth0, weight 1, 00:00:44
[...]
```

```
SW3# sh run
Building configuration...
[...]
interface eth0
 ipv6 nd ra-interval 10
 no ipv6 nd suppress-ra
!
[...]
frr version 7.5.1
[...]
router bgp 64513
[...]
 neighbor PG peer-group
 neighbor PG remote-as external
 neighbor PG capability extended-nexthop
 neighbor eth0 interface peer-group PG
 neighbor eth1 interface peer-group PG
 !
 address-family ipv4 unicast
  redistribute connected
 exit-address-family
 !
 address-family ipv6 unicast
  neighbor PG activate
 exit-address-family
```

Tutorial de Alejandro Acosta (LACNIC) —> https://www.youtube.com/watch?v=ndoKomM7ONo

Overlay

Data Plane: VXLAN

Resumen ejecutivo de VXLAN

- **Para el público en general:**
  - Permite extender una red ethernet a través de fronteras demarcadas por routers.
- **Para los que saben de Mikrotik:**
  - Es como los túneles EoIP pero en vez de GRE usa UDP y es un estándar IETF.
- **Para los que no saben lo que es EoIP:**
  - Es encapsular tramas Ethernet dentro de datagramas UDP enrutados eficientemente gracias a los algoritmos de balanceo de carga L3 (ECMP).
- **Para los que saben de MPLS / VPLS:**
  - Funcionalidades similares, pero mejor adaptado a entornos de DC.
  - AToM soluciona algunas cosas que no se pueden hacer con VXLAN.
- **Para los que no saben que es UDP:**
  - Es el protocolo que usan para enviarte ataques de denegación de servicio cuando los administradores no configuran bien los DNS/NTP/SNMP/etc.
  - ~~También es el encargado de que te puedas descargar tus series desde redes P2P~~.

¿Necesitamos VXLAN?

VXLAN no es la única forma de transportar tramas Ethernet a través de una red de transporte. Ya hemos pasado por Frame Relay, ATM, MPLS, etc...

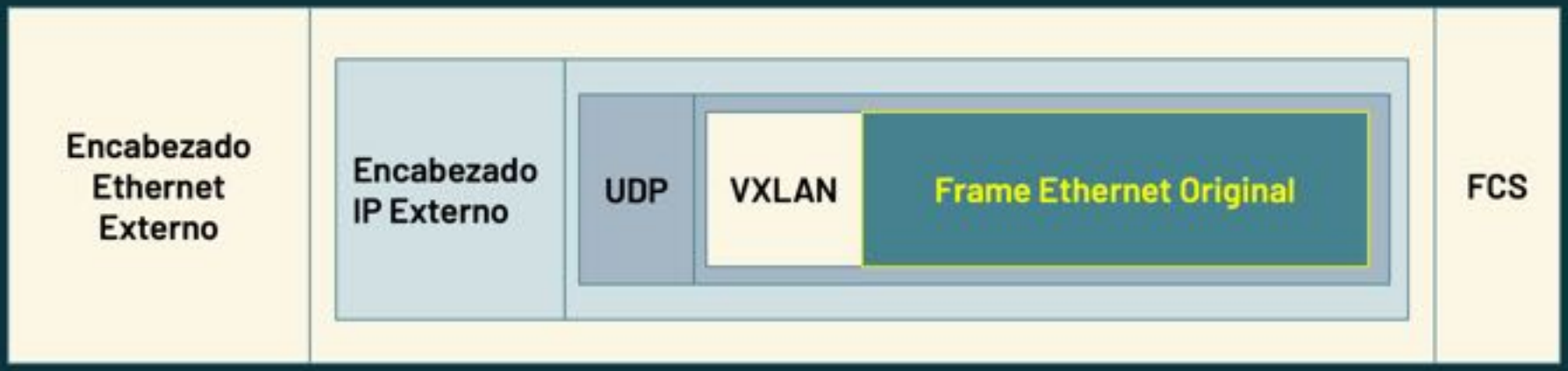
Pero si ignoramos las tecnologías antiguas y los caos para configurar VPLS o ingeniería de tráfico (MPLS-TE), podemos hacer la siguiente pregunta:

¿Por qué se preferiría VXLAN en vez de EoMPLS / VPLS?

En un mundo MPLS, el dispositivo PE debe estar estrechamente relacionado con el core de la red, ya que necesita un LSP de extremo a extremo para proporcionar servicios de transporte Ethernet.

Todo lo que necesitamos en un mundo VXLAN es conectividad IP.

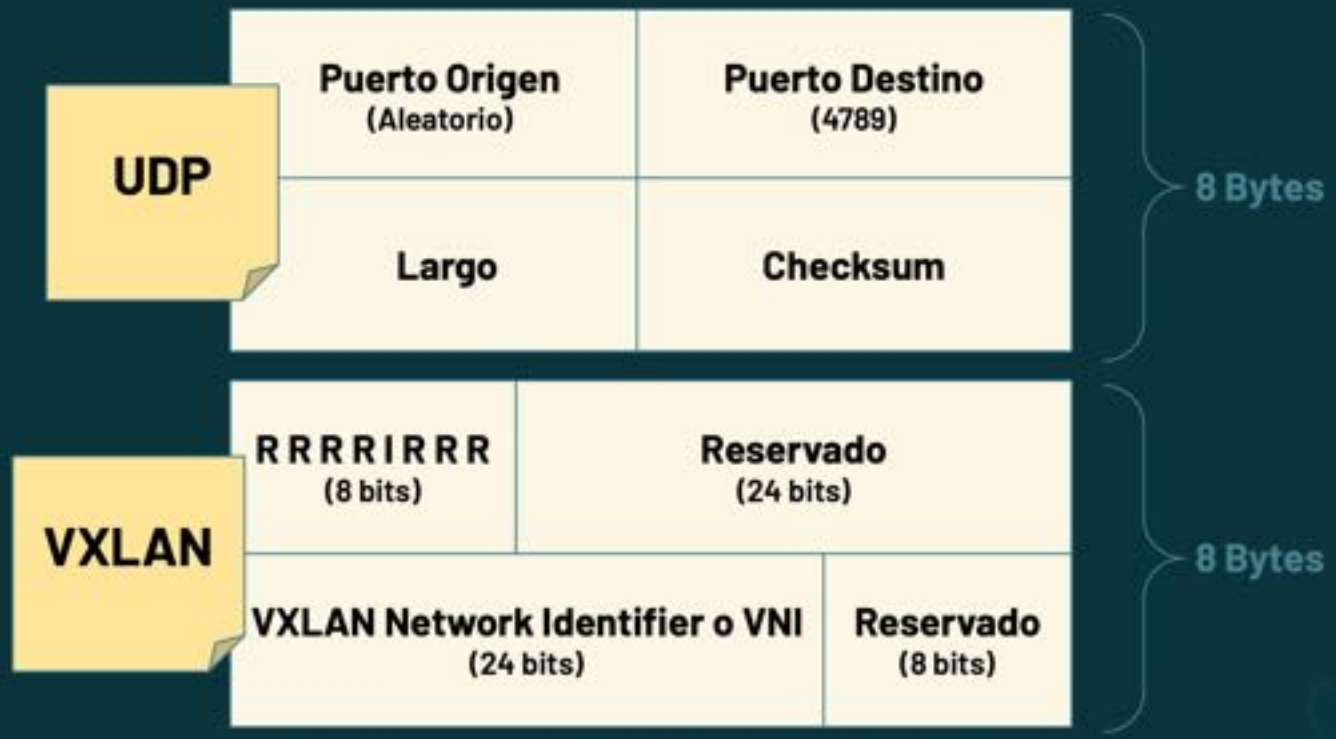
Encapsulamiento VXLAN



El overhead necesario es de **al menos 50 bytes**, por lo que se deben soportar jumbo frames a lo largo de la red.

- 14 bytes de Encabezado Ethernet Externo (18 bytes si hay 802.1q).
- 20 bytes de Encabezado IPv4 Externo (40+ si es sobre IPv6).
- 8 bytes de encabezado UDP.
- 8 bytes de encabezado VXLAN.
- + bytes si contamos doble tag 802.1q, etiquetas MPLS, etc...

Encabezado UDP & VXLAN



VLAN ID => 12 bits => 4.096 IDs diferentes (4.094 usables).
VXLAN VNI => 24 bits => 16.777.216 IDs diferentes.

VNI

" VXLAN Network Identifier (o VXLAN Segment ID)"

RFC7348

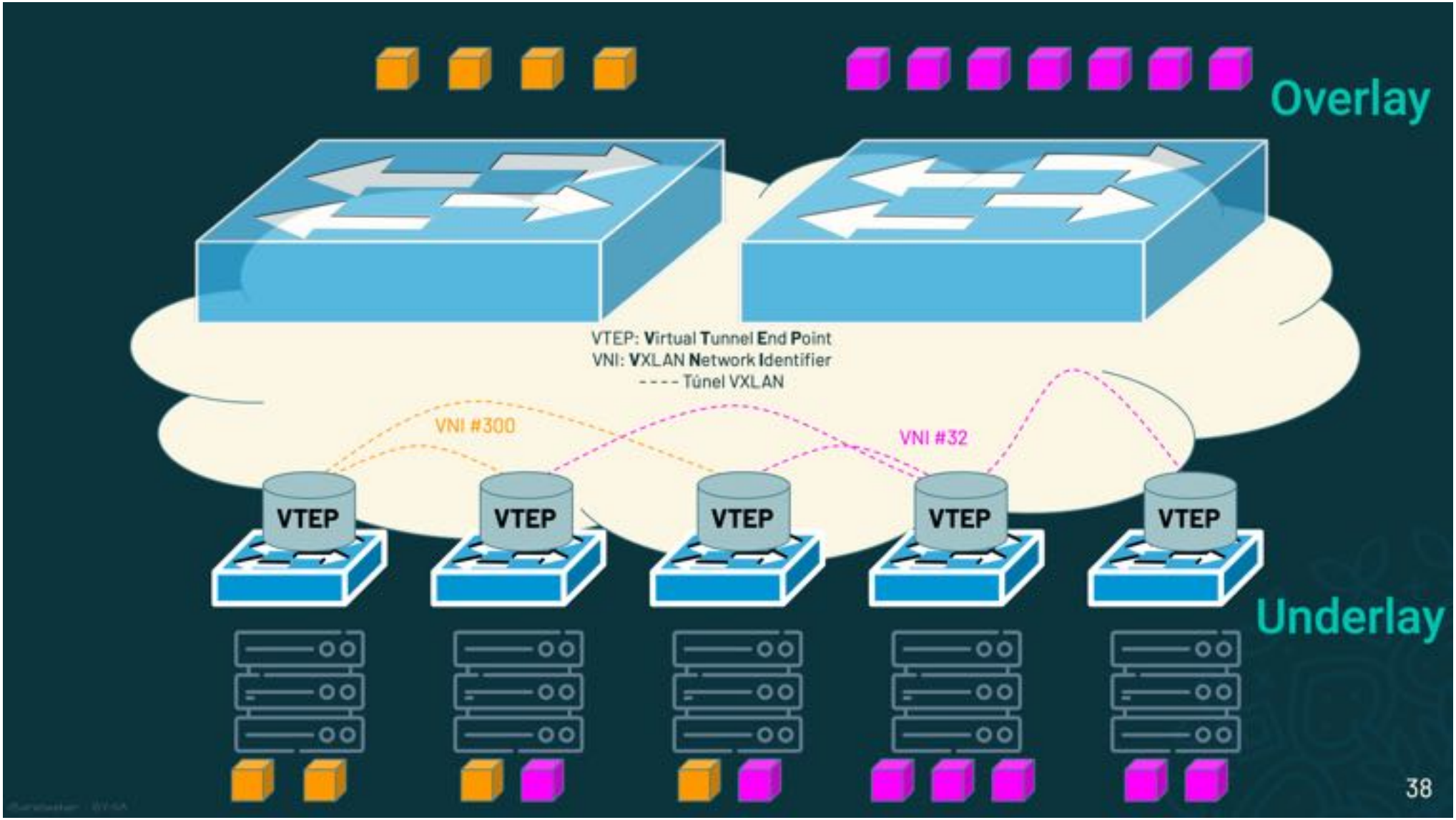
VXLAN Network Identifier

- VNI es un campo de 24 bits incluído en el encabezado de VXLAN.
- Se utiliza para identificar de forma única una VXLAN.
- El VNI es similar a una ID de VLAN (*802.1q*), pero al tener 24 bits le permite crear muchas más VXLANs (16,7 millones) que los 4094 posibles ID de VLAN.
- Básicamente podemos pensar en un mapeo desde un VLAN-ID en el underlay a un VNI en el overlay.

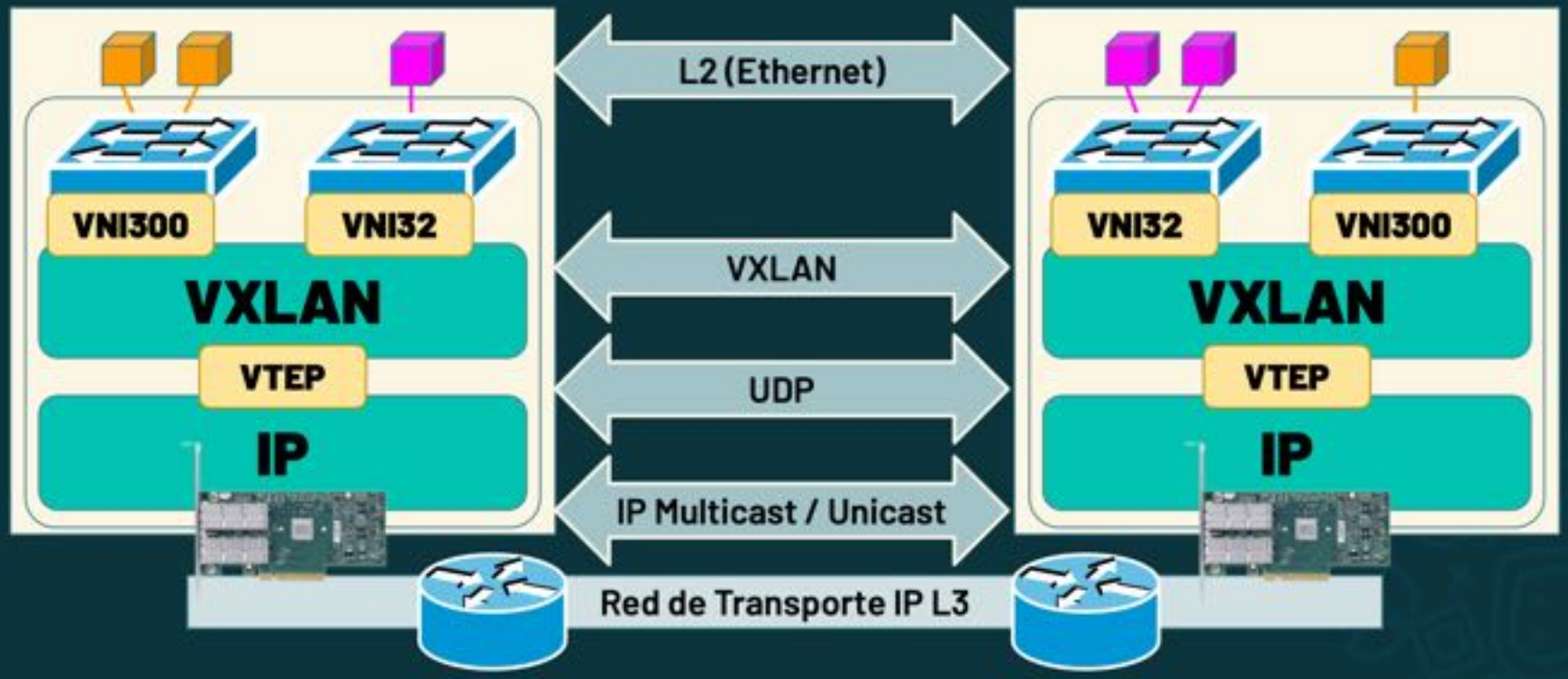
VTEP

" Entidad que origina y/o termina túneles VXLAN"

RFC7348


Overlay
VTEP: Virtual Tunnel End Point
VNI: VXLAN Network Identifier
- - - - Túnel VXLAN
VNI #300
VNI #32
VTEP
VTEP
VTEP
VTEP
VTEP
Underlay

Los equipos finales también pueden hablar VXLAN



Consideraciones Importantes

Consideraciones

- Generalmente cada Switch LEAF genera un único VTEP desde alguna interface Loopback, por lo que hablar de LEAF, EDGE o VTEP generalmente es hablar de lo mismo.
- Algunas de las ventajas de usar protocolos de estado enlace para establecer el underlay y EVPN por sobre el overlay tiene que ver con poder *apagar* algunas partes de la red para poder hacer mantenimientos programados.
- Siempre se debe comprender los mecanismos de balanceo de carga de los equipos de red para poder aprovechar todos los caminos de la manera más eficiente en los entornos con ECMP.
- Siempre que hablemos de Multicast en el underlay, sigue siendo necesaria la implementación de los protocolos que lo hacen funcionar (IGMP, MLD, PIM [SSM; BIDIR], etc).
- En esta presentación no consideramos temas avanzados tales como ethernet multihoming, designated forwarding y rutas EVPN Tipo 1; [4-255] ni EVPN sobre redes MPLS.

Consideraciones sobre cómo manejar algunos tipos de tráfico de las redes Ethernet

Broadcast

Tráfico destinado a todos los hosts (solo IPv4)

Unknown Unicast

Tráfico destinado a un host que no existe en la FDB

Multicast

Tráfico destinado a un grupo de los hosts.

1) Data Plane

Source Address Learning, idéntico a los bridges actuales y asignaciones estáticas.

2) Multicast

Source Address Learning, unir todos los VTEP de un mismo VNI a una dirección de grupo de Multicast específica.

3) Control Plane Distribuído

BGP EVPN

Configuración de VXLAN con flooding Unicast estático a todos los VTEP

GNU/Linux
Kernel v5+

```
root@srv1:~ # ip link add vxlan100 type vxlan \
  id 100 \
  dstport 4789 \
  local 172.22.2.2
#
ip link set up dev vxlan100
#
bridge fdb append 00:00:00:00:00:00 dev vxlan100 dst 172.24.24.24 # vtep2
bridge fdb append 00:00:00:00:00:00 dev vxlan100 dst 172.25.25.25 # vtep3
[...]
bridge fdb append 00:00:00:00:00:00 dev vxlan100 dst 172.31.31.31 # vtepN
```

Configuración de VXLAN con direcciones MAC estáticas

GNU/Linux
Kernel v5+

```
root@srv1:~ # ip link add vxlan100 type vxlan \
  id 100 \
  dstport 4789 \
  local 172.22.2.2 \
  nolearning
#
ip link set up dev vxlan100
#
bridge fdb append 00:00:00:00:00:00 dev vxlan100 dst 172.24.24.24 # flood
bridge fdb append 00:00:00:00:00:00 dev vxlan100 dst 172.25.25.25 # flood
bridge fdb append f0:18:98:41:a9:1e dev vxlan100 dst 172.24.24.24
bridge fdb append aa:20:a1:8a:1b:bd dev vxlan100 dst 172.25.25.25
```

Configuración de VXLAN con Multicast

GNU/Linux
Kernel v5+

```
root@srv1:~ # ip link add vxlan100 type vxlan \
  id 100 \
  dstport 4789 \
  local 172.22.2.2 \
  group 239.0.0.100 \
  dev eth0 \
  ttl 5
#
brctl addbr br100
brctl addif br100 vxlan100
brctl addif br100 veth10 # vm1
brctl addif br100 veth11 # vm2
brctl stp br100 off
#
ip link set up dev br100
ip link set up dev vxlan100
```

```
root@srv2:~ # ip link add vxlan100 type vxlan \
  id 100 \
  dstport 4789 \
  local 172.25.25.25 \
  group 239.0.0.100 \
  dev ens18 \
  ttl 5
#
brctl addbr br100
brctl addif br100 vxlan100
brctl addif br100 veth35 # vm3
brctl addif br100 veth45 # vm4
brctl stp br100 off
#
ip link set up dev br100
ip link set up dev vxlan100
```

Overlay

Control Plane con BGP EVPN

¿Cómo aprendo qué direcciones IP tienen los VTEP de la red?

¿Cómo aprendo hacia que VTEP debo hacer un túnel para llegar a una determinada MAC?

(VTEP a VNI)

BGP EVPN

- La utilización de *VXLAN* claramente ofrece muchas ventajas.
  - Aunque se evita el *STP*, aún se requieren herramientas para emular el flooding.
- Para crear un control plane distribuído que escale a los escenarios actuales, se implementó un nuevo *SAFI* para el *AFI L2VPN* de *BGP*, llamado *Ethernet VPN* o simplemente *EVPN*.
  - Similar a lo que se hizo para soportar *MPLS L3VPN* y *MPLS L2VPN / VPLS*.
  - *EVPN* comenzó su desarrollo para trabajar como un nuevo control plane de MPLS, pero luego se portó a otros escenarios, como por ejemplo VXLAN.
  - Se reutilizan los conceptos de *Route Distinguishers* y *Route Targets*.
    - Se pueden configurar automáticamente.
- Algunos datos que los equipos que hablan EVPN pueden informar a sus peers:
  - Direcciones *MAC* aprendidas en cada *VNI* de cada *VTEP* [ruta *EVPN* tipo 2].
  - Direcciones de los *VTEP* y *VNI* para manejar el tráfico *BUM*. [ruta *EVPN* tipo 3].
  - Prefijos y rutas *IP* [ruta *EVPN* tipo 5].
  - Información sobre *MAC Mobility*.

EVPN Route Type 2

"MAC/IP Advertisement Route"

Son responsables por la publicación de información de alcanzabilidad de direcciones MAC e IP de equipos finales.

Además pueden estar presentes atributos de communities adicionales, como el tipo de encapsulación (Encap 8: VXLAN), Route Target, o MAC Mobility.

RD (8 bytes)
ESI (10 bytes)
Ethernet Tag ID (4 bytes)
(*) MAC Address Length (1 byte)
(*) MAC Address (6 bytes)
IP Address Length (1 byte)
IP Address (0, 4 ó 16 bytes)
MPLS Label 1 (3 bytes) / L2VNI
MPLS Label 2 (0 ó 3 bytes) / L3VNI

(*) = Campos requeridos

EVPN Route Type 3

"Inclusive Multicast Ethernet Tag Route"

Proporciona una manera de replicar el tráfico BUM en forma de unicast.

Este tipo de ruta se genera inmediatamente y se envía a todos los VTEP que participan en la H-E-R tan pronto como se configura y se vuelve operativo un VNI en un VTEP.

De esta forma, cada VTEP conoce todos los demás VTEP remotos a los que se les debe enviar una copia de un paquete BUM en un VNI determinado.

El Data Plane debe soportar la generación de N copias en modo unicast para cada paquete BUM.

RD (8 bytes)
ESI (10 bytes)
Ethernet Tag ID (4 bytes)
IP Address Length (1 byte)
Originating Router IP Address (4 ó 16 bytes)

Ejemplo de configuración de BGP EVPN en LEAF

FRRouting8+

```
router bgp {ASN}
 bgp router-id {RID}
 [...]
 neighbor FABRIC peer-group
 neighbor FABRIC remote-as external
 neighbor FABRIC capability extended-nexthop
 neighbor {INTERFACE} interface peer-group FABRIC
 [...]
 !
 address-family {AF} unicast
  maximum-paths 64
  neighbor FABRIC activate
  redistribute connected
 exit-address-family
!
 address-family l2vpn evpn
  neighbor FABRIC activate
  advertise-all-vni
 exit-address-family
!
end
```

Ejemplo de configuración de BGP EVPN en SPINE

FRRouting8+

```
router bgp {ASN}
 bgp router-id {RID}
 [...]
 neighbor LEAFs peer-group
 neighbor LEAFs remote-as external
 neighbor LEAFs capability extended-nexthop
 neighbor {INTERFACE} interface peer-group LEAFs
 [...]
 !
 address-family {AF} unicast
  maximum-paths 64
  neighbor LEAFs activate
  redistribute connected
 exit-address-family
!
 address-family l2vpn evpn
  neighbor LEAFs activate
 exit-address-family
!
end
```

Direcciones MAC aprendidas en un VNI

FRRouting8+

```
leaf10# show evpn mac vni 101

Number of MACs (local and remote) known for this VNI: 3
Flags: N=sync-neighs, I=local-inactive, P=peer-active, X=peer-proxy

MAC               Type   Flags Intf/Remote ES/VTEP            VLAN  Seq #'s
aa:c1:ab:e6:99:af remote       100.100.100.100                     0/0
a0:c1:ab:e0:25:58 remote       100.100.100.32                      0/0
[...]
aa:c1:ab:93:63:02 local        eth3                                0/0
```

Rutas tipo 2 recibidas por BGP EVPN

FRRouting8+

```
leaf10# show bgp l2vpn evpn route
BGP table version is 1, local router ID is 100.100.100.10
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal
Origin codes: i - IGP, e - EGP, ? - incomplete
EVPN type-1 prefix: [1]:[EthTag]:[ESI]:[IPlen]:[VTEP-IP]:[Frag-id]
EVPN type-2 prefix: [2]:[EthTag]:[MAClen]:[MAC]:[IPlen]:[IP]
EVPN type-3 prefix: [3]:[EthTag]:[IPlen]:[OrigIP]
EVPN type-4 prefix: [4]:[ESI]:[IPlen]:[OrigIP]
EVPN type-5 prefix: [5]:[EthTag]:[IPlen]:[IP]

Network          Next Hop            Metric LocPrf Weight Path
                     Extended Community
Route Distinguisher: 100.100.100.32:2
*> [2]:[0]:[48]:[aa:c1:ab:e6:99:af]
                     100.100.100.32                         0 65007 32 i
                     RT:[Y]:101 ET:8
```

Rutas tipo 3 recibidas por BGP EVPN

FRRouting8+

```
leaf10# show bgp l2vpn evpn route
BGP table version is 1, local router ID is 100.100.100.10
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal
Origin codes: i - IGP, e - EGP, ? - incomplete
EVPN type-1 prefix: [1]:[EthTag]:[ESI]:[IPlen]:[VTEP-IP]:[Frag-id]
EVPN type-2 prefix: [2]:[EthTag]:[MAClen]:[MAC]:[IPlen]:[IP]
EVPN type-3 prefix: [3]:[EthTag]:[IPlen]:[OrigIP]
EVPN type-4 prefix: [4]:[ESI]:[IPlen]:[OrigIP]
EVPN type-5 prefix: [5]:[EthTag]:[IPlen]:[IP]

Network          Next Hop            Metric LocPrf Weight Path
                     Extended Community
Route Distinguisher: 100.100.100.32:2
*> [3]:[0]:[32]:[100.100.100.32]
                     100.100.100.32                          0 65007 32 i
                     RT:32:101 ET:8
```

Resumen de EVPN para Bridging

- Aprendizaje de Direcciones MAC:
  - MACs locales se aprenden como en los bridges tradicionales.
  - MACs remotas solamente se aprenden desde el control plane.
- Manejo de tráfico BUM:
  - Flooding optimizado gracias a las rutas tipo 3.
  - Mejoras gracias al Head End Replication.
  - Usar multicast L3 estático es posible según el escenario.
- Supresión de ARP:
  - Facilita la resolución de ARP lo más cerca posible del host que pregunta.
- Hosts con doble AC:
  - Se puede elegir entre mLAG o EVPN puro.
- Configuración simple:
  - FRR utiliza configuraciones por default que están preparadas para despliegues rápidos y seguros.

¡Gracias!

¿Preguntas?

ariel@weher.net
@arielweher
@lacnog
https://lacnog.org