

Teoría de las Comunicaciones

Claudio Enrique Righetti – Rodrigo Castro

Segundo Cuatrimestre de 2017

**Departamento de Computación
Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires
Argentina**



Nivel de Red



Ruteo

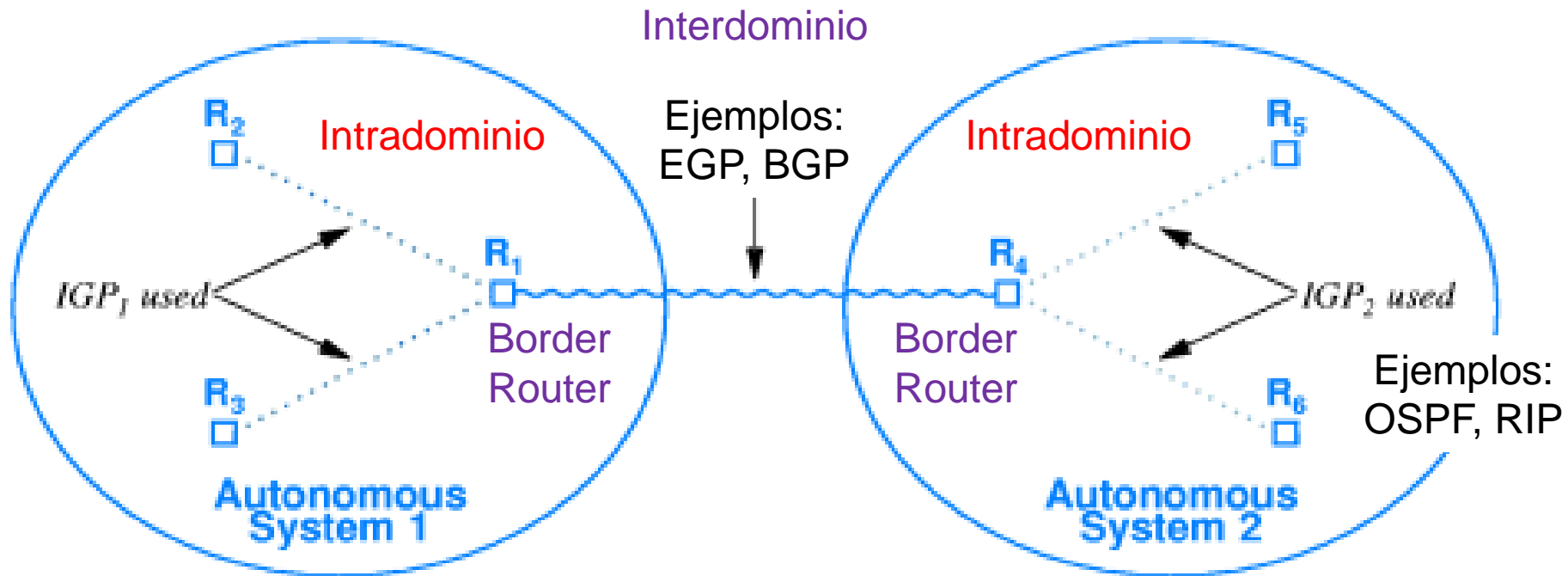
Agenda

- ▶ Introducción: ruteo interno y externo
- ▶ Algoritmos y protocolos
- ▶ Escalabilidad

Introducción

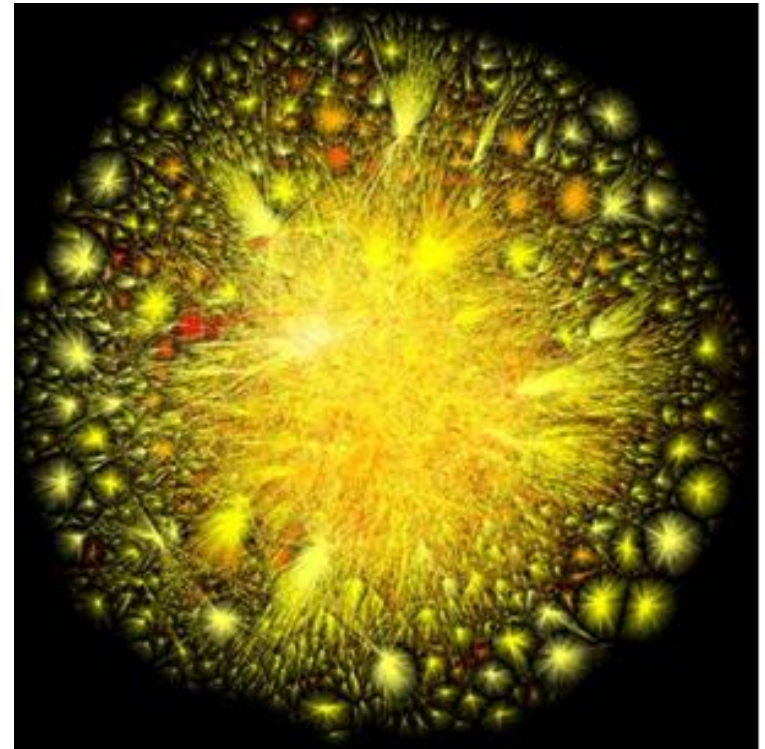
- ▶ Los protocolos de ruteo pueden clasificarse en :
 - ▶ Ruteo Interno (**IGP**, *Internal Gateway Protocols*)
 - ▶ Ruteo Externo (**EGP**, *External Gateway Protocols*)
- ▶ De Ruteo Interno: su dominio de ruteo es **dentro de un Sistema Autónomo (AS)**. Son protocolos **intradominio**.
- ▶ De Ruteo Externo: aplican a **interdominios**, es decir rutean **entre distintos AS**
- ▶ Podemos decir que **Internet es una interconexión de muchos AS**

Sistemas Autónomos



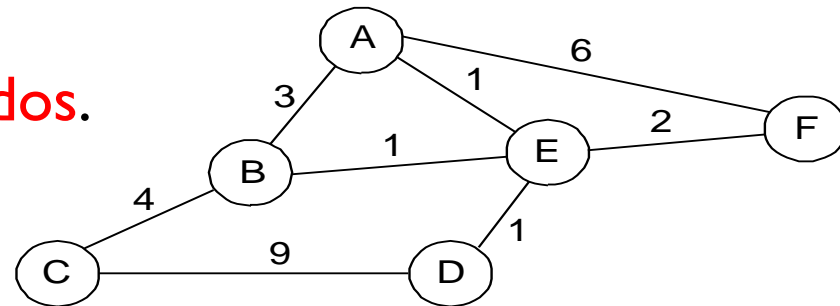
Introducción

- ▶ Un grafo o “red”
- ▶ En Internet existen varios “planos”: **la red de routers**, el grafo de links web (www), el grafo de Name Servers, etc.
- ▶ También existen otros grafos como P2P, CDNs, etc.

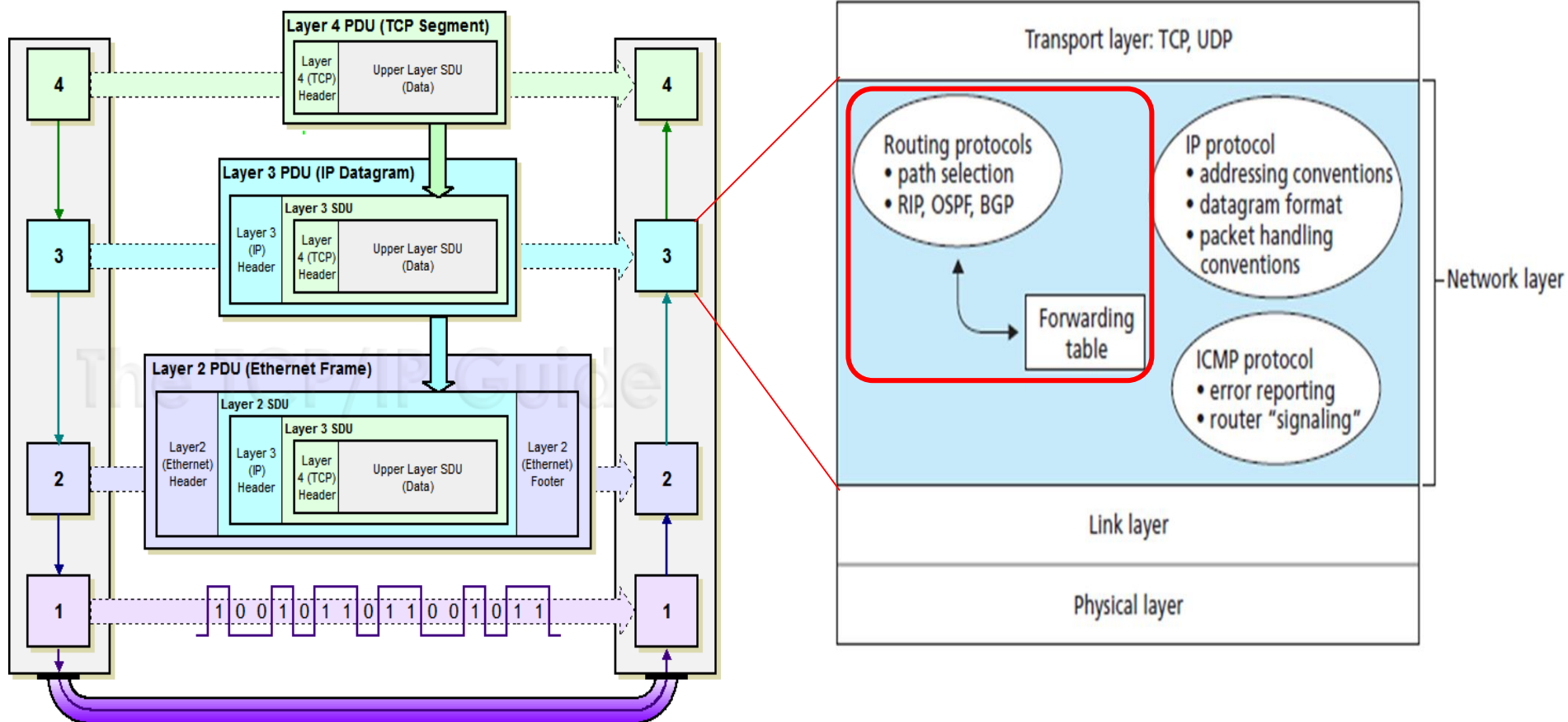


Introducción

- ▶ Re-envío (Forwarding) vs. Ruteo (Routing)
 - ▶ **Re-envío**: proceso para seleccionar una **puerta de salida** basado en la **dirección destino** y las **tablas de ruteo**.
 - ▶ **Ruteo**: proceso mediante el cual son construidas las **tablas de ruteo**.
- ▶ El Ruteo es un problema de **grafos**, **optimización** y **algoritmos distribuidos**.
 - ▶ La red vista como un conjunto de nodos y arcos pesados.
- ▶ **Problema**: Encontrar el **camino de menor costo** entre dos nodos, en **tiempos razonables**, usando **recursos mínimos**.
- ▶ Tipos de ruteo:
 - ▶ **Estático**: Configuración manual, no reactiva.
 - ▶ **Dinámico**: Configuración autónoma y adaptativa (acorde a fallas, carga de nodos, estado de enlaces, etc.)



Ruteo en el modelo OSI



Routing

(a)	
Prefix/Length	Next Hop
18/8	171.69.245.10

(b)		
Prefix/Length	Interface	MAC Address
18/8	if0	8:0:2b:e4:b:1:2

- (a) Tabla de **routing**
- (b) Tabla de **forwarding**

Protocolos de Ruteo Interno

RIP

OSPF

Vector de distancia vs. Estado del enlace

Cada Router	DISTANCE-VECTOR	LINK-STATE
Qué informa ?	• Toda su Tabla de Ruteo	• Solo el Estado de sus Enlaces
A quién le pasa información ?	• Solo a sus vecinos	• A toda la red (inundación)
Algoritmo utilizado	• Bellman-Ford Distribuido	• Dijkstra
Datos utilizados	• Información de los vecinos	• Estado de Enlaces de cada nodo
Estructuras de Datos	<ul style="list-style-type: none"> • Tabla de Distancias • Tabla de Ruteo 	<ul style="list-style-type: none"> • Tabla de Estado de Enlaces • Tabla de Ruteo
Características	<ul style="list-style-type: none"> • Ciclos de Ruteo • Gran variedad de Algoritmos: <ul style="list-style-type: none"> • Merlin-Segall • Jaffe-Moss • Esquema OP • Diffusing Comp • Cheng • Cálculo Distribuido 	<ul style="list-style-type: none"> • Visión Consistente de la Red • Gran uso de CPU y Memoria • Algoritmo Básico único • Cálculo Centralizado
Ejemplo de Protocolos de Internet	RIP	OSPF

Protocolos de Ruteo Interno más utilizados

- ▶ **RIP**: Routing Information Protocol
 - ▶ usa algoritmo de **vector de distancia**
 - ▶ basado en cuenta de hops
 - ▶ desarrollado por XNS
 - ▶ distribuido con Unix
- ▶ **OSPF**: *Open* Shortest Path First
 - ▶ usa algoritmo de **estado de enlaces**
 - ▶ soporta balanceo de carga y QoS (Quality of Service)
 - ▶ soporta autenticación
 - ▶ muchos textos dicen “reciente estándar en Internet” (!)



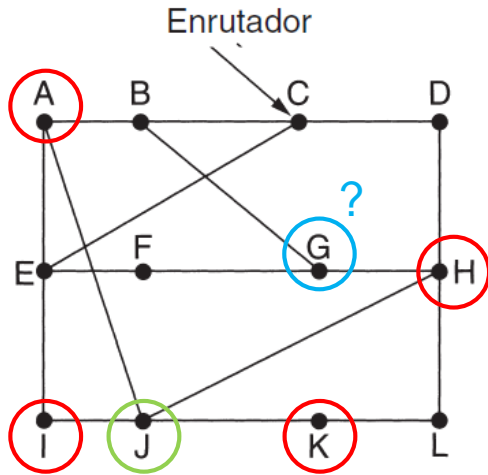
Distance Vector



Vector de Distancia

- ▶ Cada nodo mantiene una tabla para **todos** los nodos
 - ▶ (**Destination**, **Cost**, NextHop)
- ▶ Intercambia actualizaciones **solo con sus vecinos directamente conectados**
 - ▶ Periódicamente (en el orden de varios segundos)
 - ▶ Cuando su tabla cambia (se habla de una actualización gatillada)
- ▶ Cada actualización es una lista de pares:
 - ▶ (**Destination**, **Cost**)
- ▶ Se **modifica** la tabla local si se recibe una **mejor ruta**
 - ▶ Costo menor
 - ▶ Llegó desde el host próximo “next-hop”
- ▶ Se refrescan rutas existentes; se borran si hay time out

Vector de Distancia (ejemplo I)



	A	I	H	K	Nuevo retardo estimado desde J	
					↓	Línea next hop
A	0	24	20	21	8	A
B	12	36	31	28	20	A
C	25	18	19	36	28	I
D	40	27	8	24	20	H
E	14	7	30	22	17	I
F	23	20	19	40	30	I
G	18	31	6	31	18	H
H	17	20	0	19	12	H
I	21	0	14	22	10	I
J	9	11	7	10	0	-
K	24	22	22	0	6	K
L	29	33	9	9	15	K

Retardo JA es de 8 Retardo JI es de 10 Retardo JH es de 12 Retardo JK es de 6

Vectores recibidos de los cuatro vecinos de J

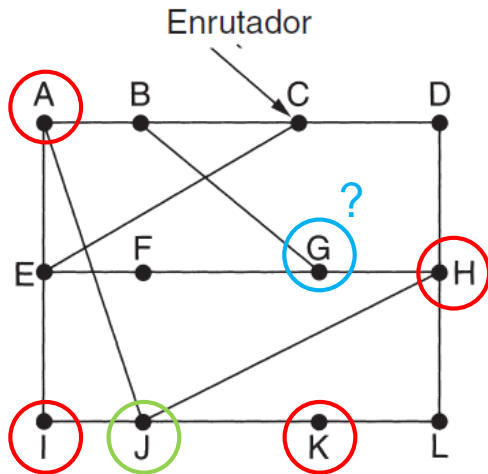
Nueva tabla de enrutamiento para J

(a)

(b)

(a) Subred. (b) Entrada de A, I, H, K y la nueva tabla de enrutamiento de J.

Vector de Distancia (ejemplo I)



	A	I	H	K		Nuevo retardo estimado desde J	Línea next hop
A	0	24	20	21		8	A
B	12	36	31	28		20	A
C	25	18	19	36		28	I
D	40	27	8	24		20	H
E	14	7	30	22		17	I
F	23	20	19	40		30	I
G	18	31	6	31		18	H
H	17	20	0	19		12	H
I	21	0	14	22		10	I
J	9	11	7	10		0	-
K	24	22	22	0		6	K
L	29	33	9	9		15	K

Retardo Retardo Retardo Retardo
JA es de 8 JI es de 10 JH es de 12 JK es de 6

Vectores recibidos de los cuatro vecinos de J

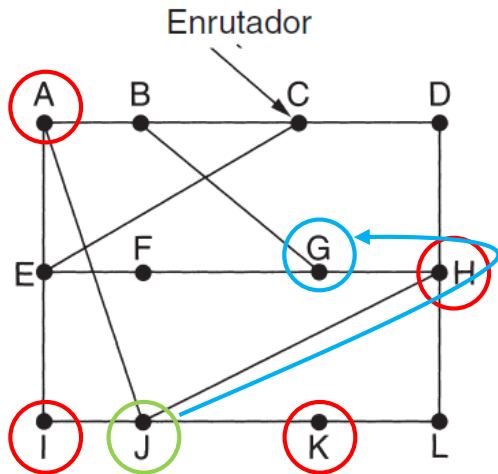
Nueva tabla de enrutamiento para J

(a)

(b)

(a) Subred. (b) Entrada de A, I, H, K y la nueva tabla de enrutamiento de J.

Vector de Distancia (ejemplo I)



Nuevo retardo estimado desde J

↓ Línea next hop

A	0	24	20	21	8	A
B	12	36	31	28	20	A
C	25	18	19	36	28	I
D	40	27	8	24	20	H
E	14	7	30	22	17	I
F	23	20	19	40	30	I
G	18	31	6	31	18	H
H	17	20	0	19	12	H
I	21	0	14	22	10	I
J	9	11	7	10	0	-
K	24	22	22	0	6	K
L	29	33	9	9	15	K

Retardo Retardo Retardo Retardo
JA es de 8 JI es de 10 JH es de 12 JK es de 6

Vectores recibidos de los cuatro vecinos de J

Nueva tabla de enrutamiento para J

Desde J Hacia G conviene vía H

(a)

(b)

(a) Subred. (b) Entrada de A, I, H, K y la nueva tabla de enrutamiento de J.

La tabla de ruteo (ejemplo I)

- ▶ Las primeras cuatro columnas de la parte (b) vectores de retardo recibidos de los vecinos del router J.
- ▶ *A indica tener un retardo de 12 mseg a B, un retardo de 25 mseg a C, un retardo de 40 mseg a D, etc.*
- ▶ *J ha medido o estimado el retardo a sus vecinos A, I, H y K en 8, 10, 12 y 6 mseg, respectivamente*
- ▶ *J calcula su nueva ruta al router G. Sabe que puede llegar a A en 8 mseg, y A indica ser capaz de llegar a G en 18 mseg, por lo que J sabe que puede contar con un retardo de 26 mseg a G si reenvía a través de A los paquetes destinados a G.*
- ▶ *J calcula el retardo a G a través de I, H y K en 41 ($31 + 10$), 18 ($6 + 12$) y 37 ($31 + 6$) mseg, respectivamente.*
- ▶ El mejor de estos valores es el 18, por lo que escribe una entrada en su tabla de enrutamiento indicando que el retardo a G es de 18 mseg, y que la ruta que se utilizará es vía H.

Distance Vector (ejemplo II)

- Cada nodo arma un vector que contiene la distancia (costos) a todos los nodos y lo envía a sus vecinos

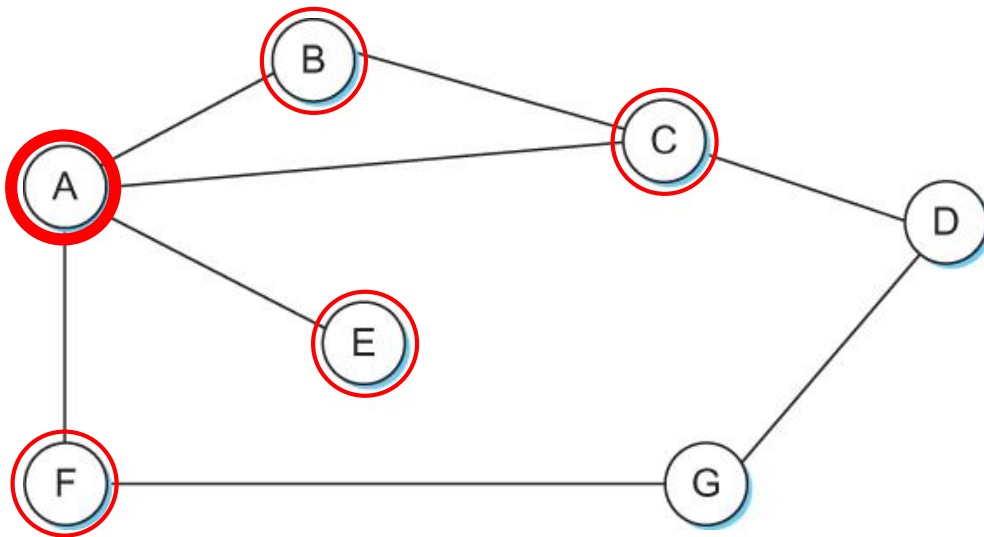


Tabla de ruteo del **Nodo A** ?

Destino	Costo	NextHop
B	1	B
C	1	C
D	2	C
E	1	E
F	1	F
G	2	F

- Suponemos acá que cada salto tiene costo igual a 1 al comenzar

Distance Vector (II)

Tabla de ruteo del **Nodo A** ?

Information Stored at Node	Distance to Reach Node						
	A	B	C	D	E	F	G
A	0	1	1	∞	1	1	∞
B	1	0	1	∞	∞	∞	∞
C	1	1	0	1	∞	∞	∞
D	∞	∞	1	0	∞	∞	1
E	1	∞	∞	∞	0	∞	∞
F	1	∞	∞	∞	∞	0	1
G	∞	∞	∞	1	∞	1	0

Estado inicial de distancia almacenado en cada nodo

Una visión global, ningún nodo conoce “todo el estado”.

Distance Vector (II)

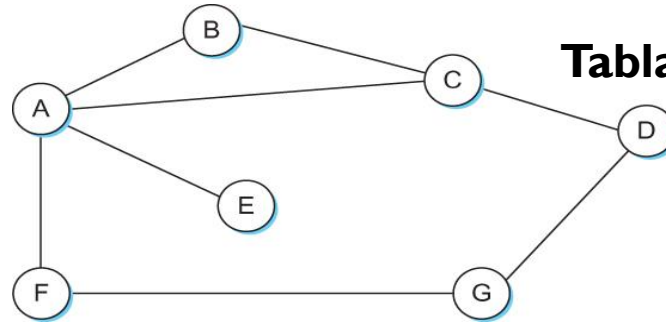


Tabla de ruteo del **Nodo A** ?

Destination	Cost	NextHop
B	1	B
C	1	C
D	∞	—
E	1	E
F	1	F
G	∞	—

Distance Vector (II)

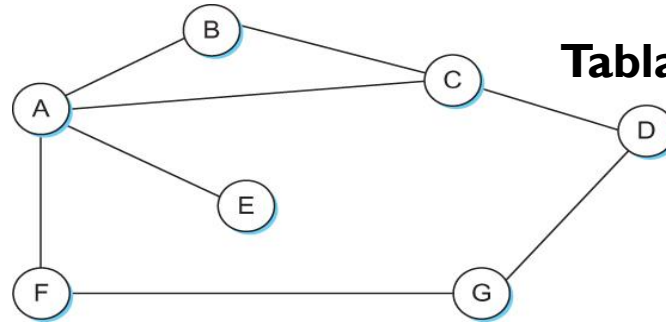


Tabla de ruteo del **Nodo A** ?

Destination	Cost	NextHop
B	1	B
C	1	C
D	2	C
E	1	E
F	1	F
G	2	F

Tabla de ruteo final de A

Distance Vector (II)

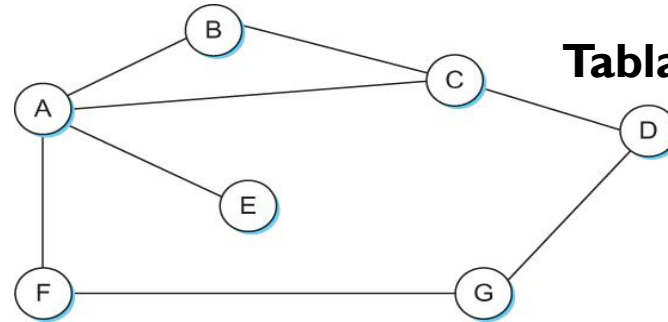


Tabla de ruteo del **Nodo A** ?

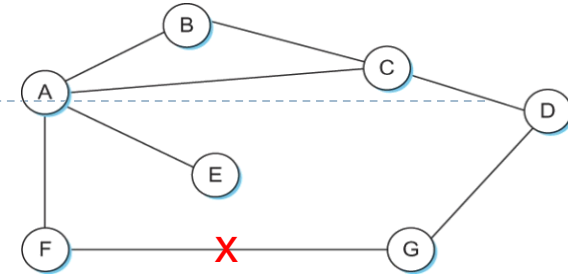
Information Stored at Node	Distance to Reach Node						
	A	B	C	D	E	F	G
A	0	1	1	2	1	1	2
B	1	0	1	2	2	2	3
C	1	1	0	1	2	2	2
D	2	2	1	0	3	2	1
E	1	2	2	3	0	2	3
F	1	2	2	2	2	0	1
G	2	3	2	1	3	1	0

Tabla final luego de una **convergencia global**

Ciclos de Actualización

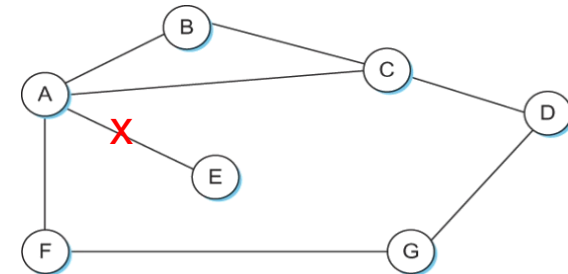
► Ejemplo 1 (caso feliz: estable)

- F detecta que su **enlace a G falla**.
- F fija distancia ∞ a **G** y envía una actualización a A
- A fija distancia ∞ a **G** porque A usa F para llegar a G
- A recibe actualización periódica de C con camino de **2 hops a G**
- A fija distancia a **G con 3** y envía actualización a F
- F decide que él puede llegar a **G en 4** hops vía A



► Ejemplo 2 (caso no feliz: inestable, conteo a infinito)

- Enlace de **A a E falla**.
- A comunica distancia ∞ a **E (timing ?)**
- B y C comunican distancia **2 a E (timing ?)**
- B decide que puede llegar a E en 3 hops; comunica esto a A
- A decide que puede llegar a E en 4 hops; comunica esto a C
- C decide que puede llegar a E en 5 hops...



Problema de conteo a Infinito

A	B	C	D	E	
•	•	•	•	•	Inicialmente
	1	•	•	•	Tras 1 intercambio
	1	2	•	•	Tras 2 intercambios
	1	2	3	•	Tras 3 intercambios
	1	2	3	4	Tras 4 intercambios

(a)

A	B	C	D	E	
•	1	2	3	4	Inicialmente
	3	2	3	4	Tras 1 intercambio
	3	4	3	4	Tras 2 intercambios
	5	4	5	4	Tras 3 intercambios
	5	6	5	6	Tras 4 intercambios
	7	6	7	6	Tras 5 intercambios
	7	8	7	8	Tras 6 intercambios
	⋮				
•	•	•	•	•	

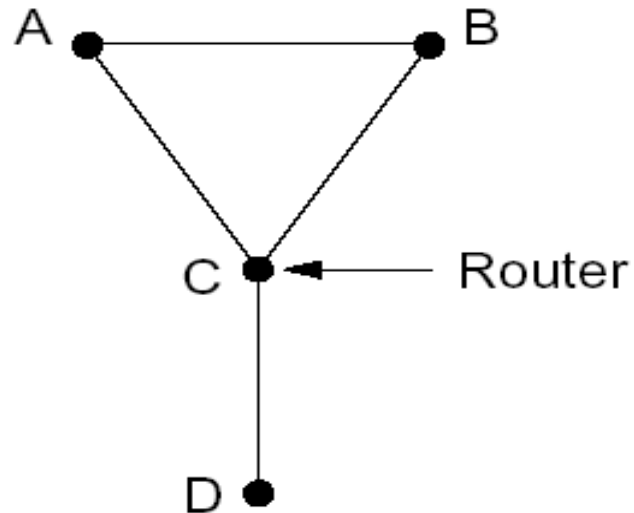
(b)

- (a) El nodo A, esta seteado como infinito, luego se activa
(b) El nodo A, esta activo, luego cae

Heurísticas para romper los ciclos

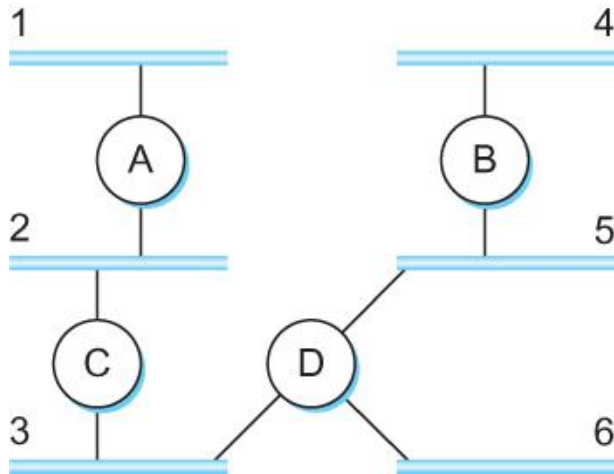
- ▶ Fijar **16 como infinito**, es decir cuando el costo llega a 16 se asume que no hay ruta al nodo
- ▶ Partir el horizonte (**Split horizon**): omite enviar información de distancia que fue aprendida por nodo al cual se le envía el vector
- ▶ Partir el horizonte con reverso venenoso (**Split horizon with poison reverse**): sí notifica entradas aprendidas desde el nodo al cual se envía el vector, pero a esos destinos se les pone costo infinito
- ▶ Las últimas dos técnicas sólo operan cuando el lazo involucra dos nodos
- ▶ La convergencia de este protocolo no es buena, se mejora con ruteo usando el **Estado de los Enlaces**

Ejemplo con Poison Reverse:



- (a) Todos los costos son uno
- (b) Falla el enlace C – D
- (c) A puede llegar a D por B
- (d)

Routing Information Protocol (RIP)

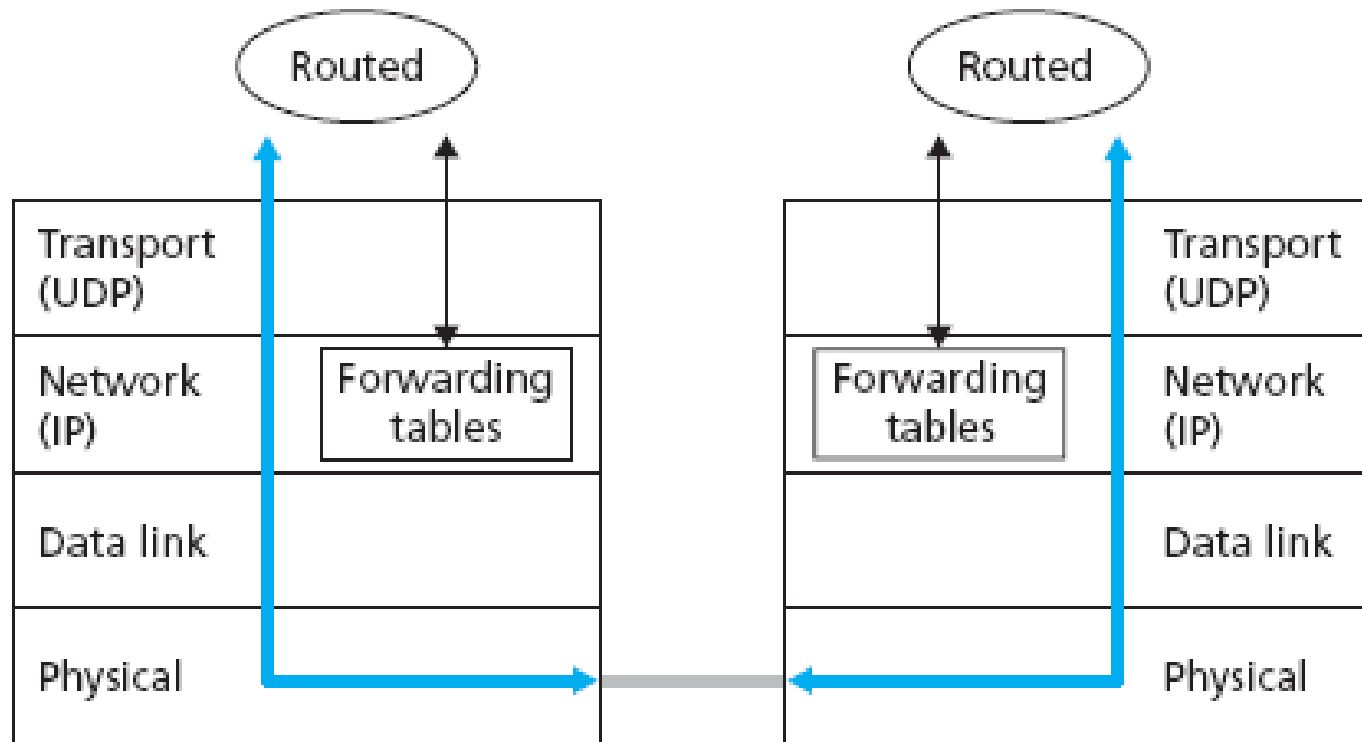


Red con RIP

0	8	16	31
Command	Version	Must be zero	
Family of net 1		Route Tags	
Address prefix of net 1			
Mask of net 1			
Distance to net 1			
Family of net 2		Route Tags	
Address prefix of net 2			
Mask of net 2			
Distance to net 2			

Formato del paquete RIPv2

RIP implementado como un “daemon”



RIP usa el protocolo de transporte UDP y tiene reservado el número de Puerto 520.

RIP implementado como un “daemon”

IP

```
Internet Protocol Version 4, Src: 10.1.12.2 (10.1.12.2), Dst: 255.255.255.255 (255.255.255.255)
  Version: 4
  Header length: 20 bytes
  Differentiated Services Field: 0xc0 (DSCP 0x30: Class Selector 6; ECN: 0x00: Not-ECT (Not ECN-Capable Transport))
    1100 00.. = Differentiated services codepoint: Class selector 6 (0x30)
    .... ..00 = Explicit Congestion Notification: Not-ECT (Not ECN-Capable Transport) (0x00)
  Total Length: 52
  Identification: 0x0000 (0)
  Flags: 0x00
    0... .... = Reserved bit: Not set
    .0.. .... = Don't fragment: Not set
    ..0. .... = More fragments: Not set
  Fragment offset: 0
  Time to live: 2
    [Expert Info (Note/Sequence): "Time To Live" only 2]
      [Message: "Time To Live" only 2]
      [Severity level: Note]
      [Group: Sequence]
  Protocol: UDP (17)
  Header checksum: 0xa1f7 [correct]
    [Good: True]
    [Bad: False]
  Source: 10.1.12.2 (10.1.12.2)
  Destination: 255.255.255.255 (255.255.255.255)
    [Source GeoIP: Unknown]
    [Destination GeoIP: Unknown]
```

UDP

```
User Datagram Protocol, Src Port: router (520), Dst Port: router (520)
  Source port: router (520)
  Destination port: router (520)
  Length: 32
  Checksum: 0xe48a [validation disabled]
    [Good Checksum: False]
    [Bad Checksum: False]
```

RIP

```
Routing Information Protocol
  Command: Request (1)
  Version: RIPv1 (1)
  Address not specified, Metric: 16
    Address Family: unspecified (0)
    Metric: 16
```



Link State

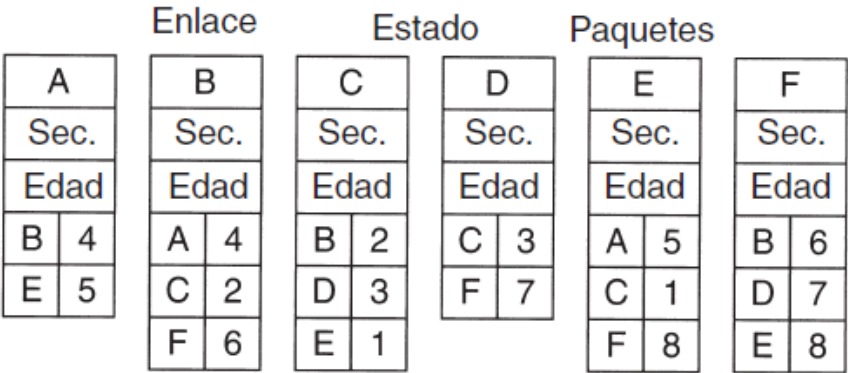


Algoritmo de Estado del Enlace

- ▶ La topología de red y costos **conocidos por todos los nodos**
- ▶ “Link State Broadcast”: Todos los nodos tienen **la misma** información
- ▶ Cómputo del camino mínimo: Dijkstra (forward search)

Algoritmo de Estado del Enlace

- ▶ Descubrir a sus vecinos y conocer sus direcciones de red
- ▶ Medir costo para cada uno de sus vecinos
- ▶ Construir un paquete que indique todo lo que acaba de aprender
- ▶ Enviar este paquete a todos los demás routers
- ▶ Calcular la ruta más corta a todos los nodos



Estado del Enlace

- ▶ Estrategia
 - ▶ Enviar a todos los nodos (no sólo los vecinos) información sobre **enlaces directamente conectados** (no la tabla completa).
 - ▶ Se “inunda” con la información.
- ▶ Paquete del estado del enlace (Link State Packet, LSP)
 - ▶ ID del router que creó el LSP
 - ▶ Costo del enlace a cada vecino directamente conectado
 - ▶ Número de secuencia (SEQNO)
 - ▶ Time-To-Live (TTL) para este paquete

Estado del Enlace (cont.)

- ▶ Inundación Confiable
 - ▶ Almacena el LSP más reciente de cada nodo
 - ▶ Re-envía LSP a todos excepto a quien me lo envió
 - ▶ Genera un nuevo LSP periódicamente
 - ▶ Incrementa SEQNO
 - ▶ Inicia SEQNO en 0 cuando se reinicia
 - ▶ Decrementa TTL de cada LSP almacenado
 - ▶ Descarta cuando TTL=0
 - ▶ **LSP ACK**

Cálculo de Ruta

- ▶ Algoritmo de Dijkstra para el camino más corto entre nodos
- ▶ Sean
 - ▶ N denota el conjunto de nodos del grafo
 - ▶ $l(i, j)$ denota costo no negativo (peso) para el arco (i, j) . Si no hay arco, el costo es infinito.
 - ▶ s denota este nodo
 - ▶ M denota el conjunto de nodos incorporados hasta ahora
 - ▶ $C(n)$ denota el costo del camino de s a n

$M = \{s\}$

for each n **in** $N - \{s\}$

$C(n) = l(s, n)$

while $(N \neq M)$

$M = M \text{ unión } \{w\} \text{ tal que } C(w) \text{ es el mínimo para todo } w \text{ en } (N - M)$

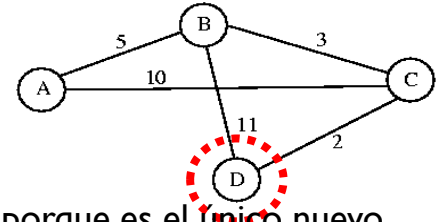
for each n **in** $(N - M)$

$C(n) = \text{MIN}(C(n), C(w) + l(w, n))$

Shortest Path Routing

- ▶ En la práctica el cálculo de la tabla de ruta se hace conforme los **LSP** (Link State Packet) van llegando, utilizando una implementación del algoritmo de Dijkstra llamada “*forward search algorithm*”
- ▶ Se manejan dos listas de entradas: las **tentativas** y las **confirmadas**.
- ▶ Cada una de esas listas tiene entradas:
(**Destination**, **Cost**, **NextHop**)

Uso del algoritmo de Dijkstra en ruteo



Paso	Confirmado	Tentativo	Comentario
1	(D,0,-)		<ul style="list-style-type: none"> Miramos el LSP de D porque es el único nuevo miembro confirmado
2	(D,0,-)	(B,11,B) (C,2,C)	<ul style="list-style-type: none"> El LSP de D nos dice cómo llegamos a B y C
3	(D,0,-) (C,2,C)	<u>(B,11,B)</u>	<ul style="list-style-type: none"> El miembro de menor costo entre los tentativos es C. Se examina el LSP del nodo recién confirmado
4	(D,0,-) (C,2,C)	<u>(B,5,C)</u> (A,12,C)	<ul style="list-style-type: none"> Con C, se actualizan los costos. Ahora llegamos a B vía C y se incorpora A
5	(D,0,-) (C,2,C) (B,5,C)	<u>(A,12,C)</u>	<ul style="list-style-type: none"> Se mueve el de menor costo de tentativos a confirmados.
6	(D,0,-) (C,2,C) (B,5,C)	<u>(A,10,C)</u>	<ul style="list-style-type: none"> Con B, se actualizan los costos. Ahora se llega a A vía B. Después de esto se mueve el menor costo de los tentativos a los confirmados. A es el único y terminamos.

(**Destination**, **Cost**, **NextHop**)

Comparación entre algoritmo Vector de Distancia y Estado de Enlaces.

- ▶ En el algoritmo **Vector de Distancia** cada nodo transmite a sus vecinos todo lo que sabe respecto de toda la red (distancia a todos los nodos).
 - ▶ Puede ser inestable. **Conteo a infinito.**
- ▶ En el algoritmo **Estado de Enlaces** cada nodo transmite a toda la red lo que sabe de sus vecinos (solo el estado de sus vecinos)
 - ▶ Es **estable**, no genera demasiado tráfico y responde rápido a cambios de topología.
 - ▶ Su problema es la cantidad de información que debe ser almacenada en los nodos (un LSP por nodo)

Métricas

- ▶ Métrica ARPANET original
 - ▶ Mide el número de paquetes encolados hacia cada enlace
 - ▶ No toma en cuenta latencia ni ancho de banda
- ▶ Métrica ARPANET nueva
 - ▶ Marca cada paquete entrante con su tiempo de llegada (**arrival time, AT**)
 - ▶ Graba tiempo de salida (**departure time, DT**)
 - ▶ Cuando llega el ACK del enlace de datos, calcula:
Delay = (DT - AT) + Transmit + Latency
Transmit y Latency son parámetros estáticos del enlace.
 - ▶ Si hay timeout, reset **DT** a tiempo de salida para retransmisión
 - ▶ Costo del enlace = retardo promediado sobre algún período
- ▶ Mejora fina (ver “revised ARPANET routing metric [1]”)
 - ▶ Reducir el “rango dinámico” para el costo
 - ▶ Moderación de la diferencia entre el peor y el mejor costo.
 - ▶ En lugar del retardo se emplea la utilización del enlace

OSPF (Open Shortest Path First)

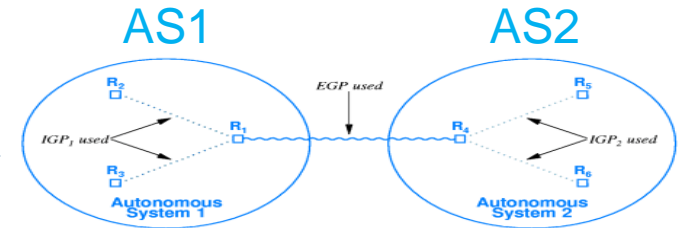
- ▶ Open=“Abierto”: disponible públicamente.
- ▶ Utiliza un algoritmo de Estado de Enlaces:
 - ▶ Distribución de paquetes del estado de enlaces.
 - ▶ Mapa de la topología en cada nodo.
 - ▶ Cómputo de la ruta usando el algoritmo de Dijkstra.
- ▶ El anuncio OSPF lleva una entrada por cada router vecino.
- ▶ Anuncios distribuidos a **todos nodos** de los **AS** (mediante inundación):
 - ▶ Transportados por mensajes en OSPF directamente sobre IP (mejor que TCP o UDP ?)

OSPF

- ▶ Seguridad: todos los mensajes OSPF están autenticados (para prevenir intrusiones malignas).
- ▶ Múltiples caminos permitidos con el mismo costo (c.f. sólo un camino en RIP).
- ▶ Para cada enlace, varias métricas de coste para distintos ToS (por ejemplo, coste del enlace satélite “rebajado” para un mayor rendimiento; alto para tiempo real)
- ▶ Soporte integrado uni- y multidifusión (Unicast - Multicast):
 - ▶ La multidifusión OSPF (MOSPF) usa la misma base de datos topológica que OSPF.
- ▶ OSPF jerárquico para grandes dominios (escala).

OSPF jerárquico

Intercambio de escalabilidad por optimalidad



AS2

Router Frontera

Router Troncal

Troncal

Routers de Frontera de Área

Routers internos

Área 3


Área 1

Área 2

OSPF jerárquico

- ▶ **Dos niveles** de jerarquía: **área local, área troncal.**
 - ▶ Anuncios de estado de enlace **sólo en el área.**
 - ▶ Cada nodo detalla la topología del área; sólo conoce la dirección (el camino más corto) a las **redes de otras áreas.**
- ▶ **Routers de Frontera de Área:** “resumen” las distancias a las redes del mismo área, anuncian a otros routers de Frontera de Área.
- ▶ **Routers Troncales:** ejecutan ruteo de **OSPF** limitados al área troncal.
- ▶ **Routers Frontera:** conectan con **otros AS.**

Tipos de Mensaje en OSPF



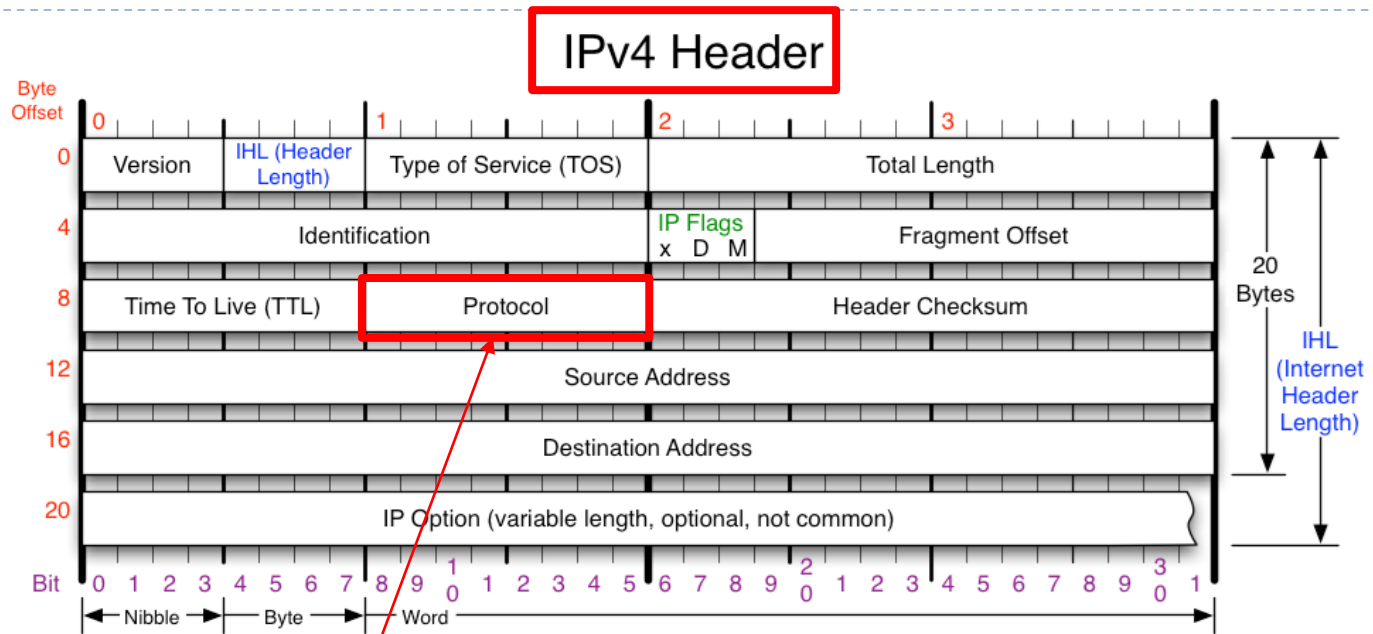
Tipo de mensaje	Descripción
Hello	Descubre quiénes son los vecinos
Link state update	Proporciona los costos del emisor a sus vecinos
Link state ack	Confirma la recepción de la actualización del estado del enlace
Database description	Anuncia qué actualizaciones tiene el emisor
Link state request	Solicita información del socio

Confiabilidad: Se confirman los *database description*, *LS request* y *LS update*

Tipos de Mensaje en OSPF

- ▶ Cada router inunda periódicamente con mensajes LINK STATE UPDATE a cada uno de routers adyacentes. Este mensaje da su estado y proporciona los costos usados en la base de datos topológica.
- ▶ **Para hacerlos confiables, se confirma la recepción de los mensajes de inundación.** Cada mensaje tiene un número de secuencia para que un enrutador pueda ver si un LINK STATE UPDATE entrante es más viejo o más nuevo que el que tiene actualmente. Los routers también envían estos mensajes cuando una línea su costo cambia.
- ▶ Los mensajes DATABASE DESCRIPTION dan los números de secuencia de todas las entradas de estado del enlace poseídas por el emisor actualmente. Comparando sus propios valores con los del emisor, el receptor puede determinar quién tiene los valores más recientes. Estos mensajes se usan cuando se activa una línea.
- ▶ Cualquier socio puede pedir información del estado del enlace al otro usando los mensajes LINK STATE REQUEST.
- ▶ El resultado de este algoritmo es que cada par de enrutadores adyacentes hace una verificación para ver quién tiene los datos más recientes, y de esta manera se difunde la nueva información a lo largo del área.

Open Shortest Path First (OSPF)



Version

Version of IP Protocol. 4 and 6 are valid. This diagram represents version 4 structure only.

Header Length

Number of 32-bit words in TCP header, minimum value of 5. Multiply by 4 to get byte count.

Protocol

IP Protocol ID. Including (but not limited to):

1 ICMP	17 UDP	57 SKIP
2 IGMP	47 GRE	88 EIGRP
6 TCP	50 ESP	89 OSPF
9 IGRP	51 AH	115 L2TP

Total Length

Total length of IP datagram, or IP fragment if fragmented. Measured in Bytes.

Fragment Offset

Fragment offset from start of IP datagram. Measured in 8 byte (2 words, 64 bits) increments. If IP datagram is fragmented, fragment size (Total Length) must be a multiple of 8 bytes.

Header Checksum

Checksum of entire IP header

IP Flags

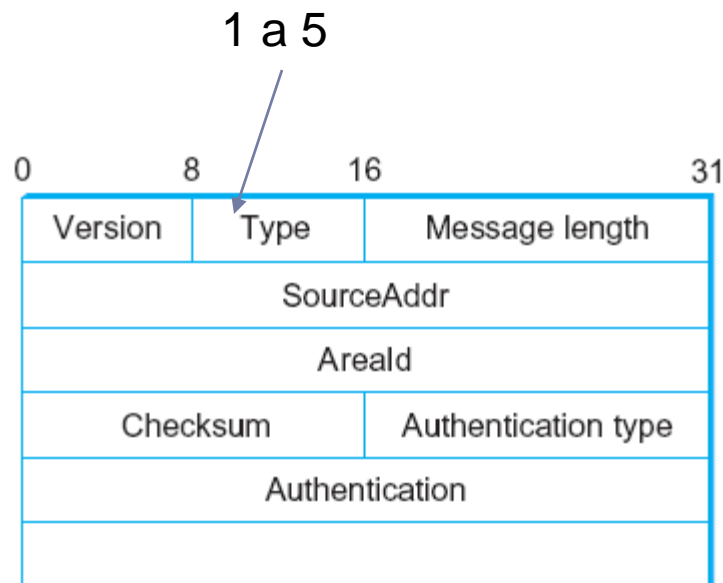
x	D	M
---	---	---

x 0x80 reserved (evil bit)
 D 0x40 Do Not Fragment
 M 0x20 More Fragments follow

RFC 791

Please refer to RFC 791 for the complete Internet Protocol (IP) Specification.

Open Shortest Path First (OSPF)



Formato header OSPF

LS Age		Options		Type = 1
Link-state ID				
Advertising router				
LS sequence number				
LS checksum			Length	
0	Flags	0	Number of links	
Link ID				
Link data				
Link type		Num_TOS	Metric	
Optional TOS information				
More links				

Formato de “OSPF Link State Advertisement” (LSA)



BGP



Ruteo Interdominio

EGP: Exterior Gateway Protocol

- ▶ Generalidades

- ▶ Diseñado para una Internet estructurada como árbol
- ▶ Se preocupa de *alcanzar* los nodos, no optimiza rutas

- ▶ Mensajes del Protocolo

- ▶ Adquisición de vecinos:

- ▶ Un router requiere que otro sea su par
- ▶ Pares intercambian información de alcance

- ▶ Alcance de vecinos:

- ▶ Un router periódicamente prueba si el otro es aún alcanzable
- ▶ Intercambia mensajes HELLO/ACK

- ▶ Actualización de rutas:

- ▶ Pares periódicamente intercambian sus tablas de ruteo (vector de distancia)

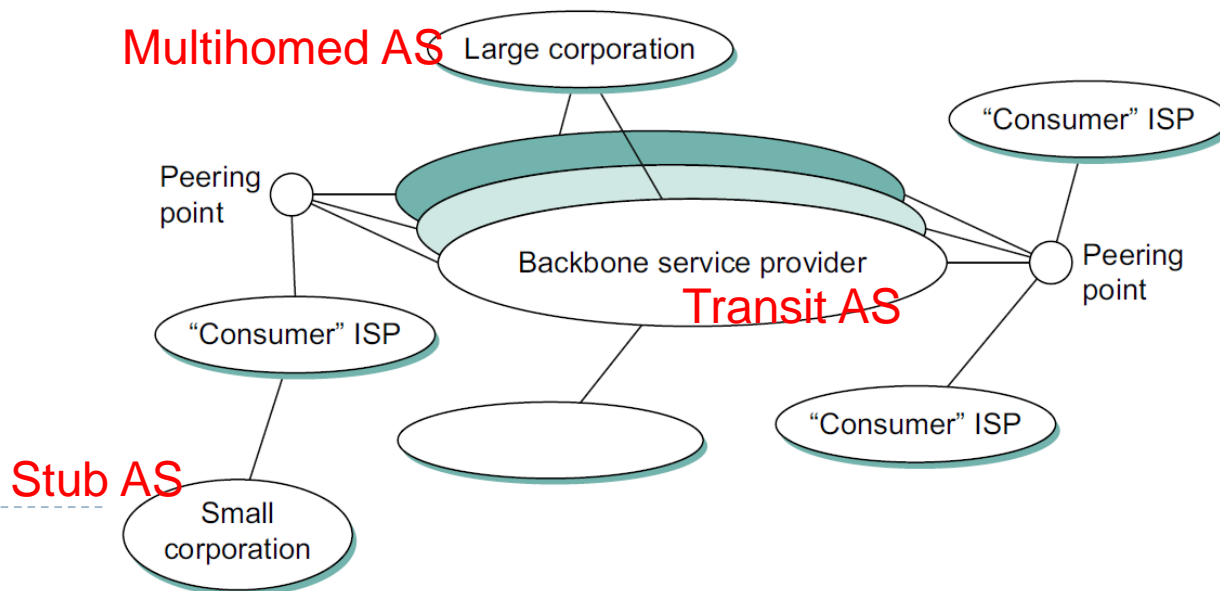
BGP: Exterior Gateway Protocol

► Generalidades

- Internet organizado como Sistemas Autónomos
 - Cada uno bajo el control de **una única entidad administrativa**
 - Una forma más (!) de agregar jerárquicamente información de enrutamiento **para alcanzar gran escala (internet)**
- El problema del ruteo se divide en dos
 - Adentro o Afuera de un Sistema Autónomo
- BGP se ocupa de ruteo Afuera de los sistemas autónomos
 - Ruteo **interdominio**
 - Cada AS puede ejecutar su propio ruteo **intradominio**
- BGP resuelve el problema de que diferentes AS compartan información de alcanzabilidad entre ellos
 - Cuales rangos de direcciones IP pueden alcanzarse en cada AS
 - Por cual ruta se puede llegar de un AS a otro AS

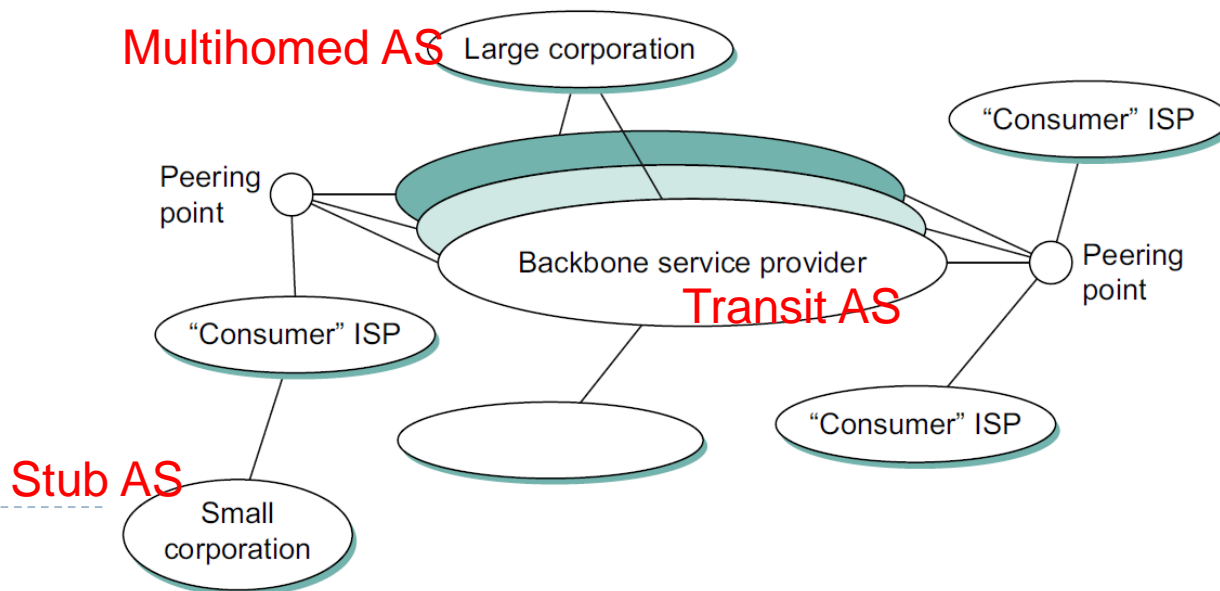
BGP: Border Gateway Protocol

- ▶ BGP no requiere jerarquía de árbol
- ▶ Tipos de AS
 - ▶ **stub AS**: tiene una única conexión a otro AS
 - ▶ transporta sólo tráfico local
 - ▶ **multihomed AS**: tiene conexiones a más de un AS
 - ▶ no transporta tráfico en transito
 - ▶ **transit AS**: tiene conexiones a más de un AS
 - ▶ transporta ambos tráfico local y en transito



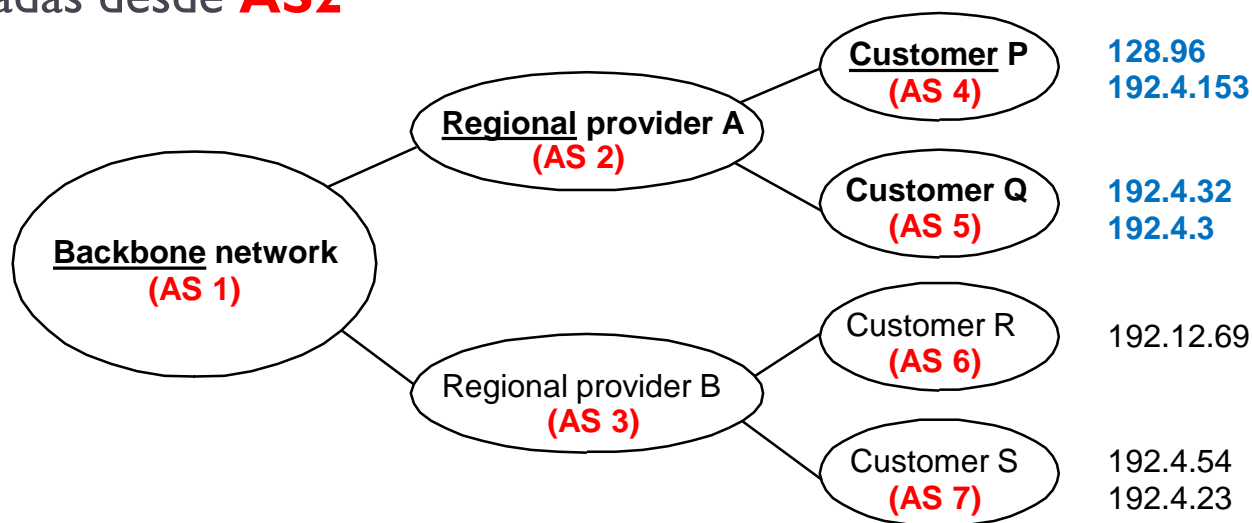
BGP: Border Gateway Protocol

- ▶ Cada AS tiene:
 - ▶ Uno o más Routers de Borde
 - ▶ Al menos un “**Portavoz**” BGP que publica:
 - ▶ Redes locales
 - ▶ Otras redes alcanzables (sólo el transit AS)
 - ▶ Información de rutas (paths AS)



Ejemplo BGP

- ▶ El **Portavoz** para **AS2** publica alcanzabilidad a **P** y **Q**
 - ▶ Redes **128.96**, **192.4.153**, **192.4.32**, y **192.4.3**, pueden ser directamente alcanzadas desde **AS2**



- ▶ El **Portavoz** para **Backbone** publica que
 - ▶ Redes **128.96**, **192.4.153**, **192.4.32** y **192.4.3** pueden ser alcanzadas a lo largo de la **ruta (AS1, AS2)**.
- ▶ Un **Portavoz** puede cancelar una ruta publicada antes

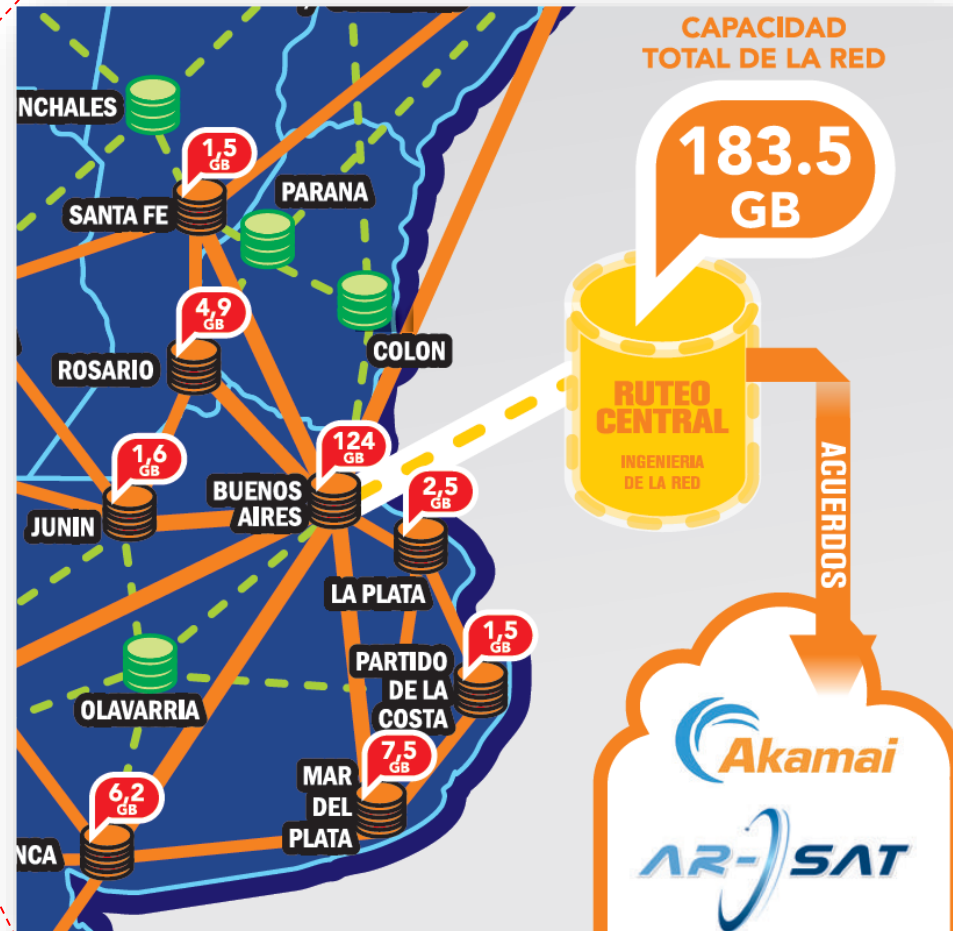
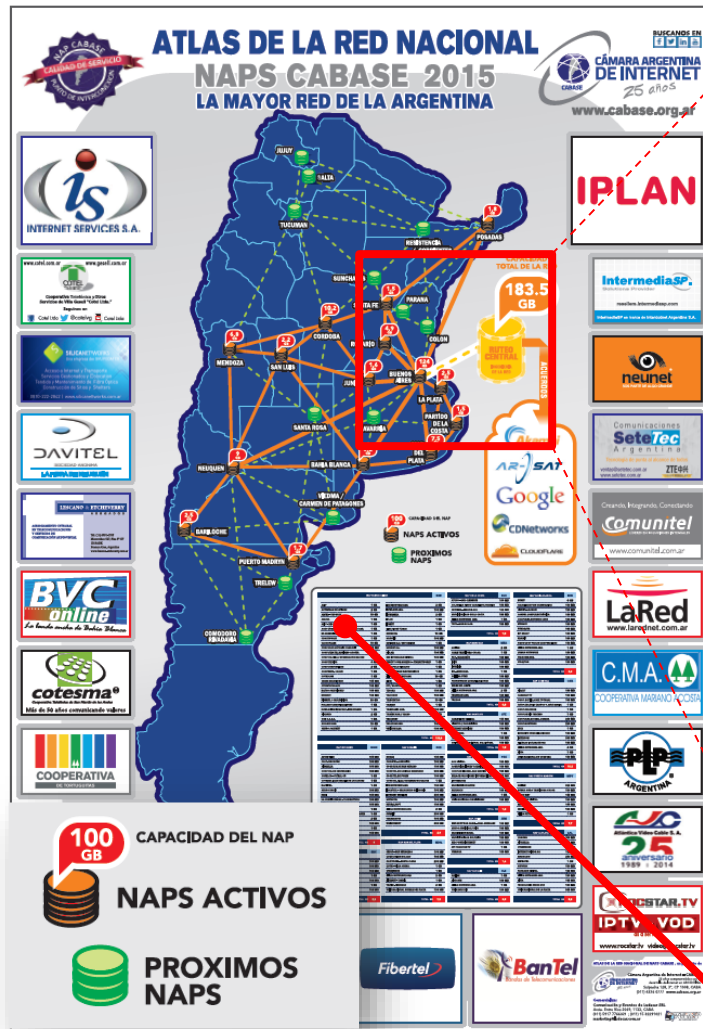
BGP y los Network Access Points (NAP)

CABASE: Cámara Argentina de Internet

- ▶ “Los NAPs (Network Access Points por sus siglas en inglés) o **también conocidos como IXs (Internet eXchanges)** son componentes fundamentales de la Red Internet.
- ▶ A través de un NAP, se produce el intercambio de tráfico entre las redes de diversas entidades (operadores, proveedores de acceso, organismos de gobierno, entidades académicas, etc.)
- ▶ Estos **puntos neurálgicos de la Red** se han construido en todo el mundo bajo distintos esquemas institucionales, topológicos y operacionales.
- ▶ No obstante, la mayoría de ellos persigue idénticos objetivos: eficientizar el ruteo de Internet, mejorando la calidad de servicio y minimizar los costos de interconexión.
- ▶ Todos los **NAPs CABASE** siguen el modelo cooperativo. Todos los miembros de los NAPs CABASE son socios de la Cámara Argentina de Internet que tienen como objetivo mejorar la calidad en las comunicaciones y reducir costos.” [1]

BGP y los Network Access Points (NAP)

<http://www.cabase.org.ar/wp-content/uploads/2015/07/Poster-Cabase15-06-20151.pdf>



CABASE: Cámara Argentina de Internet

BUE -> ARIU -> UBA 157.92/16
REDES DE INTERCONEXION UNIVERSITARIA

BGP: Los NAPs ...

- ▶ Brasil tiene muchos NAPs, motivos ?
- ▶ Como se organizan las interconexiones entre proveedores de servicios ? Caso de un ISP que quedó con ruteo internacional ...
- ▶ Hoy se busca que los NAPs accedan a CDNs ...

Bibliografía Básica

- ▶ Computer Networks, (Fifth Edition):A Systems Approach. Larry L. Peterson, Bruce S. Davie 2011
 - ▶ Routing – págs. 240-466
- ▶ Computer Networks (5th Edition) Andrew S.Tanenbaum, David J.Wetherall 2010
 - ▶ Routing Algorithms - págs. 392-404

Algunas Referencias

▶ RIPV1

- ▶ RFC-1058 - Charles Hedrick, June 1988

▶ RIPV2

- ▶ RFC 1388 (Jan. 1993), RFC 1723 (Nov. 1994), RFC 2453 (Nov. 1998),,
Gary Malkin

▶ OSPFV1

- ▶ RFC 1131 J. Moy, Oct. 1989

▶ OSPFV2

- ▶ RFC 1247 (July 1991), RFC 1583 (March 1994), RFC 2178 (July 1997),
RFC 2328 (April 1998), J. Moy