Protocolos de Ruteo

TPI-UNQ
Leandro Moscheni
Luis Zagarella

Sumario

- Introducción
- Clasificación
- Forwarding vs Routing
- Distance Vector Routing
- Flooding
- Link State Routing
- RIP
- Conclusiones

Introducción

- ***¿Que es el Routing?***
- *¿Que son los Protocolos de Ruteo?*
- *Definiendo el camino óptimo*

¿Que es el Routing?

Se lo podría definir como la función de buscar un camino determinado(basado en algún criterio “X”) entre todos los posibles caminos que hay, para llegar de un punto a otro en una red.

Introducción

- *¿Que es el Routing?*
- ***¿Que son los Protocolos de Ruteo?***
- *Definiendo el camino óptimo*

¿Que son los protocolos de Ruteo?

Son las diferentes estrategias(Algoritmos), que se pueden utilizar para seleccionar el camino optimo para llevar información de un punto “A” a un punto “B”, también denominados *nodos*.

Introducción

- ¿Que es el Routing?
- *¿Que son los Protocolos de Ruteo?*
- ***Definiendo el camino óptimo***

Definiendo el camino óptimo

Para decir que un camino es óptimo o mas conveniente, es necesario establecer un criterio de evalución, en este caso nos regiremos por dos parámetros:

- *Métrica*
- *Mejor Ruta*

Métrica

Esta relacionado con los aspectos que podemos cuantificar, por ejemplo la cantidad de nodos que hay que recorrer para llegar de un punto a otro, a eso se lo denomina saltos.

Mejor Ruta

Se entiende como “Mejor Ruta” aquella que:

- Consigue mantener acotado el Retardo entre nodos de la red.
- Consigue ofrecer altas cadencias efectivas independientemente del retardo medio del tránsito.
- Ofrece el menor costo.

Clasificación

- **Definiendo un Sistema Autónomo**
- Según su Dominio
- *Según su Funcionamiento*

Sistema Autónomo

Se define a un SA, como un conjunto de redes IP (maquinas y routers) administrado por una única entidad(a veces un grupo de ellas), que poseen una política propia de la gestión del trafico interno. A cada SA, se lo identifica con un ASN(Número de SA) Por ejemplo se puede considerar una SA a una red LAN.

Clasificación

- Definiendo un Sistema Autónomo
- **Según su Dominio**
- *Según su Funcionamiento*

Según su Dominio

Según su dominio significa dentro de que limites trabaja, en otras palabras si el protocolo trabaja dentro de un Sistema Autónomo(SA) o si trabaja comunicando a dos SA.
A partir de esto se desprenden dos subcategorías:

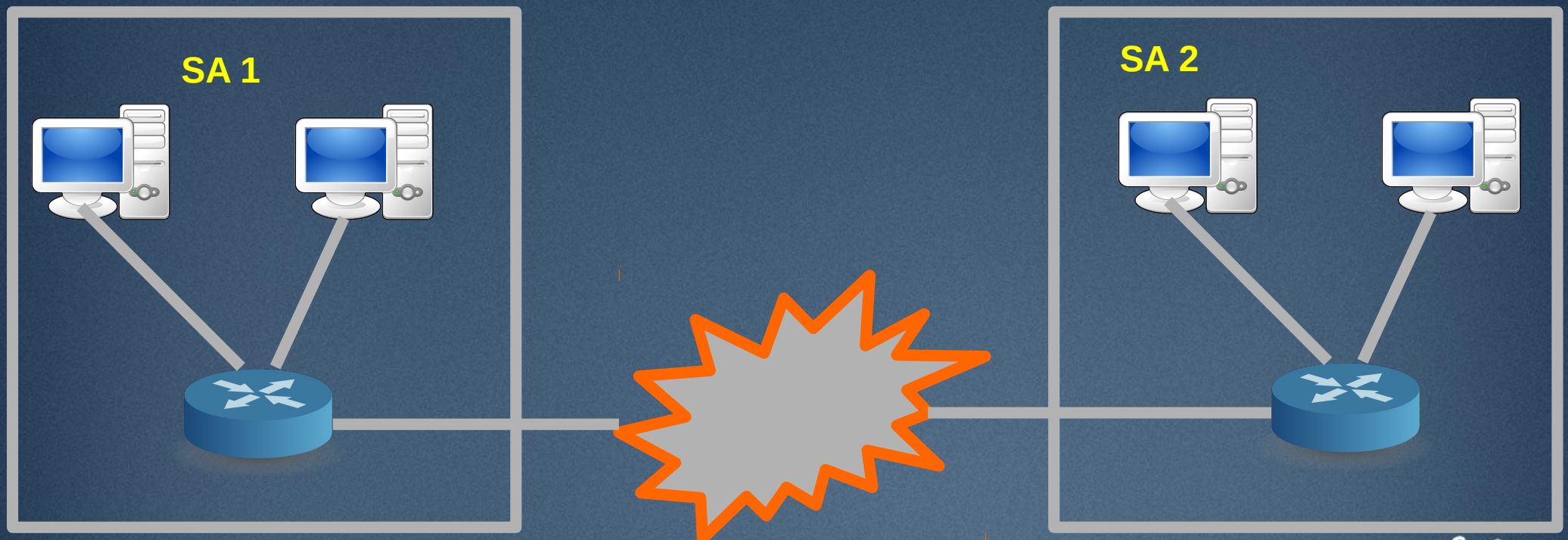
- *Interior Gateway Protocol(IGP)*
- *Exterior Gateway Protocol(EGP)*

Interior Gateway Protocol

Son los protocolos utilizados dentro de un SA, ejemplos de estos protocolos son RIP, IS-IS, IGRP, etc.

Exterior Gateway Protocol

Son empleados para la comunicación entre SA. Ejemplos de estos pueden ser EGP y BGP.



Clasificación

- Definiendo un Sistema Autónomo
- Según su Dominio
- ***Según su Funcionamiento***

Según su Funcionamiento

Cuando hablamos de funcionamiento, hablamos de la forma en la generan la información de ruteo, o también de como generan las Tablas de Ruteo o Forwarding.
A partir de esto, se puede subclasificar a estos algoritmos en dos:

- Determinísticos/Estáticos
- Adaptativos/Dinámicos

Determinísticos/Estáticos

Se caracterizan por:

- No tener en cuenta el estado de la subred al decidir el camino óptimo.
- Tener tablas de Ruteo Estáticas configuradas manualmente.
- El calculo del camino óptimo es offline, haciendo que no importe ni la complejidad algorítmica, ni los tiempos de convergencia requeridos.

Determinísticos/Estáticos

Por eso es que estos algoritmos son simples, y rápidos ya que su implementación es fácil. Aunque presentan una gran contra, y es que suelen ser los que peores decisiones toman.
Un ejemplo de estos algoritmos es el Dijkstra.

Adaptativos/Dinámicos

Estos algoritmos son mas tolerantes a cambios en las subredes, como lo pueden ser la variación del tráfico, el incremento del retardo o fallos en la topología de las redes.

Adaptativos/Dinámicos

A su vez, estos algoritmos se pueden clasificar en tres subcategorías dependiendo de donde se tomen las decisiones, las cuales son:

- *Adaptativo Centralizado:* hay un nodo central que recolecta los datos y con ellos genera la tabla de ruteo.
- *Adaptativo Distribuido:* cada nodo de la red calcula su propia tabla, utilizando su propia información y la de sus vecinos.
- *Adaptativo Aislado:* su respuesta a los cambios, se obtiene a partir de su información propia.

Forwarding vs Routing

Los podemos definir como:

- *Forwarding:* Es la selección de un puerto de salida basándose en la dirección de destino y la tabla de ruteo.
- *Routing:* Es el proceso por el que se *arma/genera* la tabla de ruteo.

Distance Vector Routing

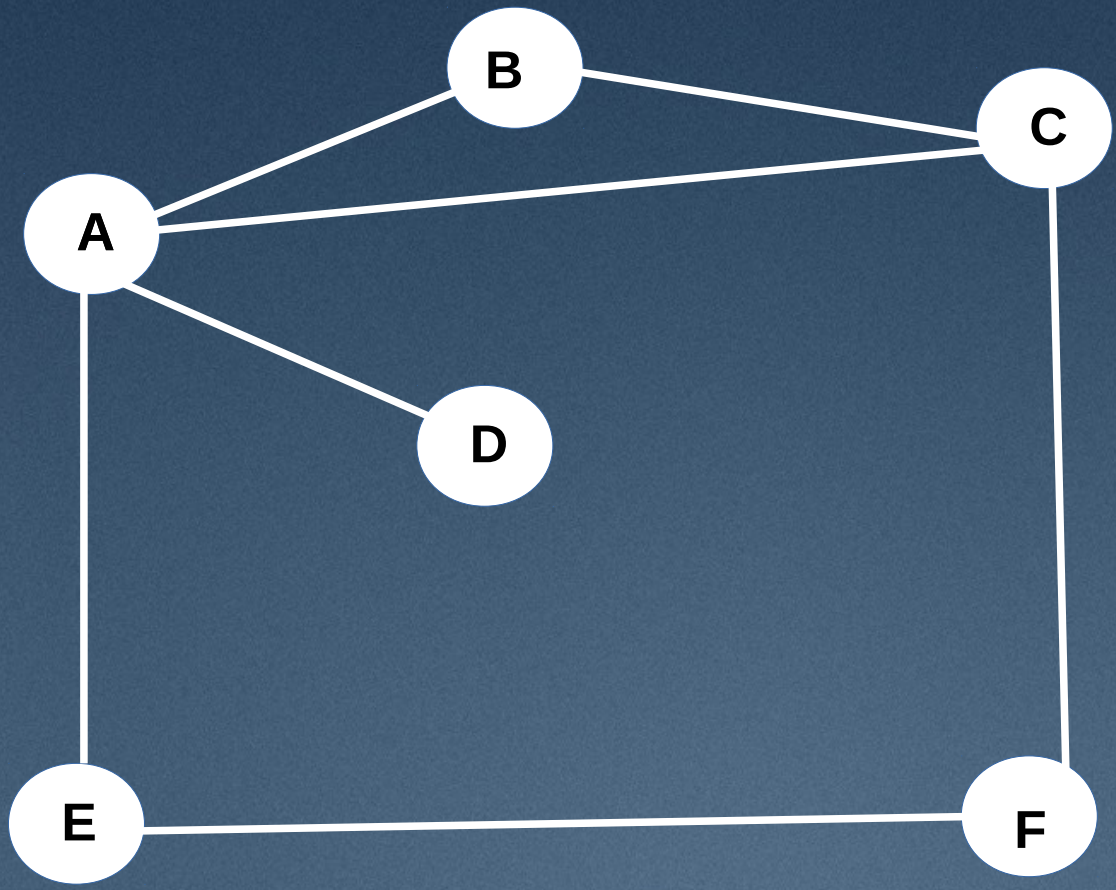
Este método de Routing esta basado en el algoritmo de Bellman-Ford, y la estrategia es que cada nodo debe generar su propia tabla de ruteo, y avisar periódicamente a sus vecinos los cambios en ella.

Distance Vector Routing

Para generar esta tabla se calcula la distancia a todos los enlaces de la red.
En la practica lo que realiza el algoritmo es generar un arreglo unidimensional(vector) que contiene las distancias(costos) a todos los otros nodos, y luego comparte ese mismo vector con el resto de sus vecinos inmediatos.

Distance Vector Routing

Para esto se supone que cada nodo conoce el costo de enlace de sus vecinos inmediatos(osea 1).



Distance Vector Routing

Las tablas de Ruteo que se generan tienen la siguiente forma:

Destino	Costo	Siguiente Salto
...	...	...
...	...	...
...	...	...
...	...	...

Distance Vector Routing

En el caso del nodo A, esta tabla iniciaría en un principio con estos valores:

Destino	Costo	Siguiente Salto
B	1	B
C	1	C
D	1	D
E	1	E
F	infinito	-

Distance Vector Routing

Y una vez que termine de generarla quedaría seteada con los siguientes valores:

Destino	Costo	Siguiente Salto
B	1	B
C	1	C
D	1	D
E	1	E
F	2	E

Distance Vector Routing

Ahora esta seria una vista general de como quedarían los vectores de distancia en cada nodo.

	A	B	C	D	E	F
A	0	1	1	1	1	2
B	1	0	1	2	2	2
C	1	1	0	2	2	1
D	1	2	2	0	2	3
E	1	2	2	2	0	1
F	2	2	1	3	1	0

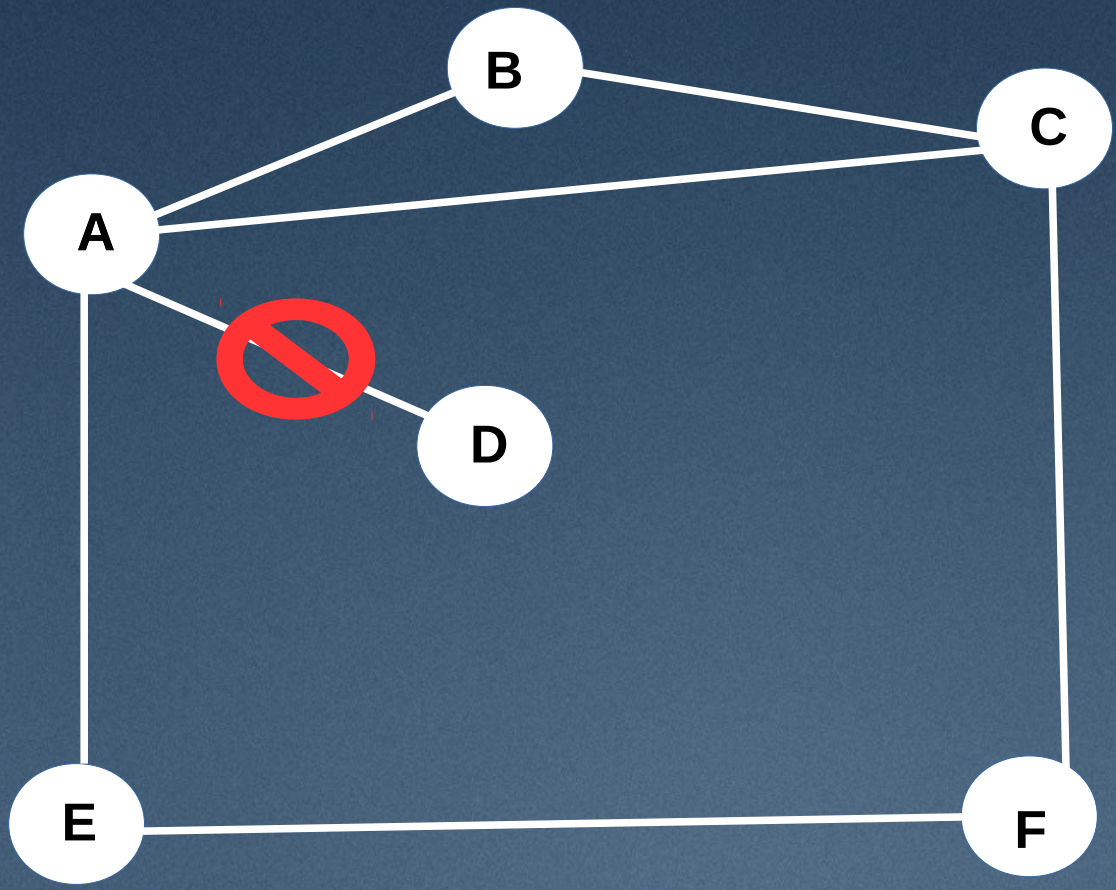
Distance Vector Routing

Estas tablas son dinámicas, esto quiere decir que cambiaran con el tiempo. Estas actualizaciones pueden ser de dos tipos:

- Periódicas: se envían aunque nada haya cambiado, simplemente para mostrar que el enlace esta vivo.
- Triggered: son enviadas por los nodos cuando un enlace falla o cambia su tabla.

Distance Vector Routing

Pero la vida no es solo felicidad, y el algoritmo puede fallar...
Por ejemplo al caerse el enlace (A → D)



Distance Vector Routing

Al pasar esto, en la siguiente ronda de actualización pasaría lo siguiente:

- **A** pone a infinito a **D**.
- **B** le dice a **A**, que **D** esta a una distancia de *2*.
- **C** le dice a **A**, que **D** esta a una distancia de *2*.

Dependiendo del timming en el que se den los eventos, podría ocurrir esto...

Distance Vector Routing

- El nodo **B**, después de escuchar que **D** puede ser alcanzado en *2* saltos desde **C**, concluye que puede alcanzar a **D** en *3* saltos y anuncia esto a **A**
- El nodo **A** concluye que puede alcanzar a **D** en *4* saltos y anuncia esto a **C**
- El nodo **C** concluye que puede alcanzar a **D** en *5* saltos; y así sucesivamente

Distance Vector Routing

Este ciclo solo se termina cuando el numero de saltos incrementa lo suficiente como para considerarlo inalcanzable. A este problema se lo denomina ***Count to infinity problem***.

Distance Vector Routing

Existen diferentes métodos para mitigar este problema, entre ellos se destacan:

- ***Limite de Saltos:*** pone un limite de saltos como aproximación a infinito. Esto hace que por ejemplo si se necesitan mas de 15 saltos, se descarte el enlace como inalcanzable.

- ***Split Horizon:*** que un nodo A no envié a un nodo B, información que aprendió de el.

Distance Vector Routing

- ***Split Horizon with poison reverse:*** que un nodo A envié a un nodo B, la información que aprendió de el, pero en forma negativa, para que sea descartada.

- ***Hold down:*** Cada nodo ignora las actualizaciones por un tiempo suficiente como para que se propaguen hacia todos los nodos.

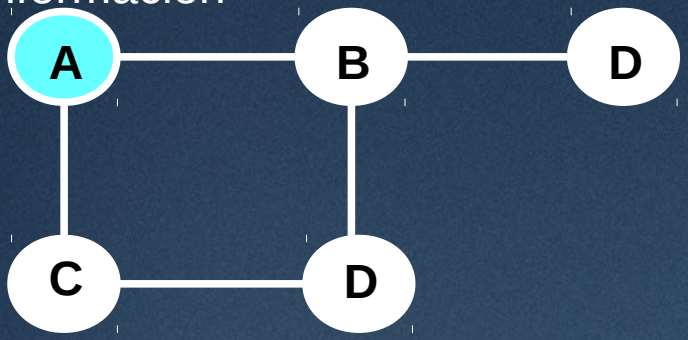
Flooding

Flooding o inundación, es un algoritmo recursivo, que lo que hacer es enviar información a todos sus nodos vecinos, y ellos a su vez la envían a todos sus vecinos menos al nodo del que proviene la información, generando una “reacción en cadena”

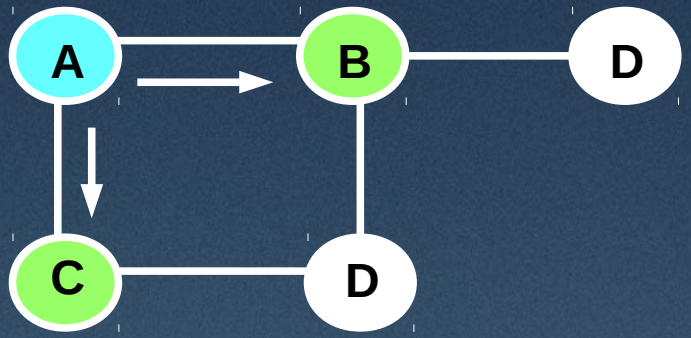
Flooding

Una secuencia en pasos de esto, seria la siguiente:

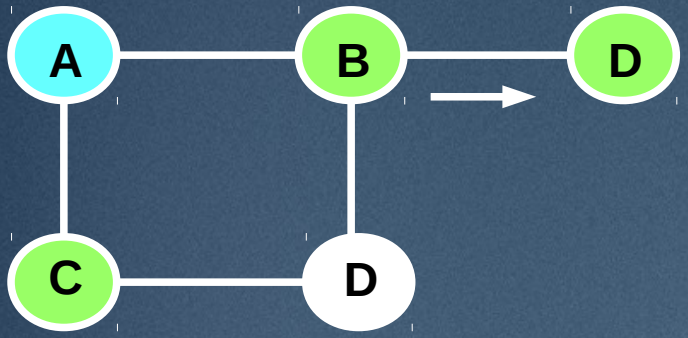
Estado inicial, nodo A con la información



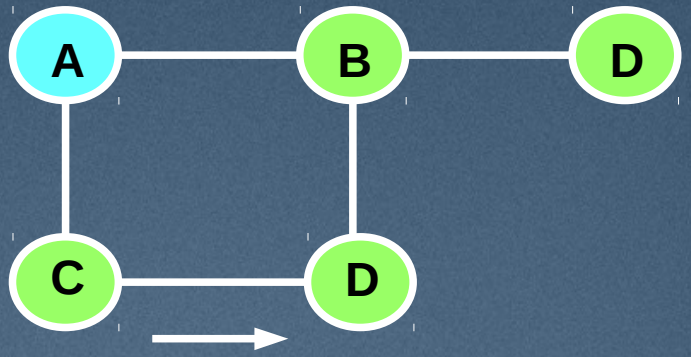
A propaga a B y C la información



B propaga la información



C propaga la información



Flooding

Debido a su forma de trabajar este algoritmo genera una gran cantidad de paquetes duplicados, aunque hay formas de solucionarlo como agregar en el header de cada paquete un contador de saltos.

Flooding

Si bien este algoritmo no es práctico en la mayoría de los casos, tiene algunos casos de uso, como por ejemplo:

- Aplicaciones militares debido a su robustez
- Bases de Datos Distribuidas(Cuando es necesario actualizar todas las bases de datos concurrentemente)

Flooding

- Como Métrica: suele ser utilizado para saber si un algoritmo es bueno o no, ya que Flooding siempre elige el camino mas corto(elige todos los caminos posibles en paralelo)
- Como algoritmo Auxiliar de otros algoritmos de Routing

Link State Routing

Este algoritmo funciona armando un mapa de la red y calculando los caminos óptimos Para esto cada router identifica a sus vecinos y la distancia a la que se encuentra de ellos.

Link State Routing

Para poder hacer eso cada router hace lo siguiente:

- Mantiene un registro de la incidencia de sus enlaces(si esta activo o no, y el costo del enlace)
- Difunde el estado del enlace(para otorgarle a cada router una visión completa del Grafo)
- Corre el Algoritmo de Dijkstra(para calcular los caminos mas cortos y construir la tabla de Forwarding)

Link State Routing

Ejemplos de este tipo de algoritmos son:

- Open Shortest Path First(OSPF)
- Intermediate System - Intermediate System (IS-IS)

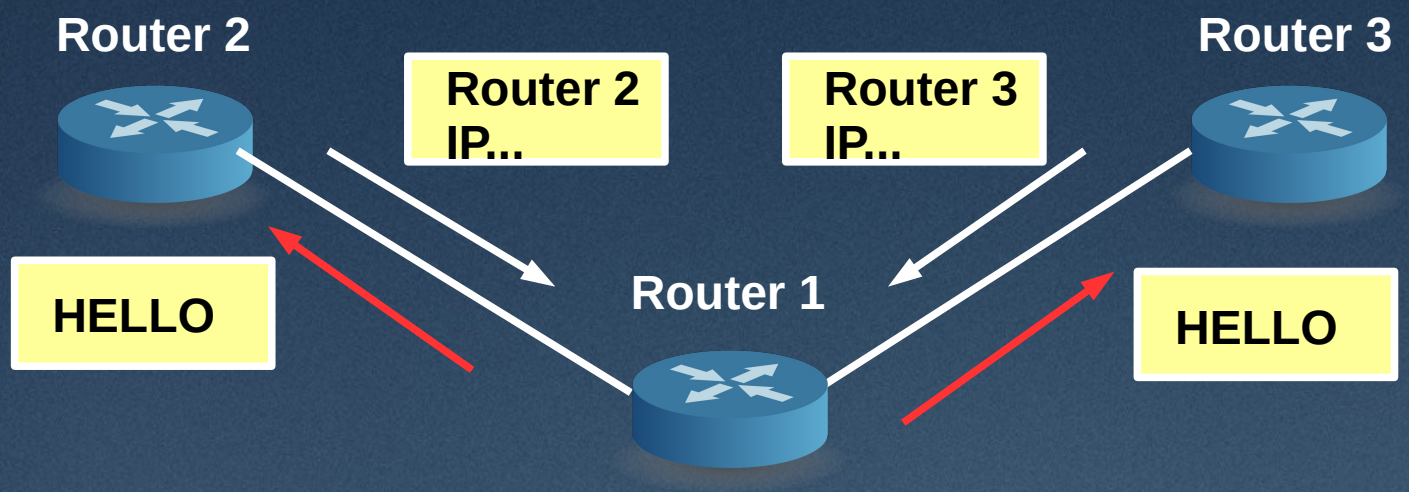
Link State Routing

El accionar de esto algoritmo se puede dividir en 5 pasos:

1. *Descubrir sus vecinos y sus direcciones.*
2. *Medir el costo a cada uno de sus vecinos.*
3. *Construir un paquete con la información recabada.*
4. *Enviar el paquete al resto de routers.*
5. *Calcular la ruta mínima al resto de los routers.*

Link State Routing

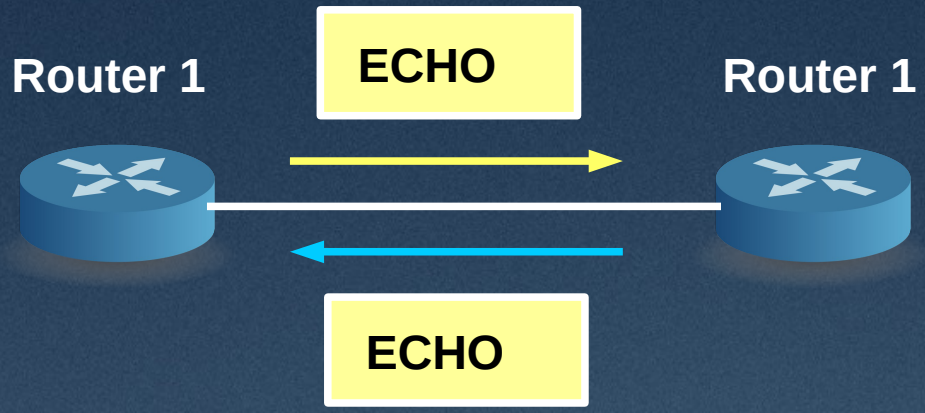
1. Descubrir a sus vecinos y sus direcciones:



Para hacer esto, cada router al activarse envía un paquete especial a todos sus vecinos, para identificarlos.
Cuando un router reciba ese paquete, debe de responder indicando su identidad.

Link State Routing

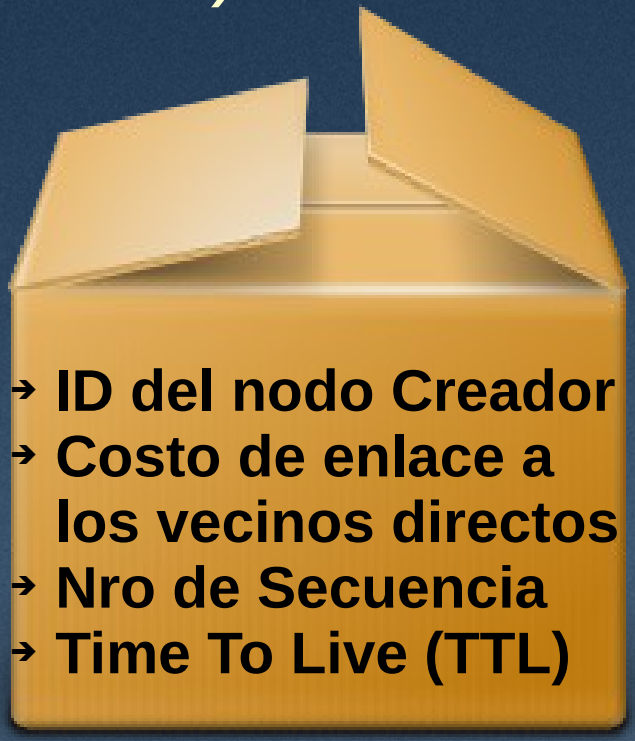
2. Medir el costo a cada uno de sus vecinos:



Para medir el retardo, se enviá un paquete “ECHO” el cual deberá volver a su lugar de origen. Este proceso lo que hará es computar el tiempo de ida y vuelta del paquete, y luego se lo dividirá entre los nodos, y así se generara una aproximación razonable al costo real de la comunicación.

Link State Routing

3. Construir el paquete con la información recabada (LSP- Link State Packtet):



En esta etapa cada Router construye un paquete con todos los datos que informan el estado de la red, dando como resultado un LSP.

Link State Routing

4. Enviar el paquete al resto de routers.

Para hacer esto se realiza el algoritmo Reliable Flooding, el cual hace lo siguiente:

- Almacena el LSP mas reciente de cada nodo.
- Forwardear el LSP a todos los nodos menos al nodo del que recibió el paquete.
- Genera periódicamente un nuevo LSP e incrementa el nro de secuencia.
- Decrementa el TTL de cada LSP almacenado, hasta que llegue a cero cuando finalmente descarta el paquete.

Link State Routing

5. Calcular la ruta mínima al resto de los routers.

Para hacer esto aplica el algoritmo de Dijkstra, para encontrar el camino mas corto.

Link State Routing

También es posible encontrarnos con algunos problemas, como lo pueden ser:

- Que todos los nodos reciban el LSP.
- Que todos los nodos tengan la ultima versión del LSP.

Link State Routing

Aunque estos problemas se pueden solventar con:

- Acknowledgments(manda ack's) y retransmisiones.
- Números de Secuencia.
- Time To Live para cada LSP.

RIP

RIP(Routing Information Protocol) es un protocolo de ruteo IGP, basado en el Distance Vector Protocol.
Básicamente hace que cada router envié un *vector de distancia* cada 30seg, o que cambie por *Triggered Update*. Ademas limita el numero de saltos a 15, siendo 16 el infinito o inalcanzable.
Este protocolo esta limitado a redes bastantes pequeñas.

RIP

Entre sus características se encuentra el soporte de múltiples de familias de direcciones.

RIP

Ademas de esto, RIP cuenta con dos versiones:

- RIPv1 (Versión original)
- RIPv2(Versión actual)

RIP

RIPv1:

Definida en la RFC1058, en donde se define a RIP como un protocolo de ruteo con clase, basado en las clases de las direcciones IP. Esto hace que no soporten ni mascaras de tamaño variable (VLSM), ni direccionamiento sin clase (CIDR), imponiendo así que la mascara de red este predefinida para su clase IP, volviéndolo ineficiente.

RIP

RIPv2:

Debido a las limitaciones de RIPv1, en 1993 se desarrollo esta nueva versión que da soporte para CIDR y VLSM. Ademas mantiene el limite de 15 saltos por retrocompatibilidad con la versión anterior. Ademas de esto RIPv2 soporta autenticación, por esto mismo para no perder la retrocompatibilidad se le añadió un interruptor de compatibilidad.

RIP

RIP funciona de la siguiente forma:

- Al iniciarse envía un mensaje a cada uno de sus vecinos, pidiendo una copia de su tabla de ruteo.
- Cuando entra en modo activo, envía parte o la totalidad de su tabla de ruteo por boradcast o punto a punto cada 30 seg a sus vecinos.
- En caso de que una métrica haya cambiado, RIP se lo comunica a los routers vecinos.

Conclusiones