

Redes de datos 1

Capa de red

Facultad de Ingeniería – Universidad de la República

Instituto de Ingeniería Eléctrica

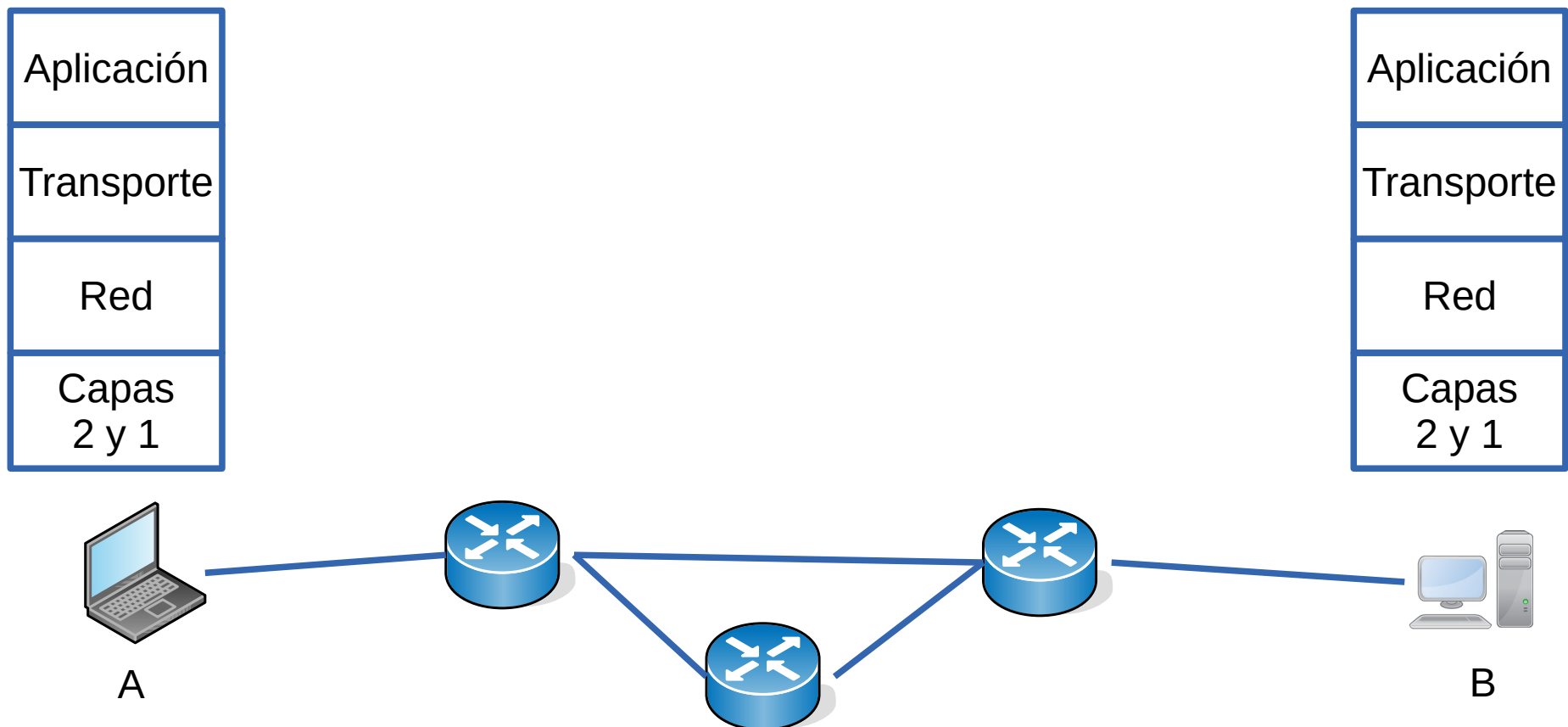
2023

Agenda

- Conceptos de capa de red
 - Objetivos de la capa de red
 - Plano de control y plano de datos
 - Arquitectura tradicional y definida por software (SDN)
 - ¿Cómo es y qué hace un enrutador o router?
- Plano de datos
- Plano de control
- Redes de circuitos virtuales

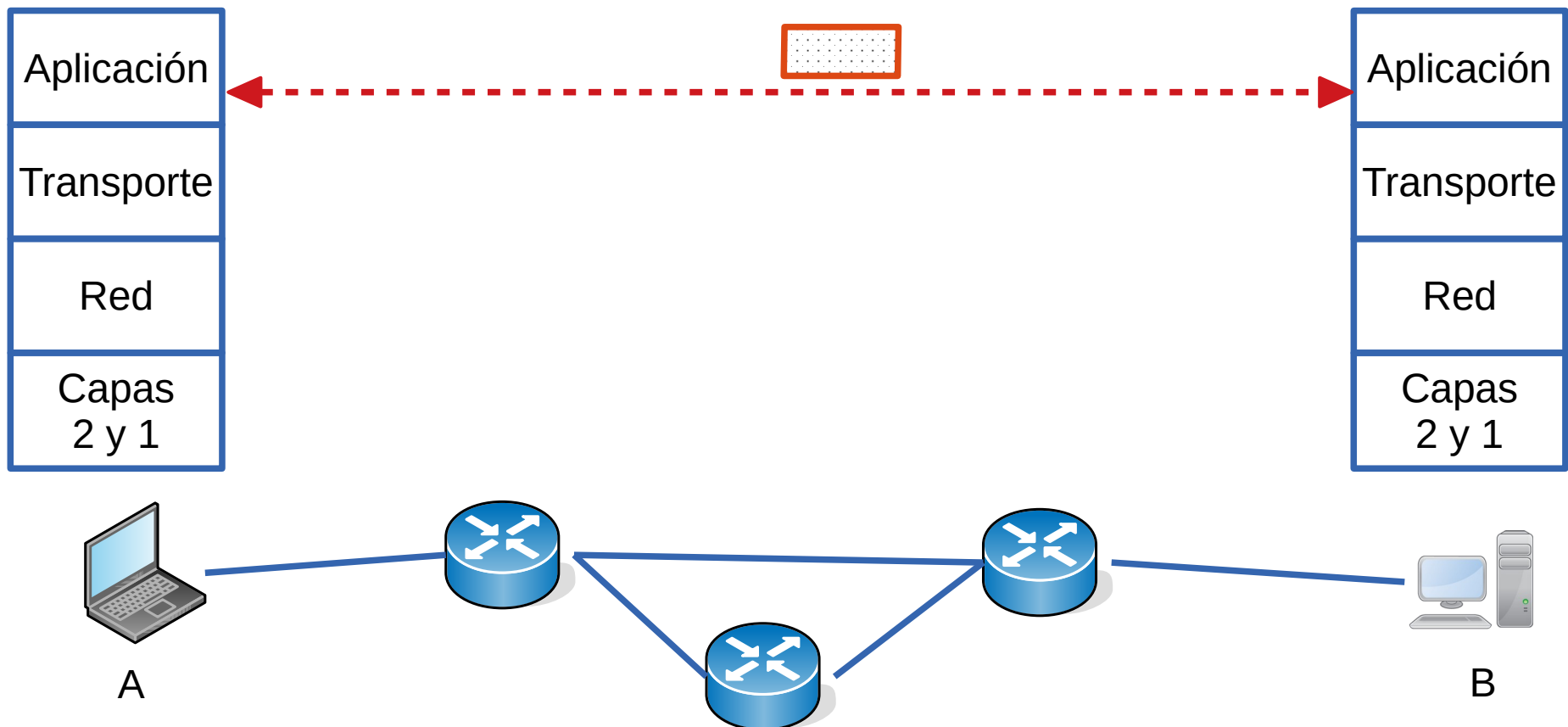
Relación con capa de transporte

- Las entidades de capa de transporte ven un intercambio directo entre los dispositivos en los extremos de la red (A y B)
- Pero ... los segmentos se envían usando los servicios de la capa de red
- La capa de red tiene que encontrar un camino en base a los recursos disponibles en la red (enlaces y nodos) y encaminar los paquetes al destino



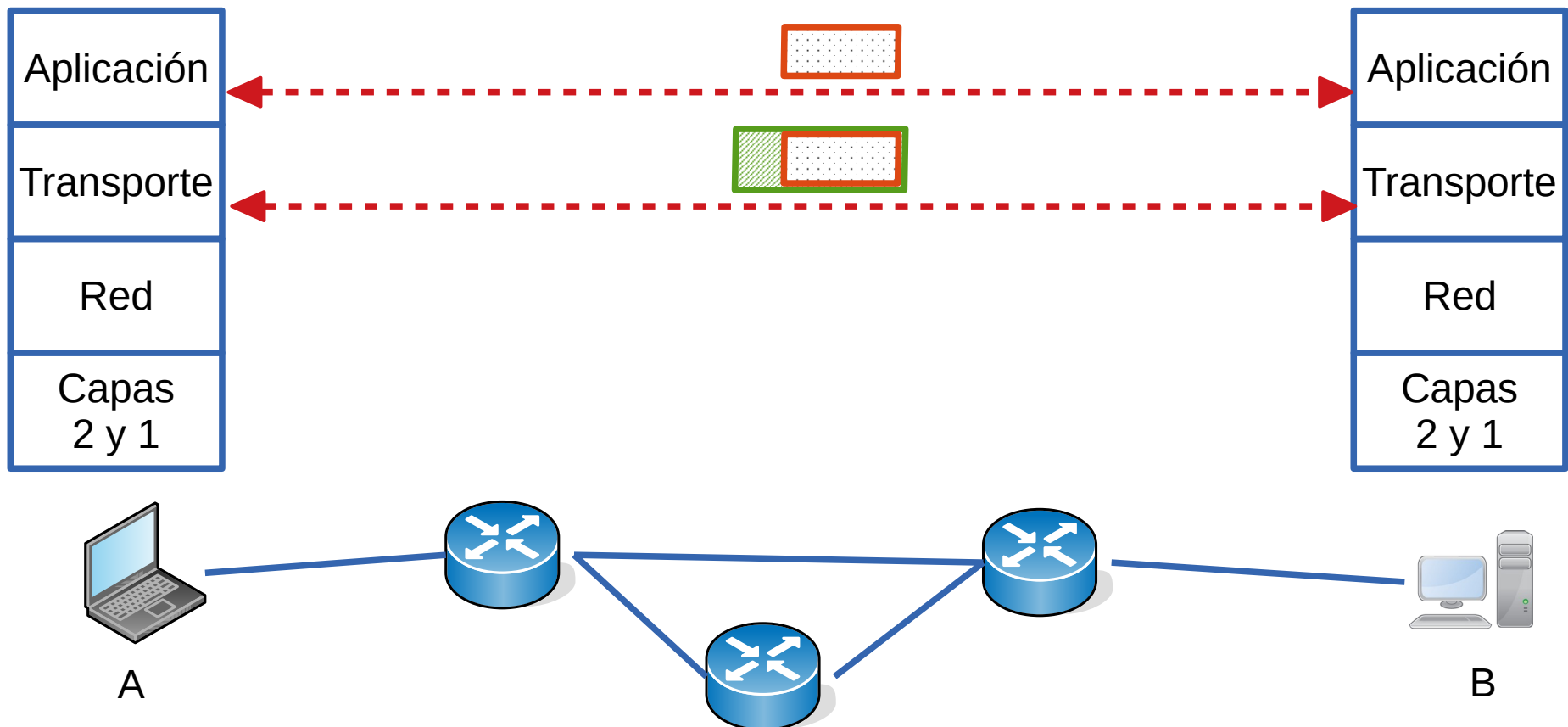
Relación con capa de transporte

- Las entidades de capa de transporte ven un intercambio directo entre los dispositivos en los extremos de la red (A y B)
- Pero ... los segmentos se envían usando los servicios de la capa de red
- La capa de red tiene que encontrar un camino en base a los recursos disponibles en la red (enlaces y nodos) y encaminar los paquetes al destino



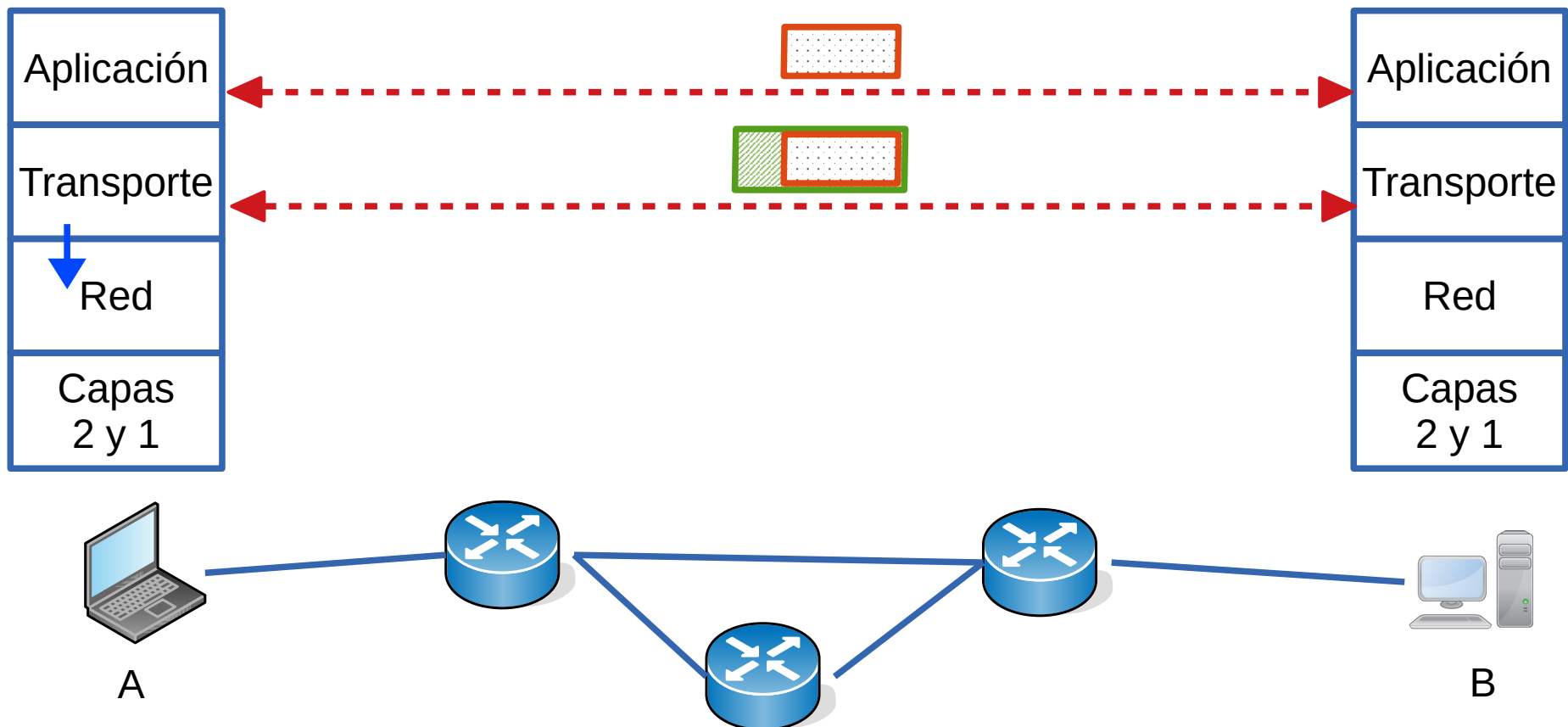
Relación con capa de transporte

- Las entidades de capa de transporte ven un intercambio directo entre los dispositivos en los extremos de la red (A y B)
- Pero ... los segmentos se envían usando los servicios de la capa de red
- La capa de red tiene que encontrar un camino en base a los recursos disponibles en la red (enlaces y nodos) y encaminar los paquetes al destino



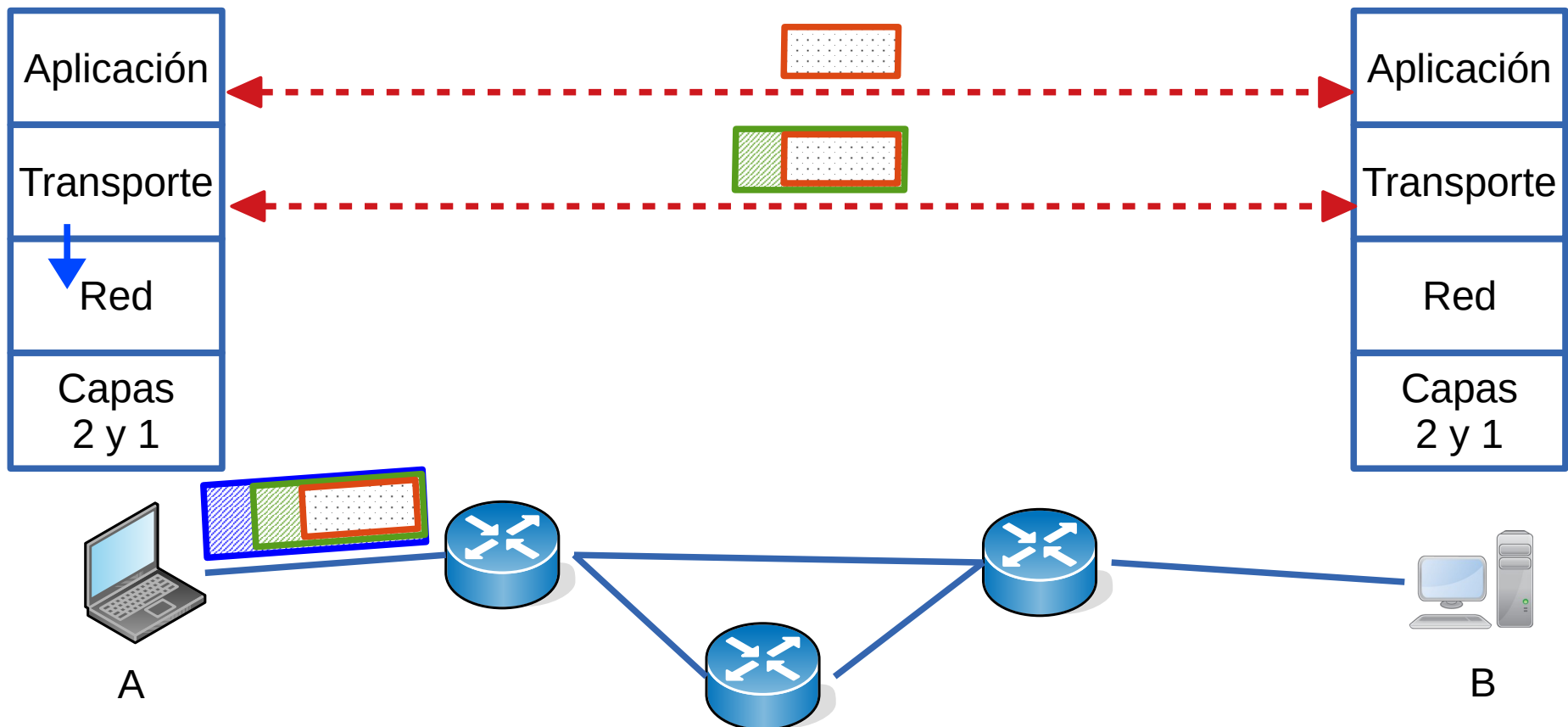
Relación con capa de transporte

- Las entidades de capa de transporte ven un intercambio directo entre los dispositivos en los extremos de la red (A y B)
- Pero ... los segmentos se envían usando los servicios de la capa de red
- La capa de red tiene que encontrar un camino en base a los recursos disponibles en la red (enlaces y nodos) y encaminar los paquetes al destino



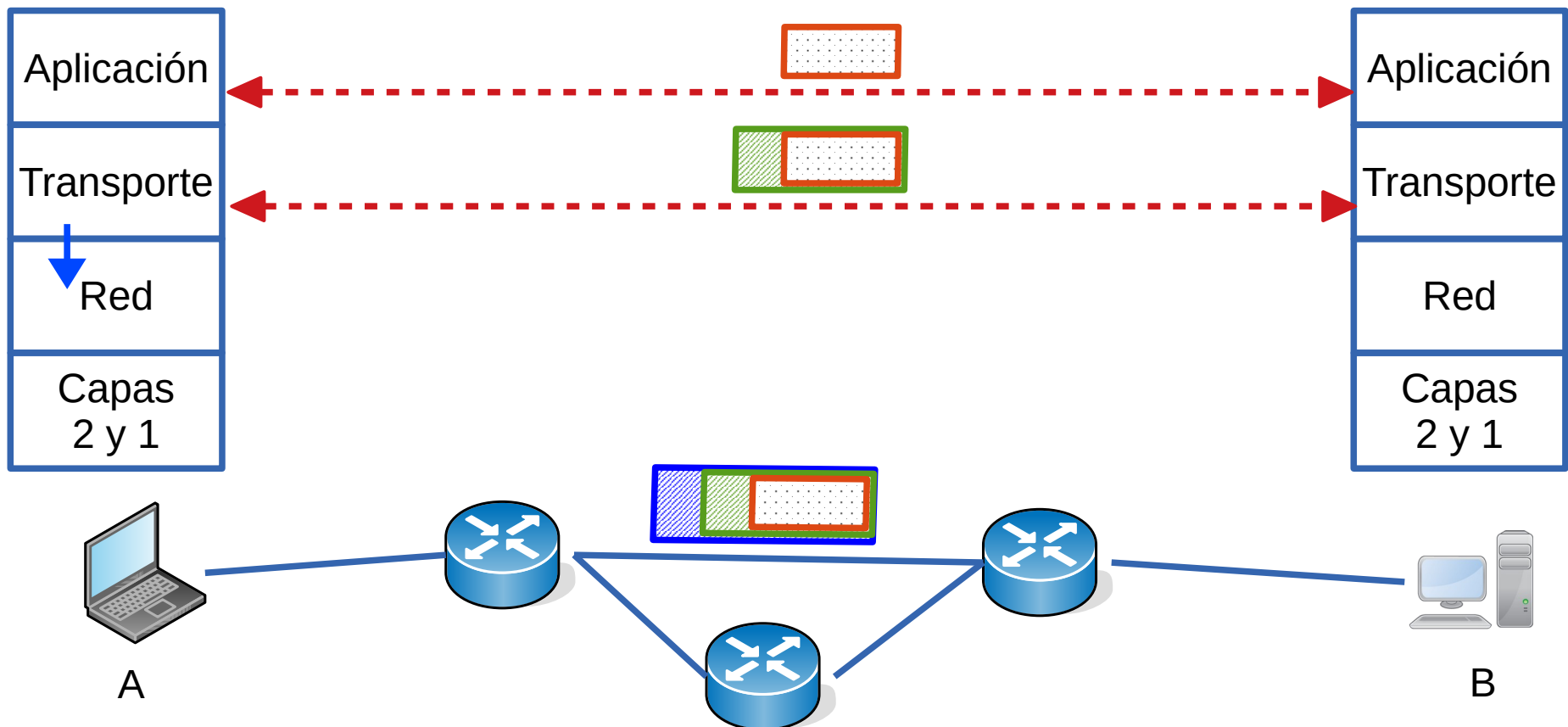
Relación con capa de transporte

- Las entidades de capa de transporte ven un intercambio directo entre los dispositivos en los extremos de la red (A y B)
- Pero ... los segmentos se envían usando los servicios de la capa de red
- La capa de red tiene que encontrar un camino en base a los recursos disponibles en la red (enlaces y nodos) y encaminar los paquetes al destino



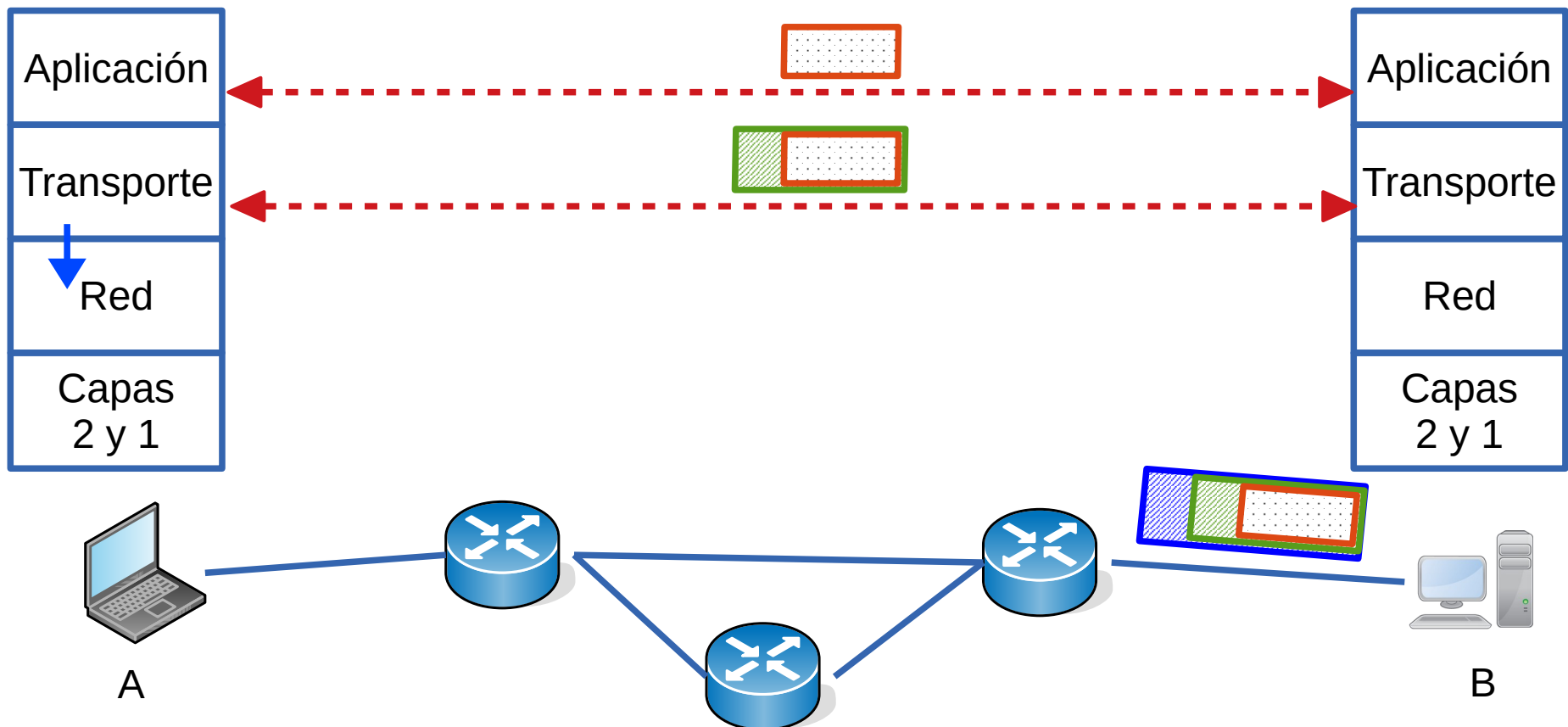
Relación con capa de transporte

- Las entidades de capa de transporte ven un intercambio directo entre los dispositivos en los extremos de la red (A y B)
- Pero ... los segmentos se envían usando los servicios de la capa de red
- La capa de red tiene que encontrar un camino en base a los recursos disponibles en la red (enlaces y nodos) y encaminar los paquetes al destino



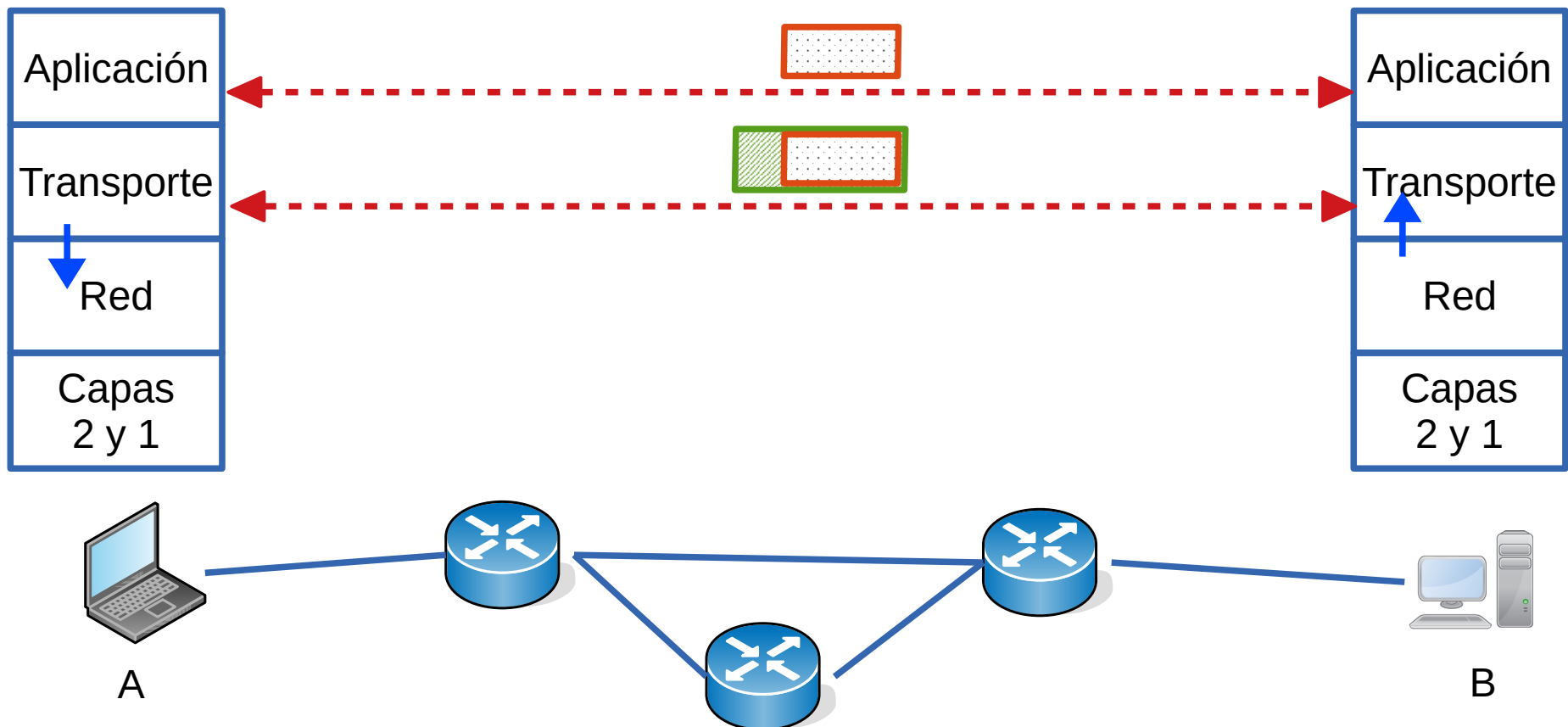
Relación con capa de transporte

- Las entidades de capa de transporte ven un intercambio directo entre los dispositivos en los extremos de la red (A y B)
- Pero ... los segmentos se envían usando los servicios de la capa de red
- La capa de red tiene que encontrar un camino en base a los recursos disponibles en la red (enlaces y nodos) y encaminar los paquetes al destino



Relación con capa de transporte

- Las entidades de capa de transporte ven un intercambio directo entre los dispositivos en los extremos de la red (A y B)
- Pero ... los segmentos se envían usando los servicios de la capa de red
- La capa de red tiene que encontrar un camino en base a los recursos disponibles en la red (enlaces y nodos) y encaminar los paquetes al destino

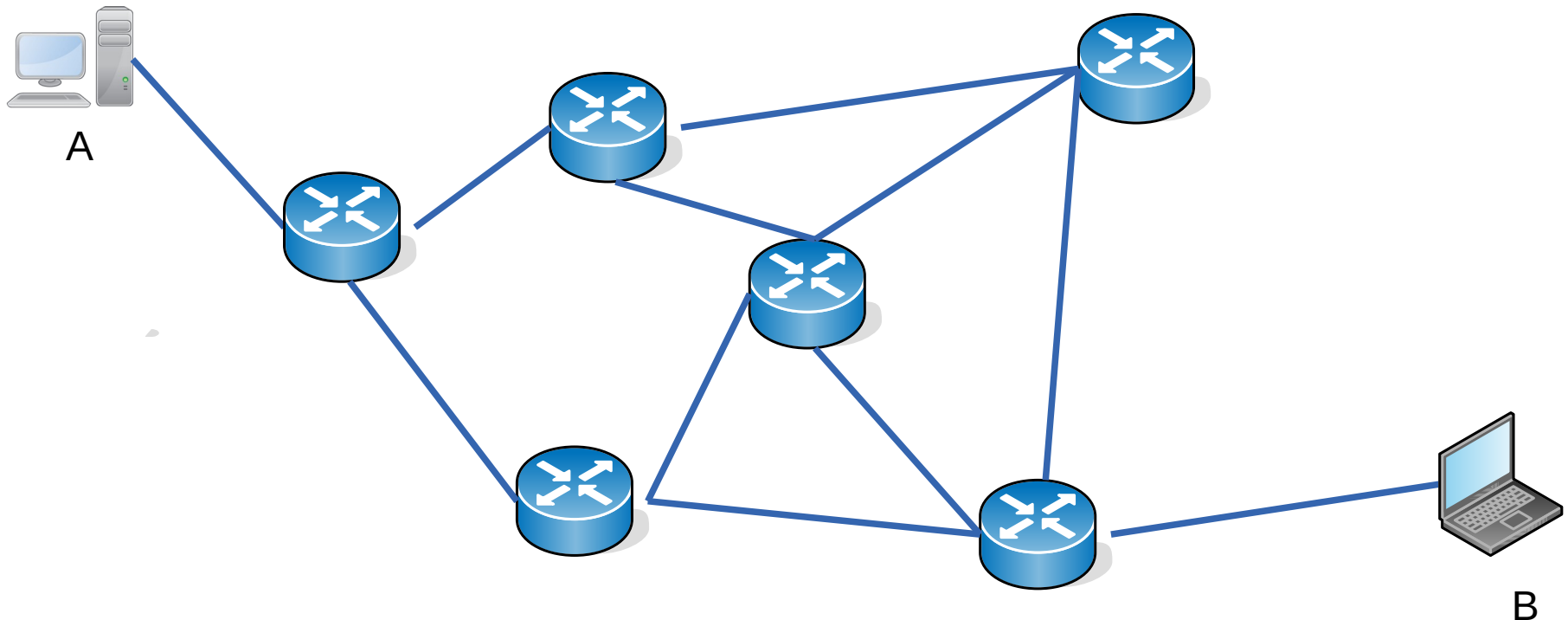


Capa de Red

- **Objetivo** principal:
 - Hacer llegar los paquetes desde A hasta B
- Se necesita identificar y localizar a los equipos en la red: **direcciones**
- Se identifican dos planos:
 - **Plano de Control:**
 - Determina los (mejores) caminos a seguir por los paquetes
 - Función de **ruteo** (**routing**)
 - **Plano de Datos:**
 - Implementa el encaminamiento en cada enrutador
 - Función de **encaminamiento** (**forwarding**)

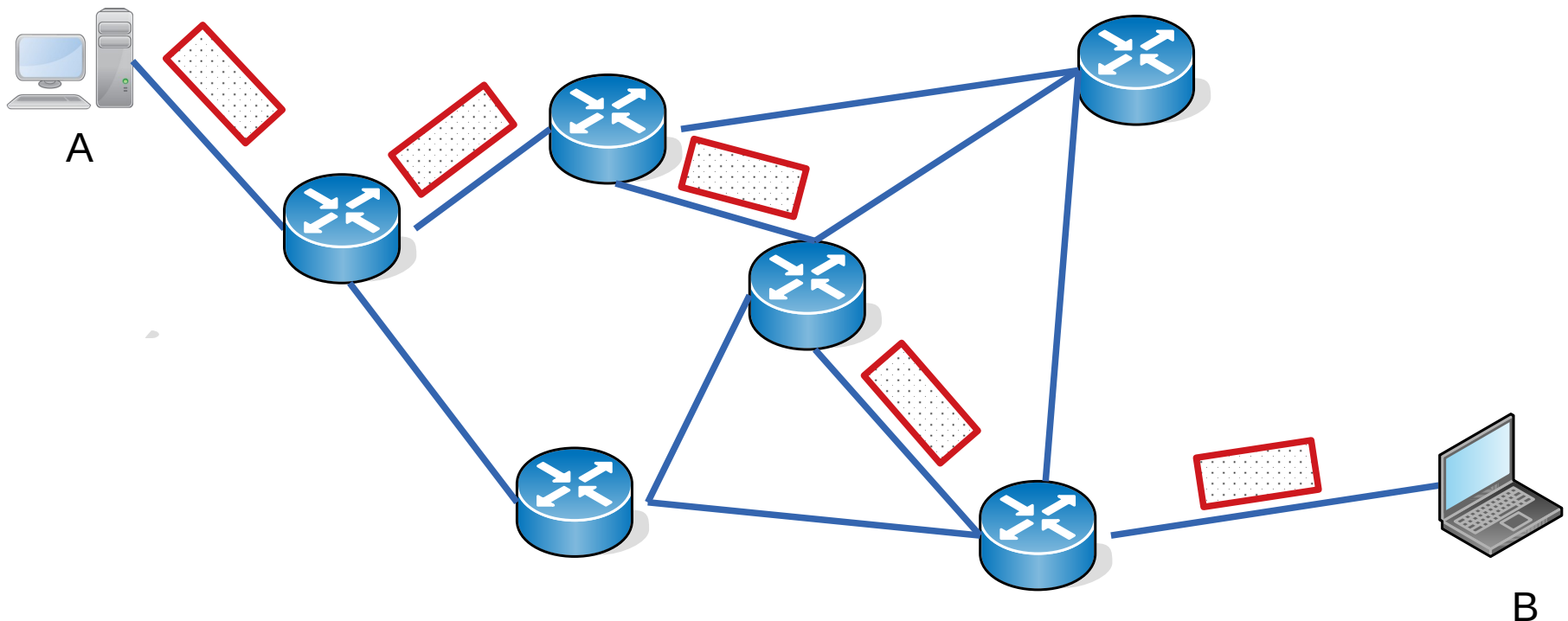
Plano de Control: función de ruteo (routing)

- Encontrar el **mejor camino** para ir de A a B
- El ruteo (routing) comprende las **decisiones** que determinan los **caminos** que deben seguir los paquetes desde un origen hacia un destino
- Se usan **algoritmos de ruteo** para determinar los mejores caminos
- Escala de tiempo: **segundos**



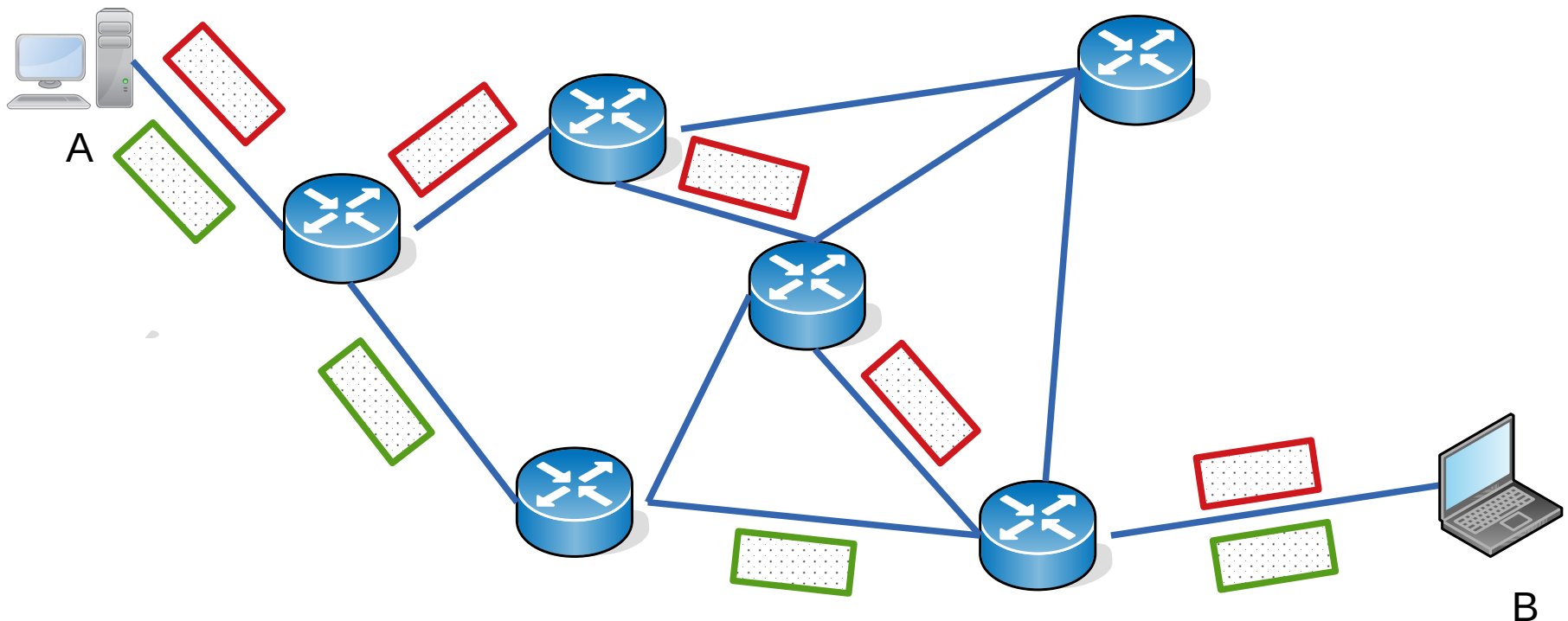
Plano de Control: función de ruteo (routing)

- Encontrar el **mejor camino** para ir de A a B
- El ruteo (routing) comprende las **decisiones** que determinan los **caminos** que deben seguir los paquetes desde un origen hacia un destino
- Se usan **algoritmos de ruteo** para determinar los mejores caminos
- Escala de tiempo: **segundos**



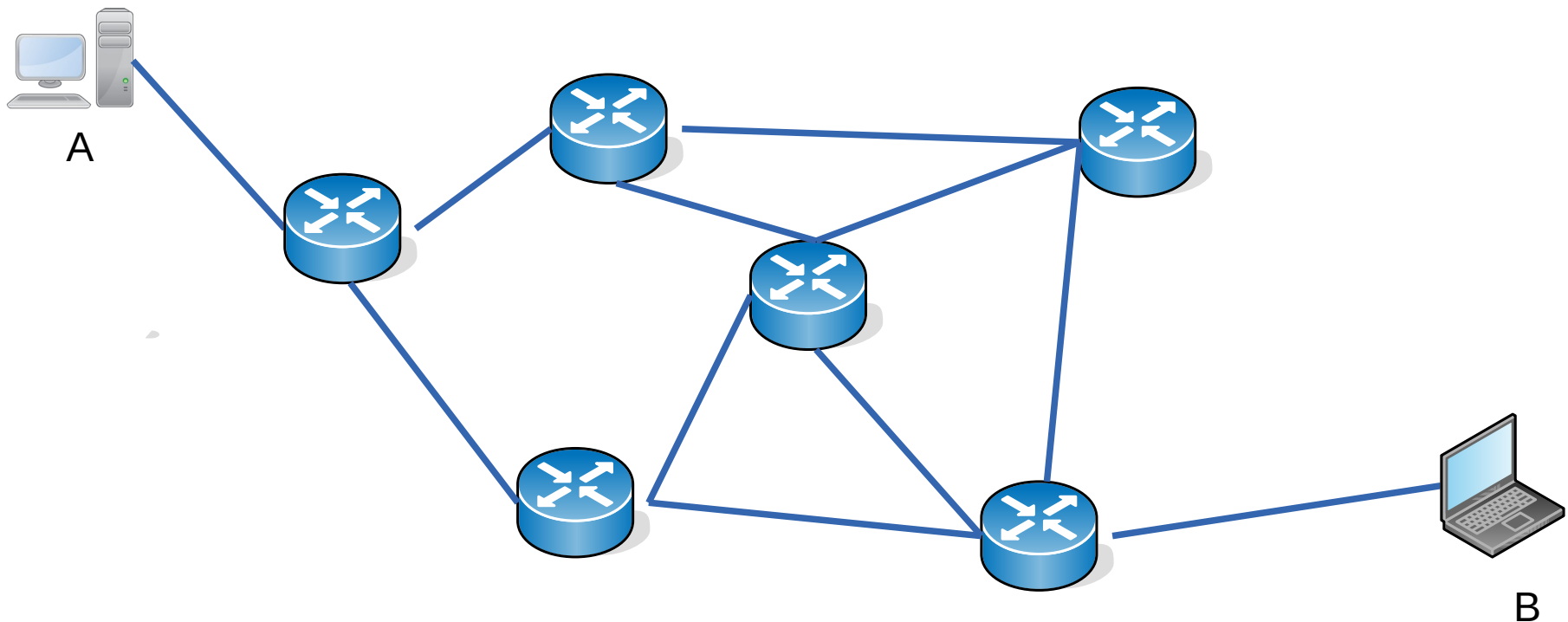
Plano de Control: función de ruteo (routing)

- Encontrar el **mejor camino** para ir de A a B
- El ruteo (routing) comprende las **decisiones** que determinan los **caminos** que deben seguir los paquetes desde un origen hacia un destino
- Se usan **algoritmos de ruteo** para determinar los mejores caminos
- Escala de tiempo: **segundos**



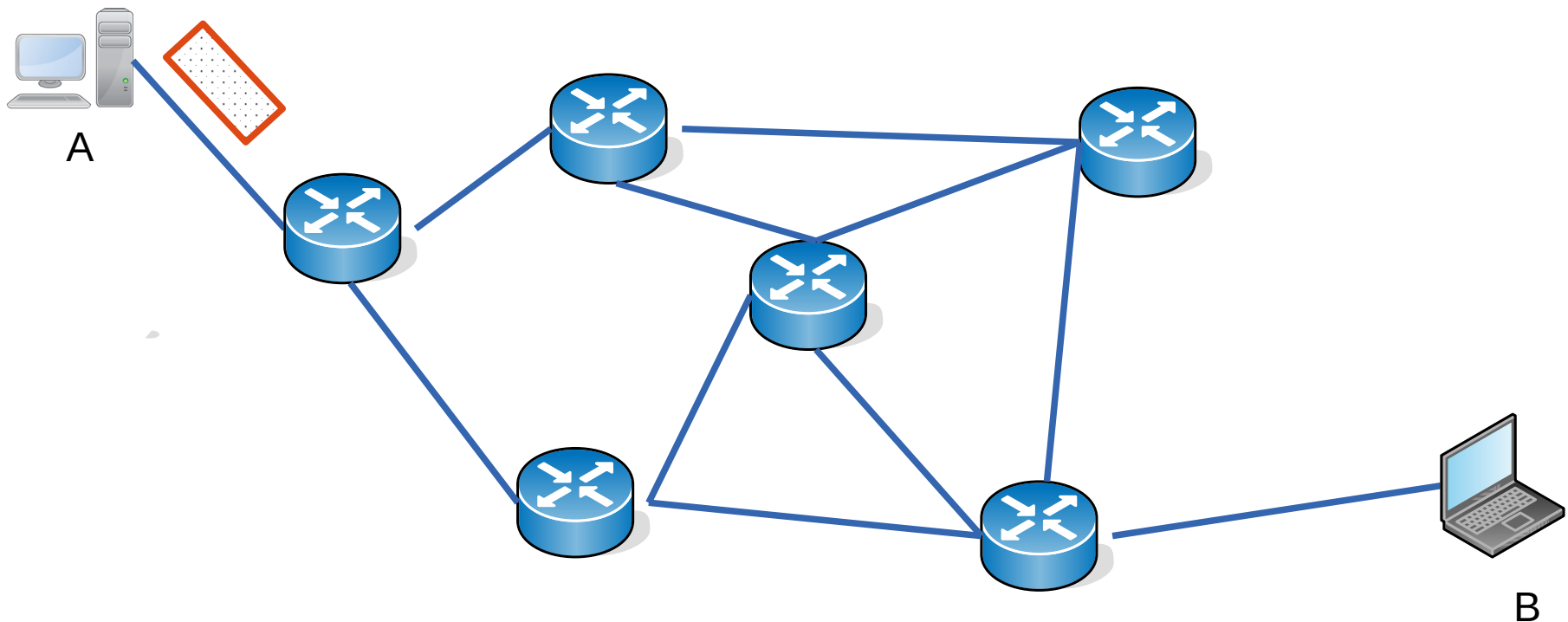
Plano de datos: encaminamiento (forwarding)

- Cuando un enrutador recibe un paquete por una de sus líneas de entrada o interfaces, debe consultar una tabla para determinar cuál es el próximo enrutador más adecuado para llegar a ese destino y entonces encaminar el paquete hacia allí
- Esas tablas se llaman **tablas de forwarding**
- Escala de tiempo: **nanosegundos**



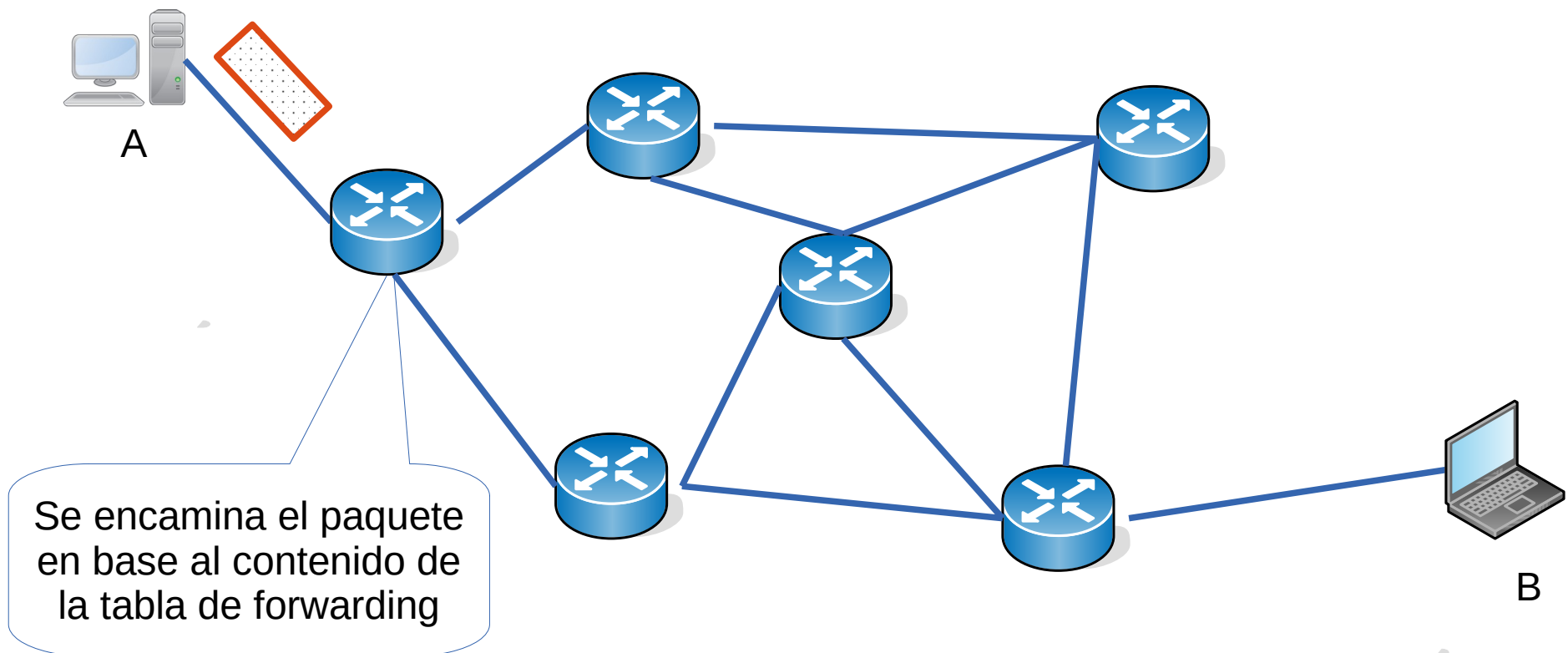
Plano de datos: encaminamiento (forwarding)

- Cuando un enrutador recibe un paquete por una de sus líneas de entrada o interfaces, debe consultar una tabla para determinar cuál es el próximo enrutador más adecuado para llegar a ese destino y entonces encaminar el paquete hacia allí
- Esas tablas se llaman **tablas de forwarding**
- Escala de tiempo: **nanosegundos**



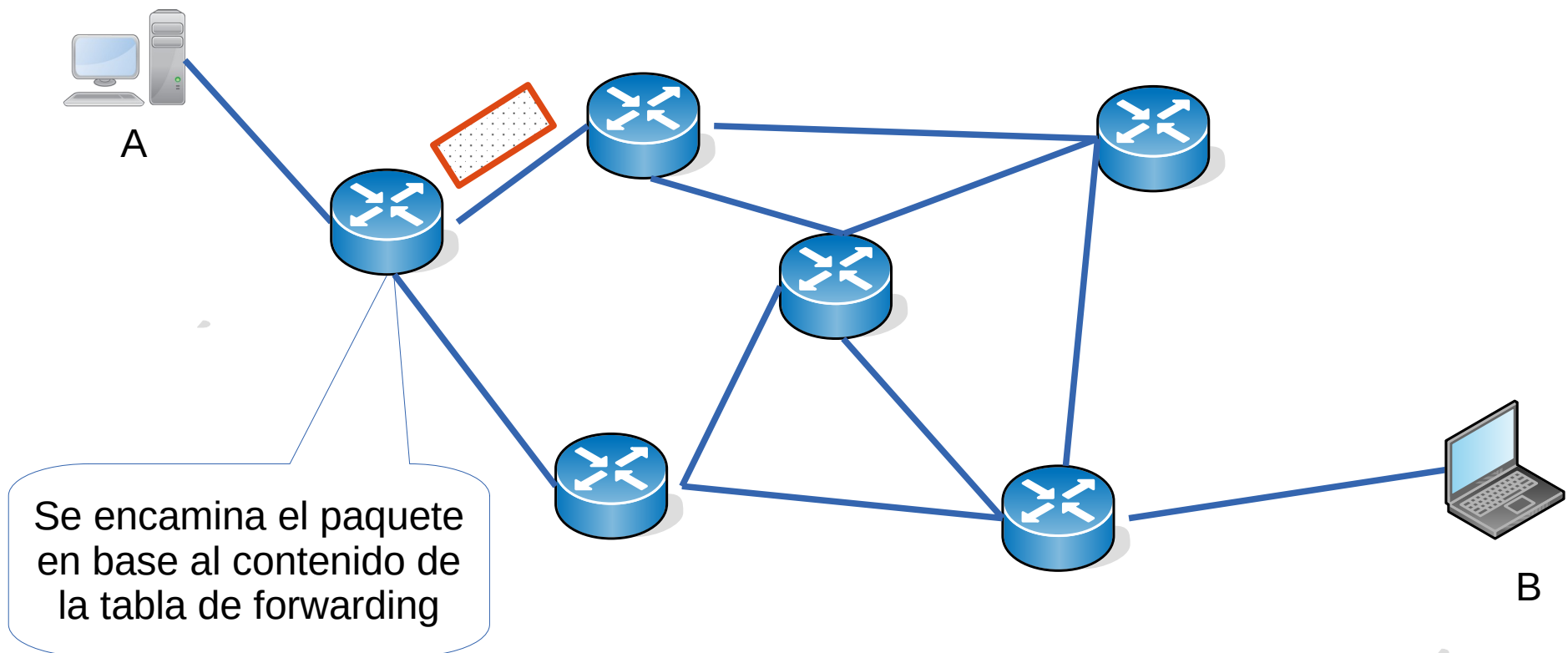
Plano de datos: encaminamiento (forwarding)

- Cuando un enrutador recibe un paquete por una de sus líneas de entrada o interfaces, debe consultar una tabla para determinar cuál es el próximo enrutador más adecuado para llegar a ese destino y entonces encaminar el paquete hacia allí
- Esas tablas se llaman **tablas de forwarding**
- Escala de tiempo: **nanosegundos**



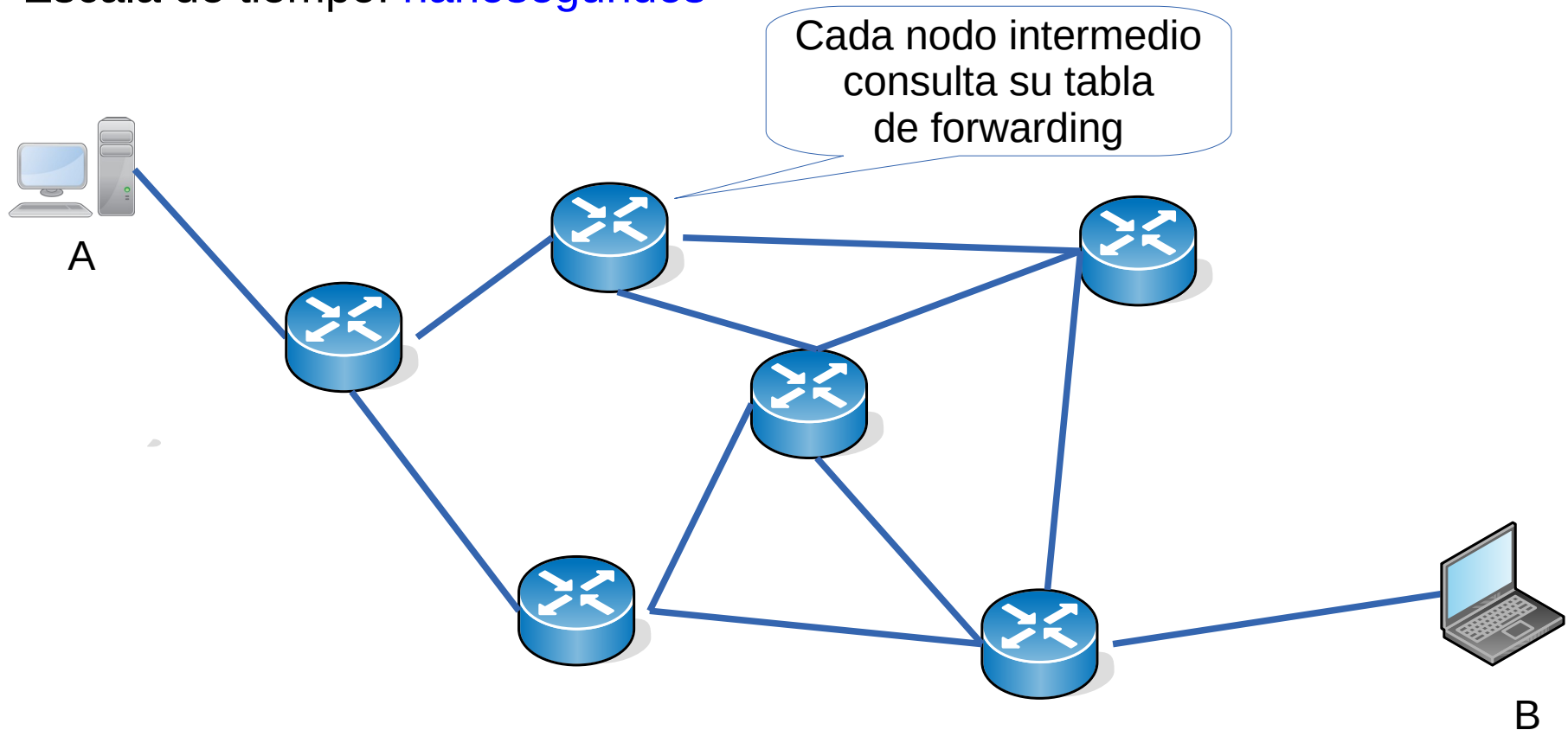
Plano de datos: encaminamiento (forwarding)

- Cuando un enrutador recibe un paquete por una de sus líneas de entrada o interfaces, debe consultar una tabla para determinar cuál es el próximo enrutador más adecuado para llegar a ese destino y entonces encaminar el paquete hacia allí
- Esas tablas se llaman **tablas de forwarding**
- Escala de tiempo: **nanosegundos**



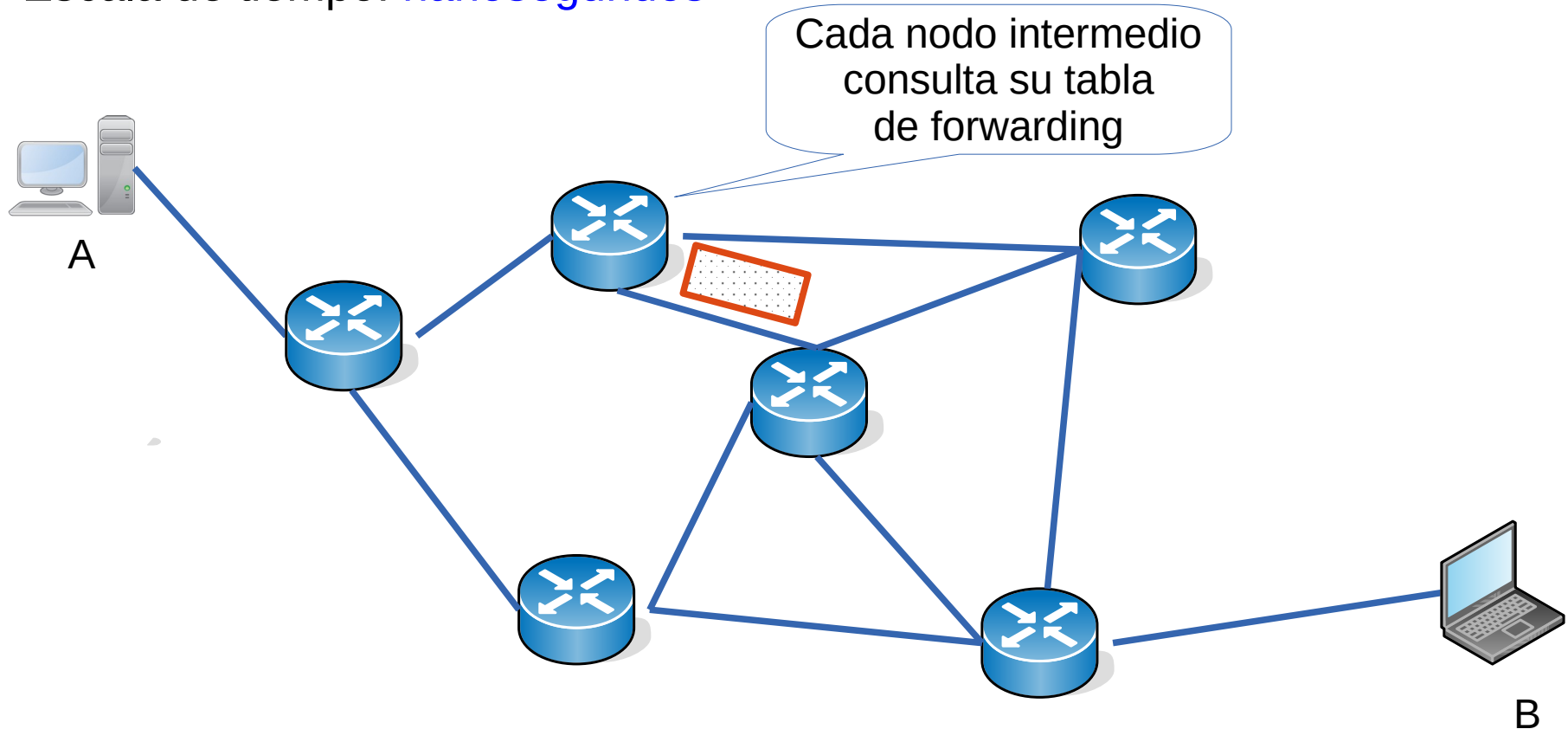
Plano de datos: encaminamiento (forwarding)

- Cuando un enrutador recibe un paquete por una de sus líneas de entrada o interfaces, debe consultar una tabla para determinar cuál es el próximo enrutador más adecuado para llegar a ese destino y entonces encaminar el paquete hacia allí
- Esas tablas se llaman **tablas de forwarding**
- Escala de tiempo: **nanosegundos**



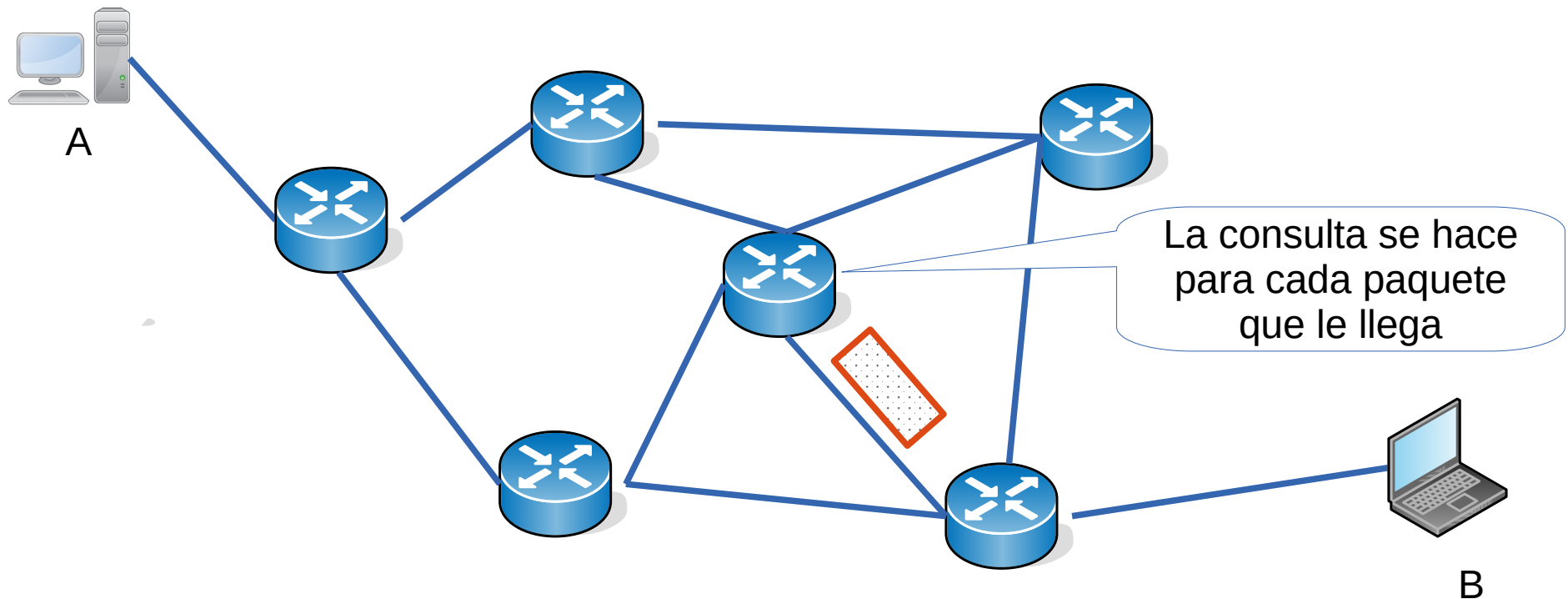
Plano de datos: encaminamiento (forwarding)

- Cuando un enrutador recibe un paquete por una de sus líneas de entrada o interfaces, debe consultar una tabla para determinar cuál es el próximo enrutador más adecuado para llegar a ese destino y entonces encaminar el paquete hacia allí
- Esas tablas se llaman **tablas de forwarding**
- Escala de tiempo: **nanosegundos**



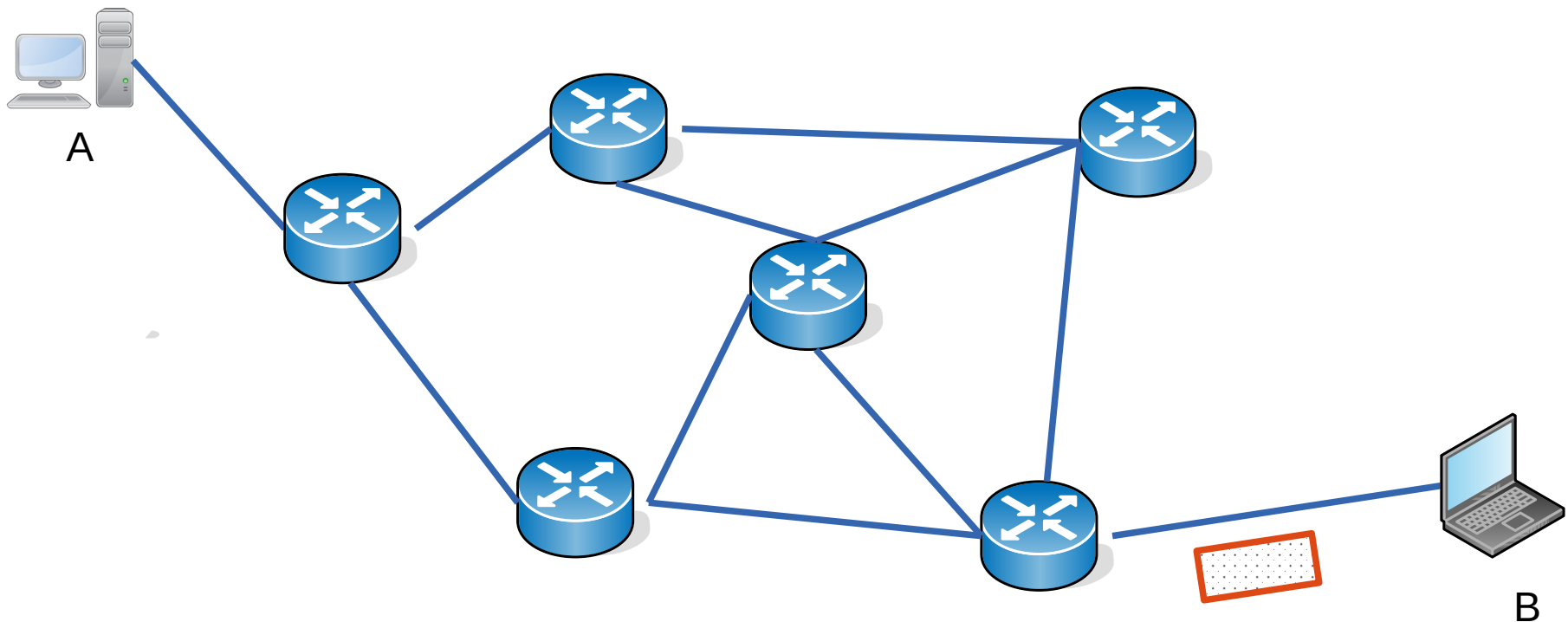
Plano de datos: encaminamiento (forwarding)

- Cuando un enrutador recibe un paquete por una de sus líneas de entrada o interfaces, debe consultar una tabla para determinar cuál es el próximo enrutador más adecuado para llegar a ese destino y entonces encaminar el paquete hacia allí
- Esas tablas se llaman **tablas de forwarding**
- Escala de tiempo: **nanosegundos**



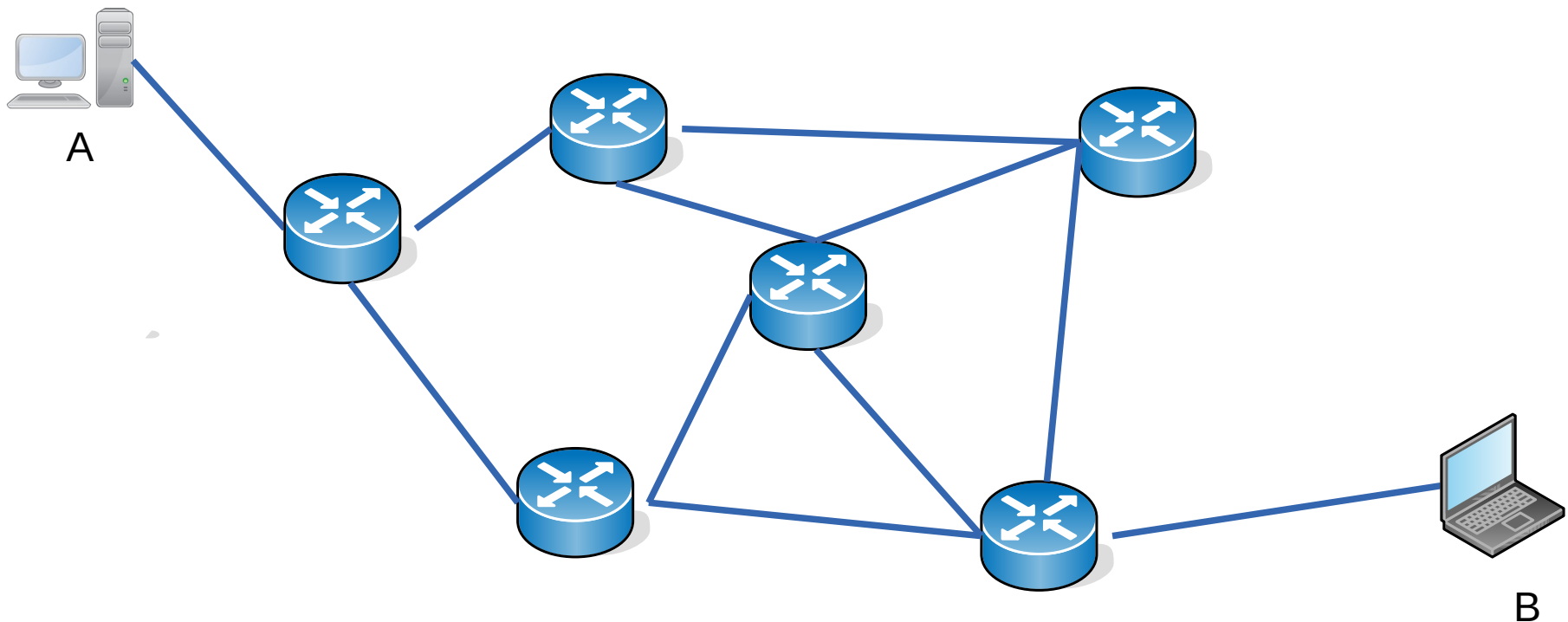
Plano de datos: encaminamiento (forwarding)

- Cuando un enrutador recibe un paquete por una de sus líneas de entrada o interfaces, debe consultar una tabla para determinar cuál es el próximo enrutador más adecuado para llegar a ese destino y entonces encaminar el paquete hacia allí
- Esas tablas se llaman **tablas de forwarding**
- Escala de tiempo: **nanosegundos**



Plano de datos: encaminamiento (forwarding)

- Cuando un enrutador recibe un paquete por una de sus líneas de entrada o interfaces, debe consultar una tabla para determinar cuál es el próximo enrutador más adecuado para llegar a ese destino y entonces encaminar el paquete hacia allí
- Esas tablas se llaman **tablas de forwarding**
- Escala de tiempo: **nanosegundos**

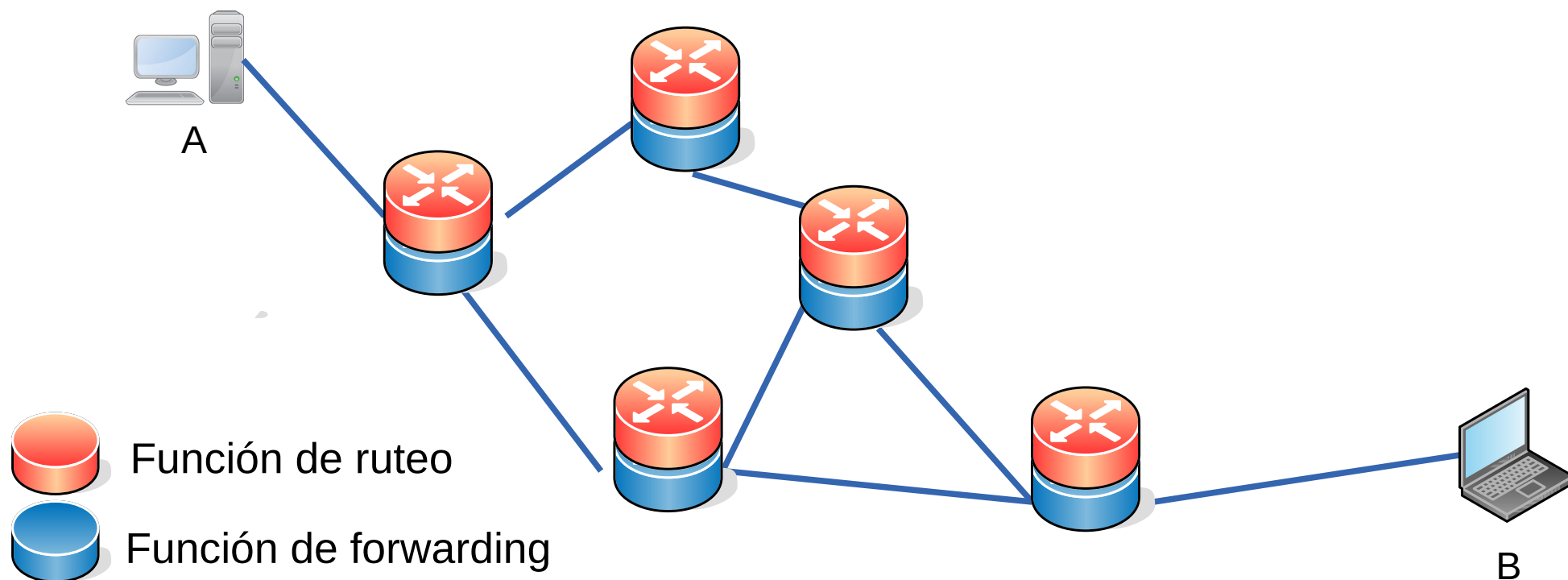


Relación entre ruteo y forwarding

- Las decisiones de ruteo (elección del camino), se plasman en el contenido de las tablas de forwarding de los equipos involucrados en el camino
- Muy sintéticamente:
 - La función de ruteo elige un camino y determina el contenido de la tabla de forwarding de cada enrutador para que los paquetes sigan el camino elegido (“escribe” las tablas)
 - En cada enrutador la función de forwarding usa esa tabla para encaminar los paquetes (“lee” la tabla)
- Una vez elegido el camino, los cambios en las tablas de forwarding pueden hacerse de forma **manual o dinámica**
- Si se hace de forma manual, cuando hay cambios hay que recorrer los equipos involucrados y configurar las nuevas tablas
- Si se hace por algún mecanismo dinámico, los cambios en el ruteo se reflejan automáticamente en las tablas de forwarding

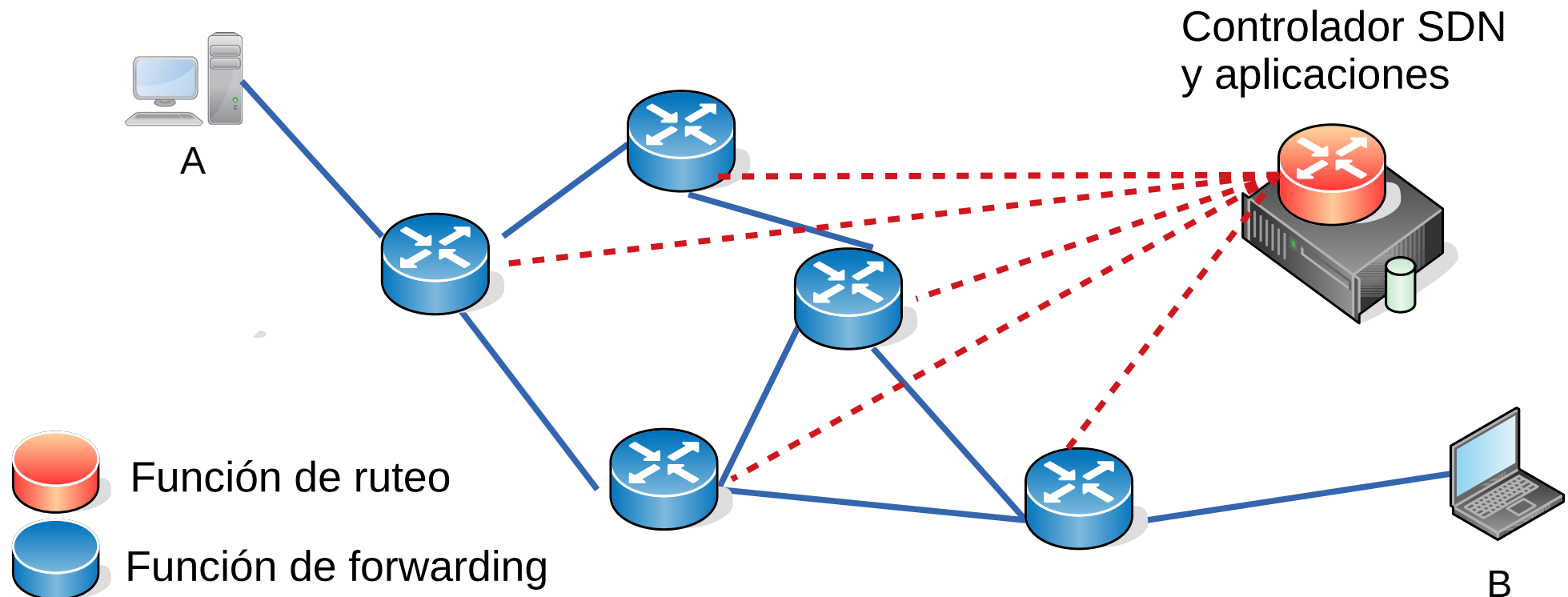
Relación entre ruteo y forwarding

- En la **arquitectura tradicional** los nodos o enrutadores pueden implementar ambas funciones: ruteo y forwarding. Por tanto, los enrutadores:
 - Intercambian información entre ellos mediante **protocolos de ruteo**
 - Con la información obtenida aplican **algoritmos de ruteo** y deciden los mejores caminos
 - Esos caminos se plasman en el **contenido de su tabla de forwarding**
 - **Encaminan los paquetes** en base a esa tabla



Relación entre ruteo y forwarding

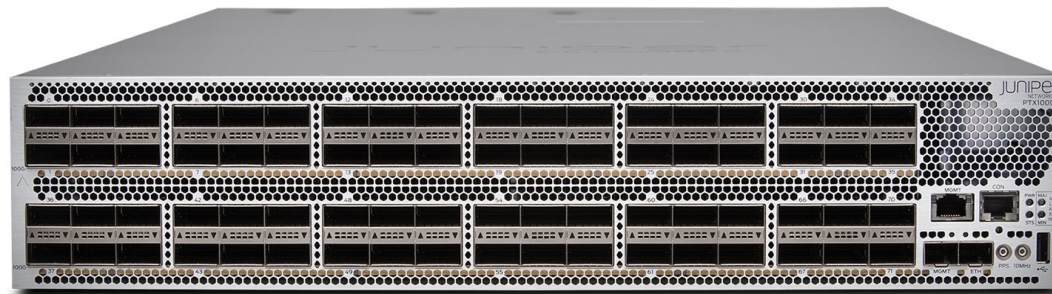
- La **arquitectura SDN** (Software Defined Networking) plantea la separación del plano de datos y de control en diferentes equipos
- **Plano de Datos:** conmutadores especializados en el forwarding de paquetes a alta velocidad
- **Plano de Control:** equipo de propósito general con un **controlador SDN y aplicaciones** (software) que implementan las decisiones (función de ruteo en particular) y distribuye las tablas de forwarding a los conmutadores



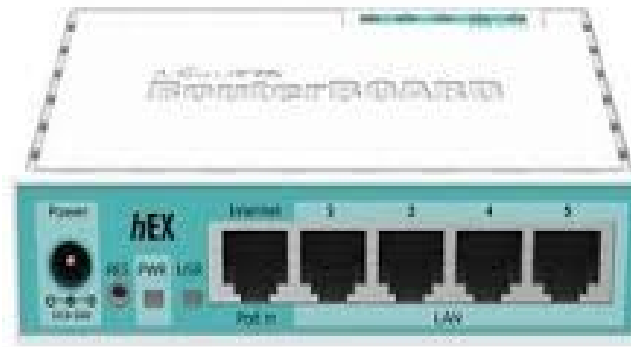
Arquitectura tradicional vs SDN

- La **arquitectura actual** de las redes presenta debilidades frente a la proliferación de nuevos servicios y la necesidad de agilidad en sus despliegues
 - Es estática y compleja
 - Dificultad para escalar
 - Dependencia de los fabricantes de equipos
- La **arquitectura SDN** permite:
 - Que el plano de datos se implemente con dispositivos optimizados para el forwarding a altas velocidades
 - Implementar cambios en el plano de control de forma más ágil (software)
 - nuevos protocolos, nuevas funcionalidades
 - El controlador SDN se entiende como un sistema operativo de red sobre el que se pueden desarrollar aplicaciones específicas:
 - Algoritmos de ruteo, Firewalls, etc
- También la arquitectura SDN implica un modelo centralizado que puede afectar la confiabilidad de la red... ¿qué pasa si se cae el controlador?

¿Como se ve un enrutador por fuera?

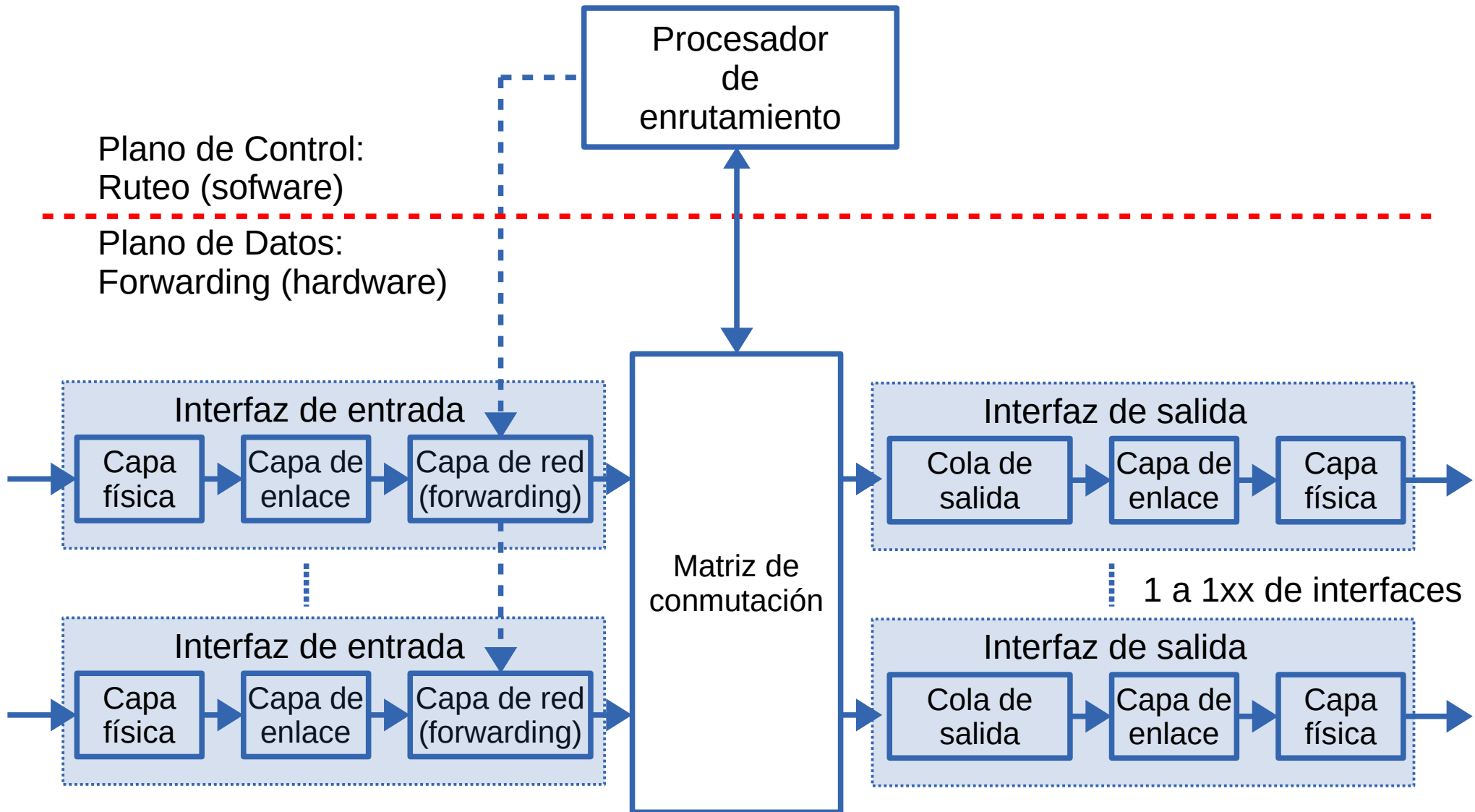


¿Como se ve un enrutador por fuera?



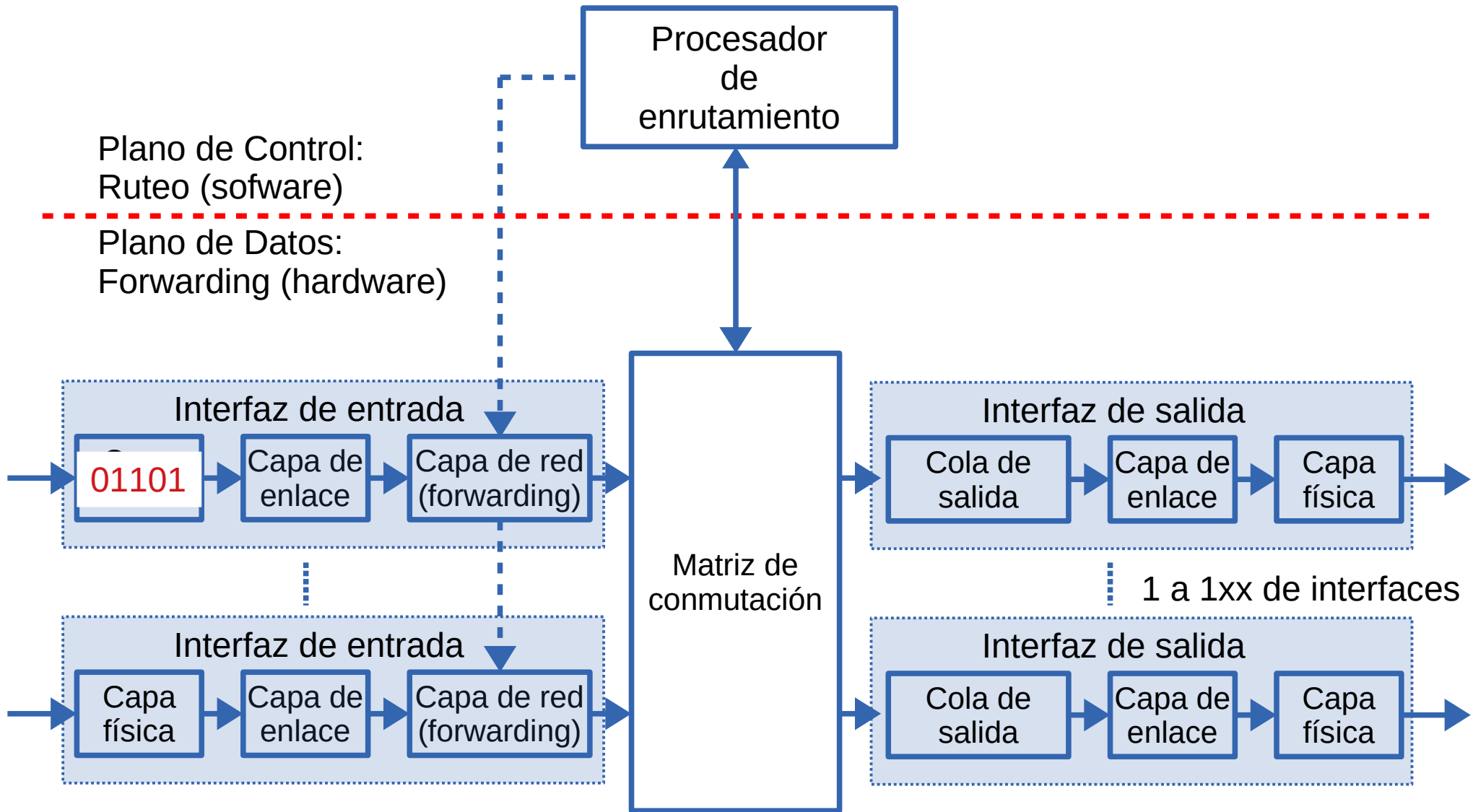
¿Cómo funciona un enrutador?

- Vista de alto nivel de un enrutador o router genérico



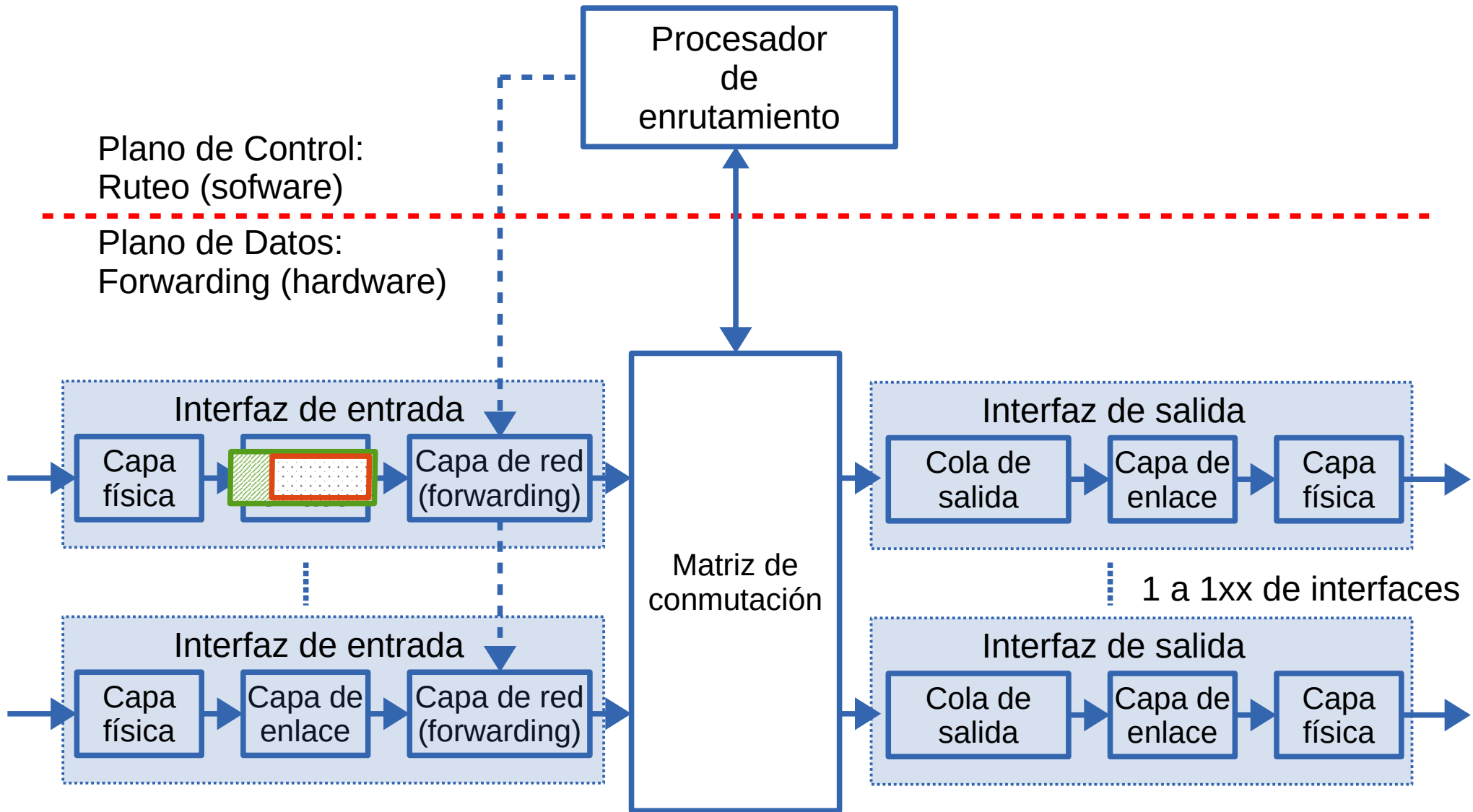
¿Cómo funciona un enrutador?

- Vista de alto nivel de un enrutador o router genérico



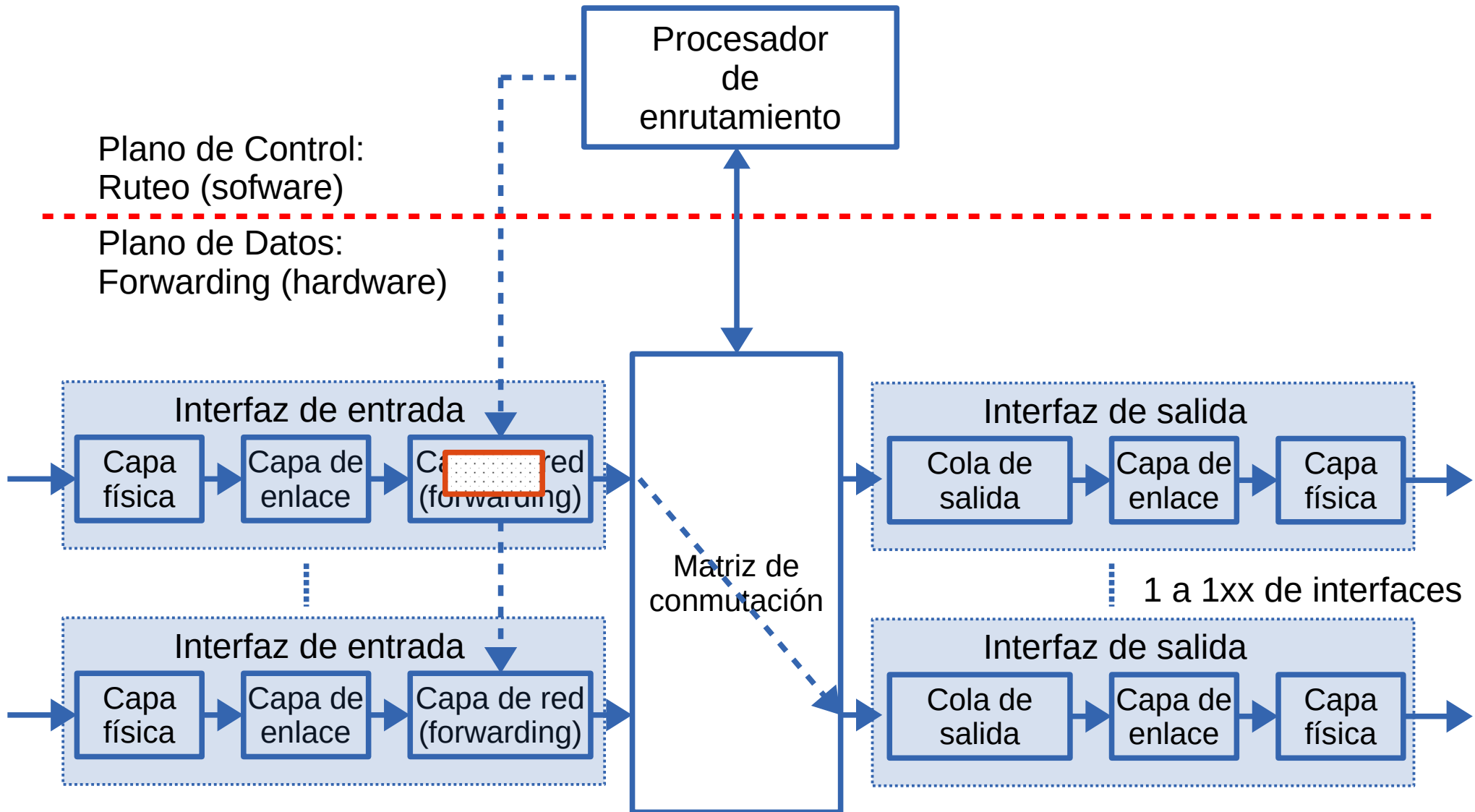
¿Cómo funciona un enrutador?

- Vista de alto nivel de un enrutador o router genérico



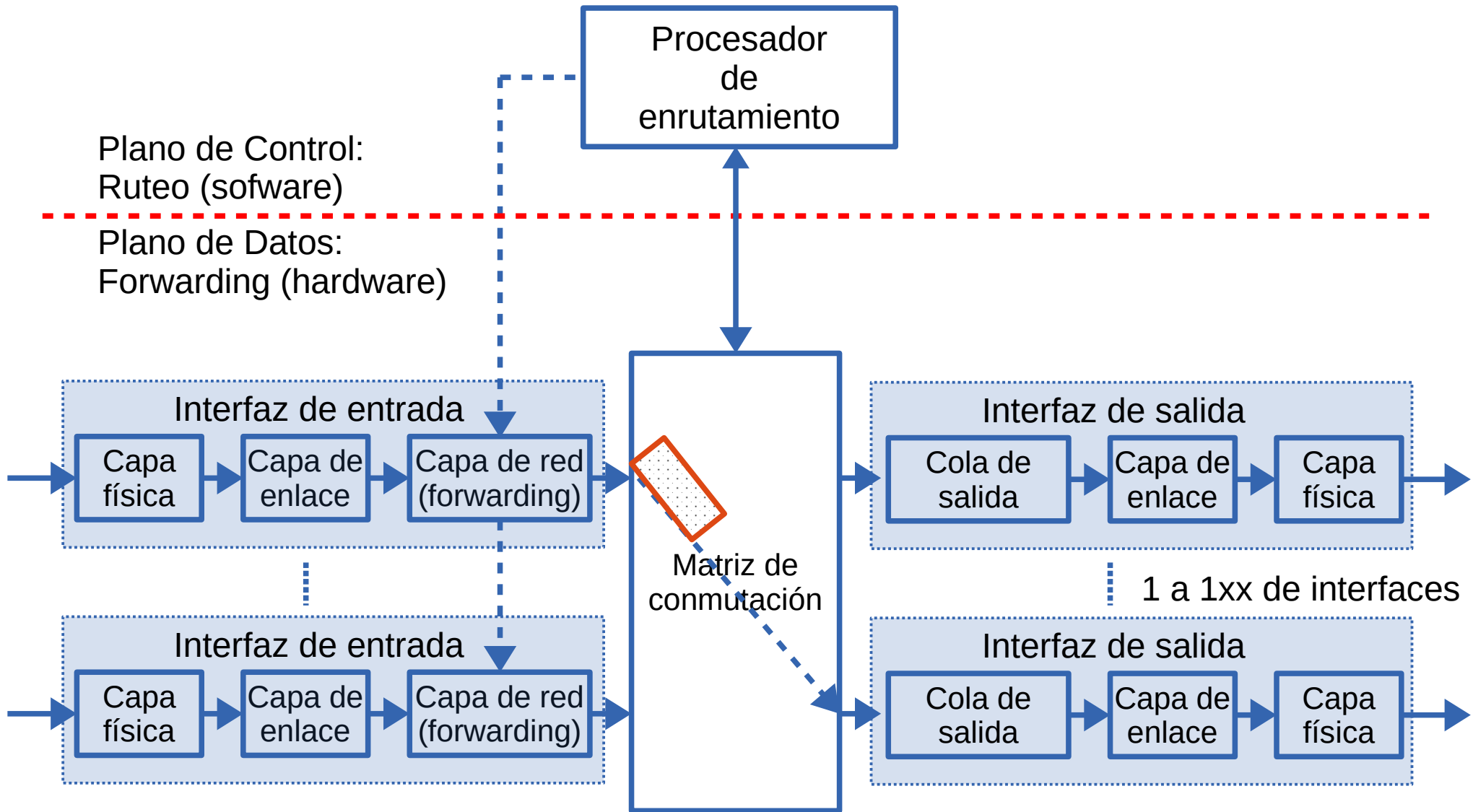
¿Cómo funciona un enrutador?

- Vista de alto nivel de un enrutador o router genérico



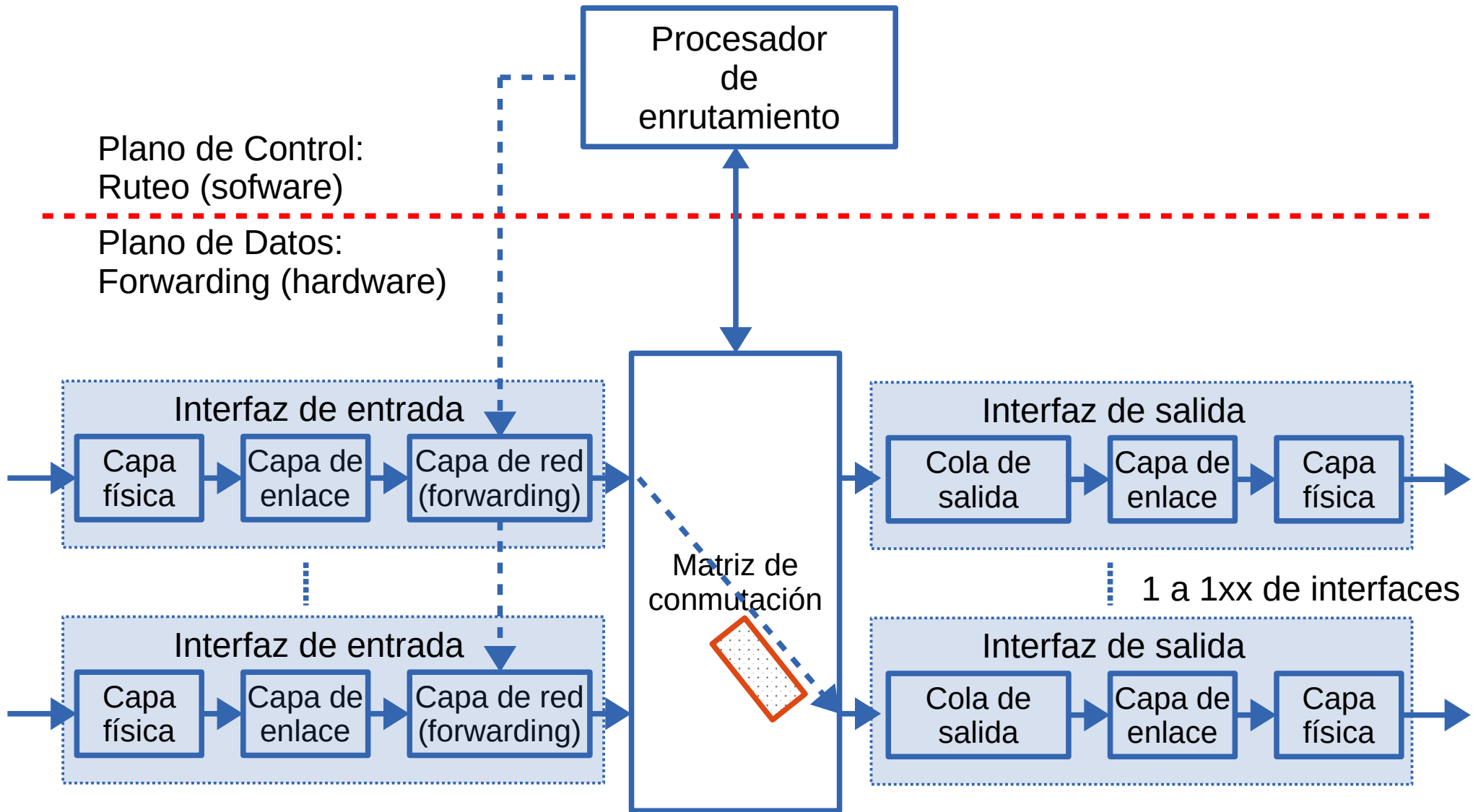
¿Cómo funciona un enrutador?

- Vista de alto nivel de un enrutador o router genérico



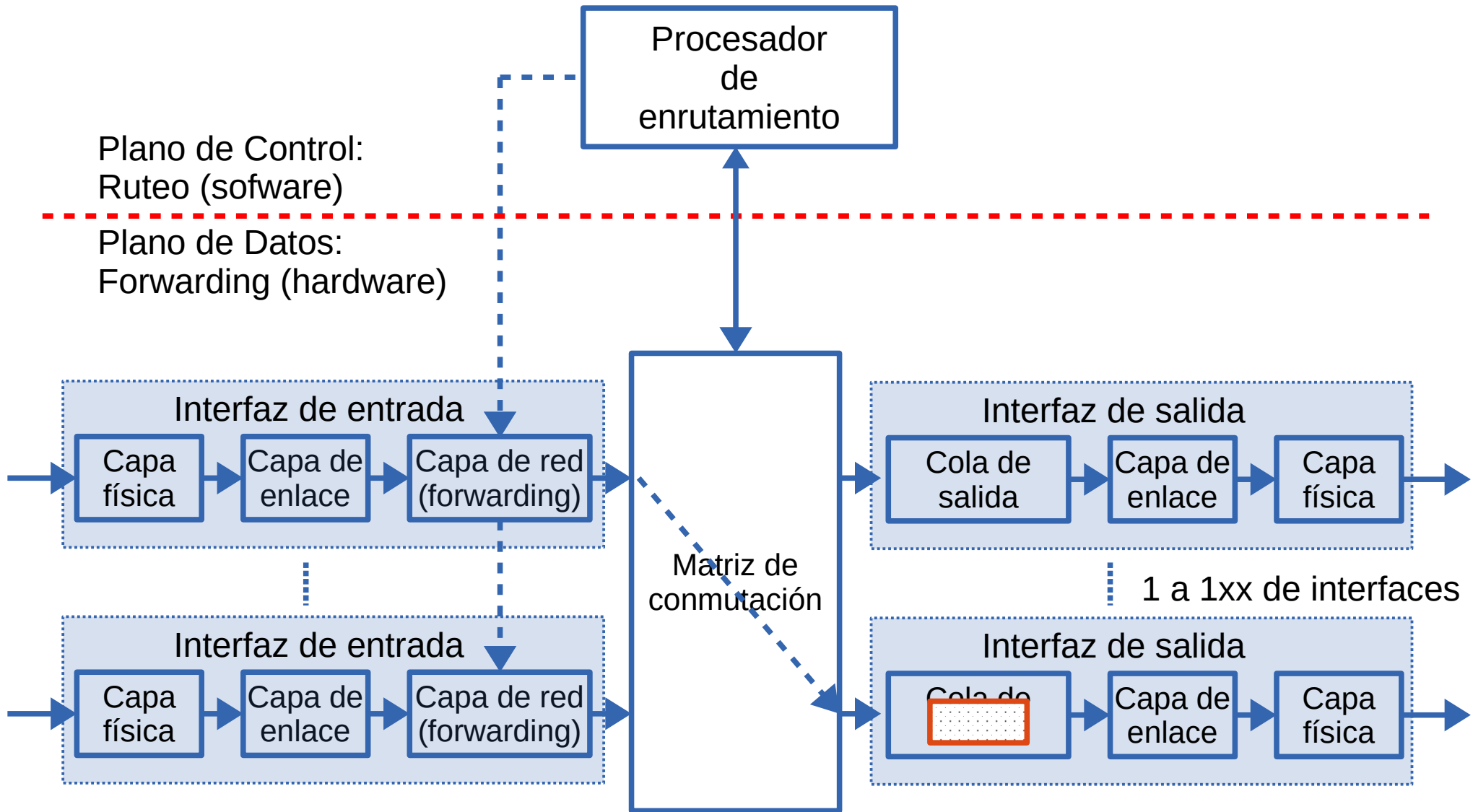
¿Cómo funciona un enrutador?

- Vista de alto nivel de un enrutador o router genérico



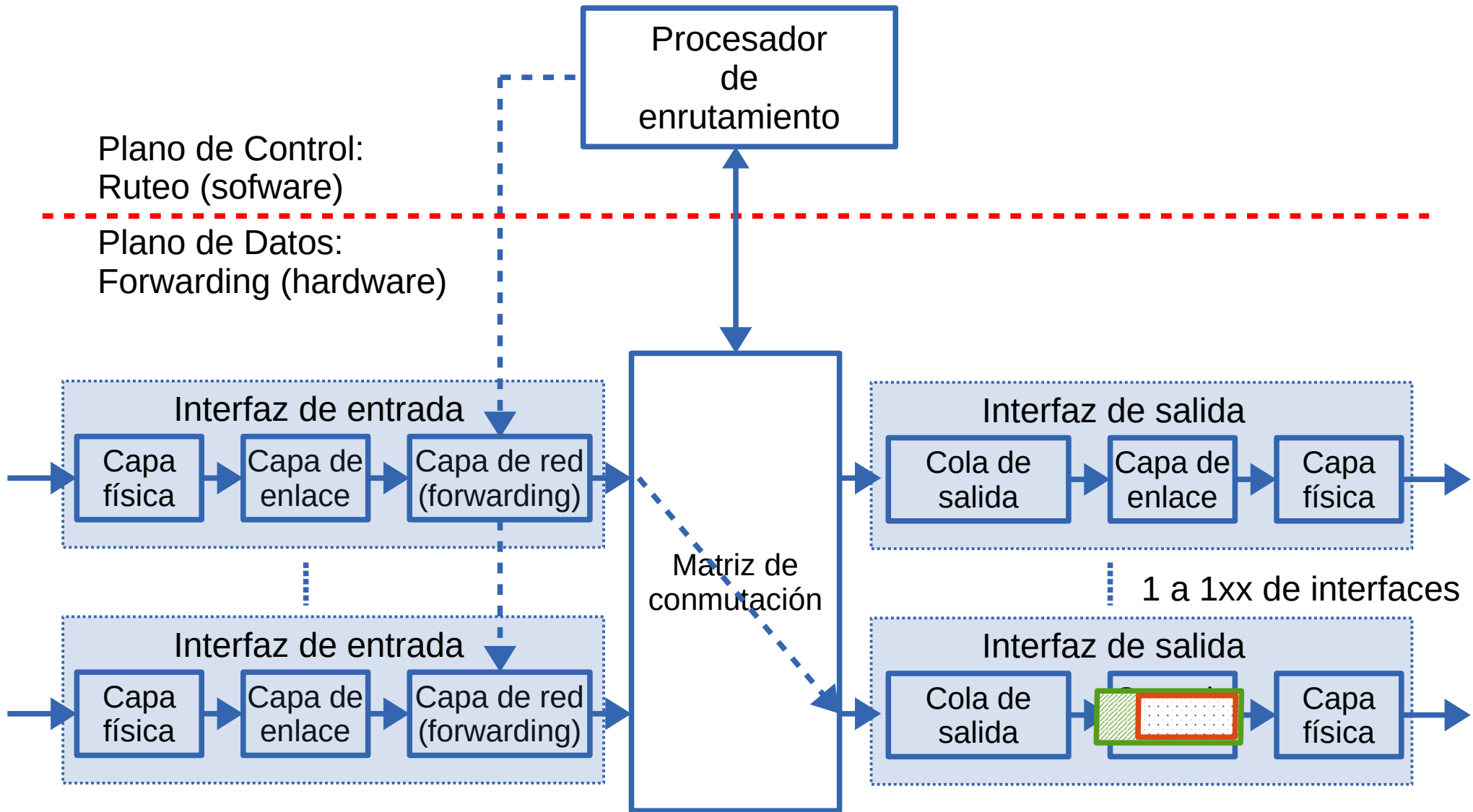
¿Cómo funciona un enrutador?

- Vista de alto nivel de un enrutador o router genérico



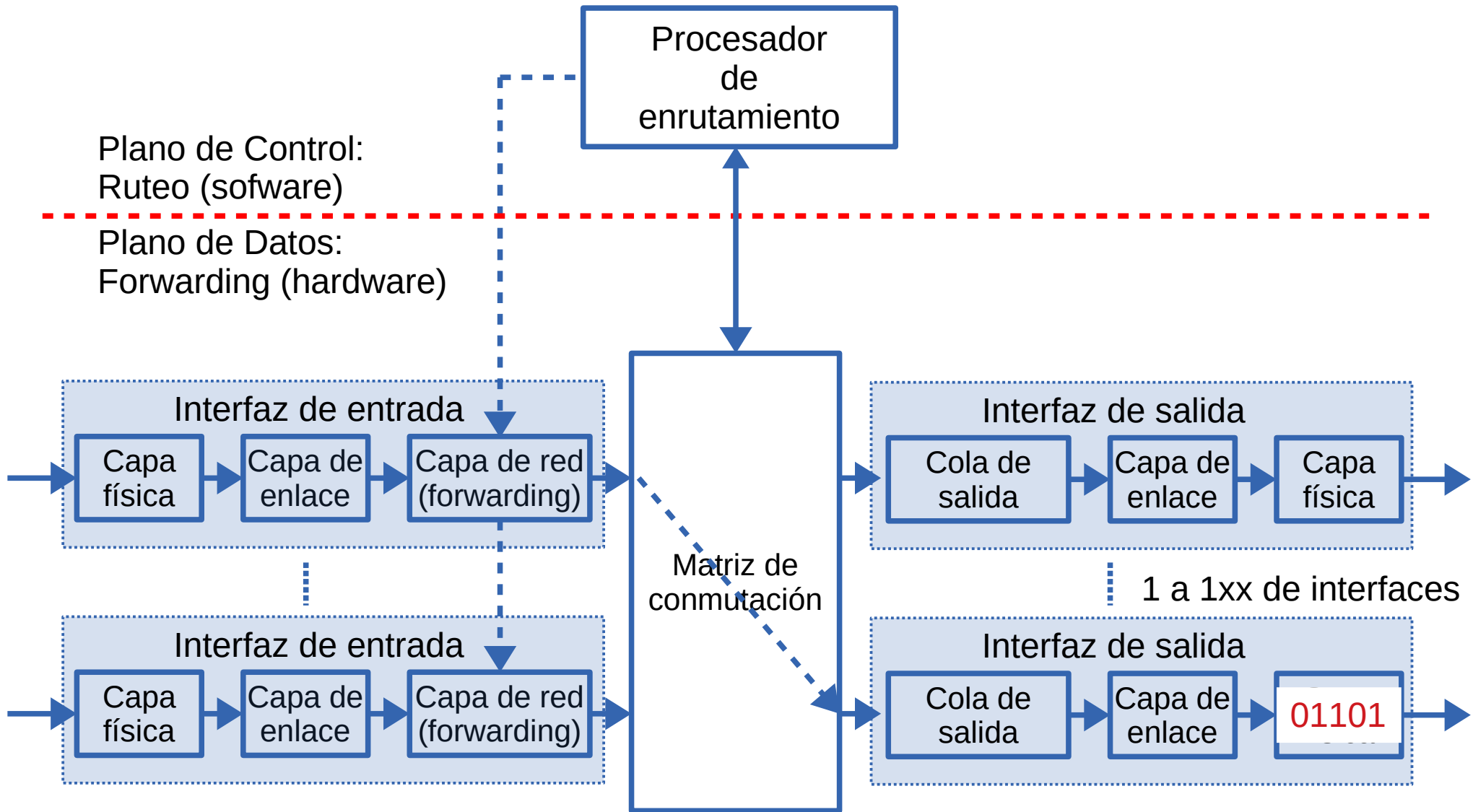
¿Cómo funciona un enrutador?

- Vista de alto nivel de un enrutador o router genérico



¿Cómo funciona un enrutador?

- Vista de alto nivel de un enrutador o router genérico



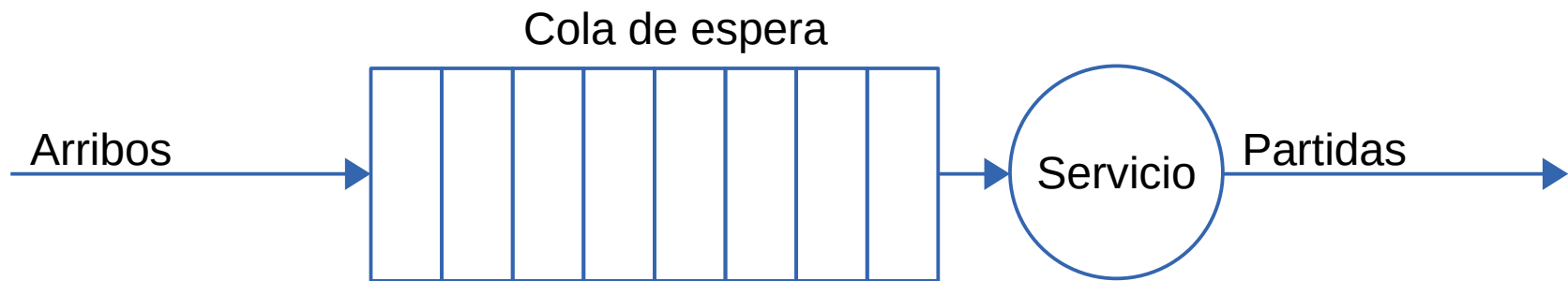
Arquitectura genérica de un enrutador

- **Interfaces de entrada** (puertas o puertos de entrada)
 - Funciones de capa física y de capa de enlace de datos
 - Consulta a la tabla de forwarding para determinar a qué interfaz de salida debe encaminarse el paquete usando la matriz de conmutación
 - Encolamiento si la matriz de conmutación no es suficientemente rápida
- **Matriz de conmutación** (switching fabric)
 - Conecta las interfaces de entrada con las de salida
 - Via memoria (compartida), via bus o por matriz de conexiones
- **Interfaces de salida**
 - Encolamiento de paquetes para ser enviados por una interfaz
 - Funciones de capa física y de capa de enlace de datos
 - Si las interfaces son bidireccionales, estará en la misma placa que la interfaz de entrada correspondiente al mismo enlace
- **Procesador de enrutamiento** (routing processor)
 - En la arquitectura tradicional implementa los protocolos y algoritmos de ruteo y calcula las tablas de forwarding en el mismo enrutador
 - En la arquitectura SDN se encarga de la comunicación con el controlador

Despacho de paquetes

- Los paquetes en cola para salir por una interfaz pueden ser tratados con diferentes **políticas**:
 - El primero que llega es el primero que se atiende:
 - **FIFO** (first-In-first-out) o FCFS (first-come-first-served)
 - Encolamiento con prioridades
 - **Priority queuing**
 - Encolamiento equitativo ponderado
 - Procesamiento de todas las colas en “round robin” pero con pesos
 - **Weighted fair queuing** (WFQ)

Despacho de paquetes: FIFO



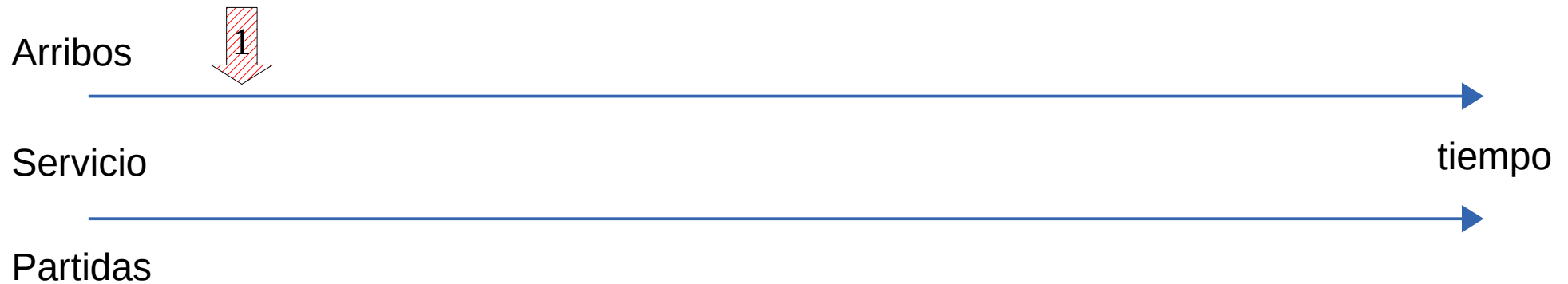
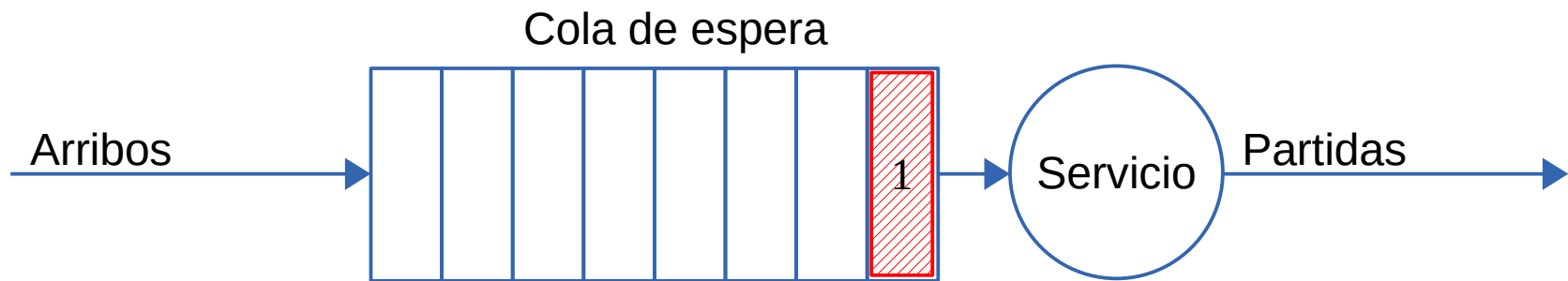
Arribos

Servicio

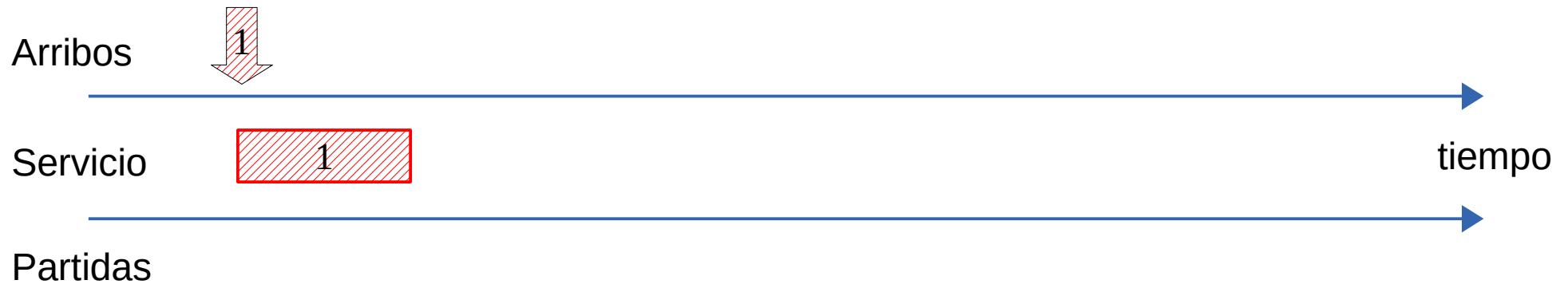
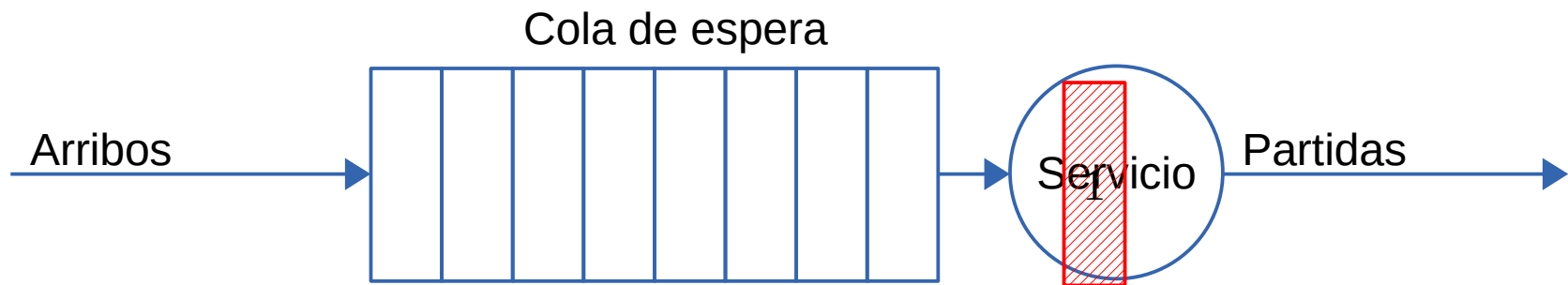
Partidas

tiempo

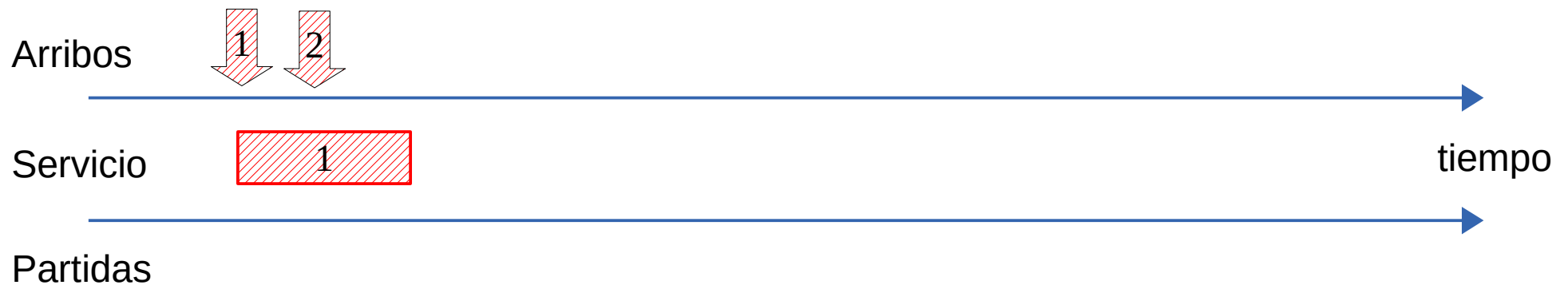
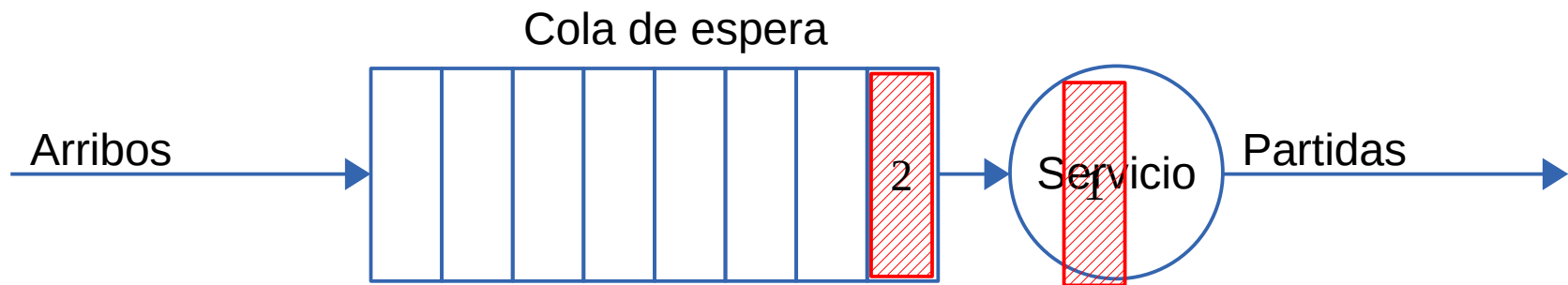
Despacho de paquetes: FIFO



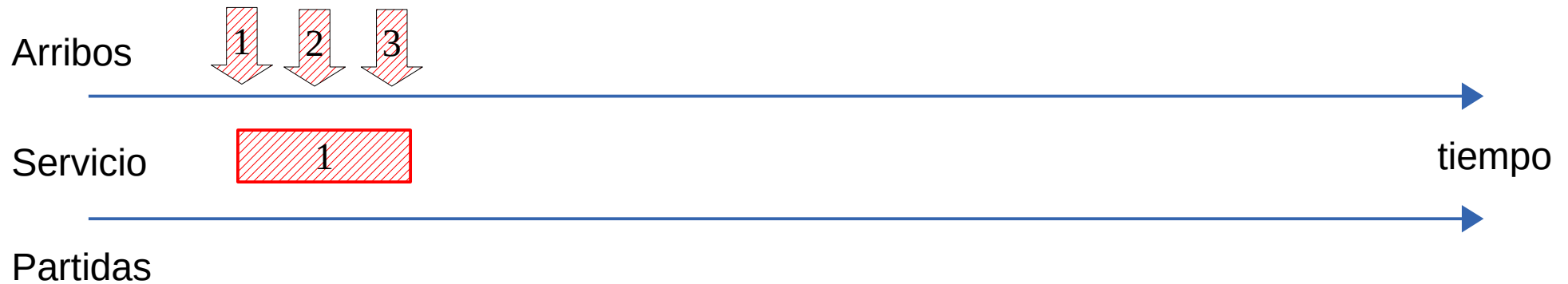
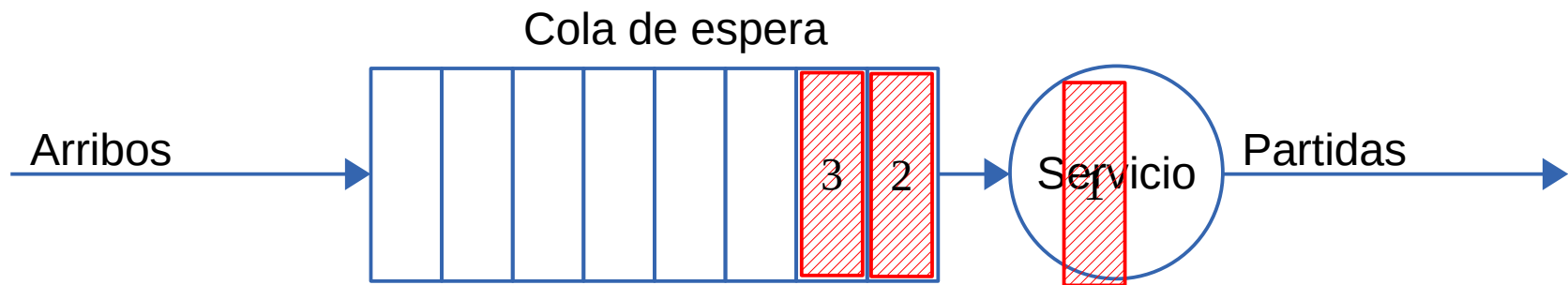
Despacho de paquetes: FIFO



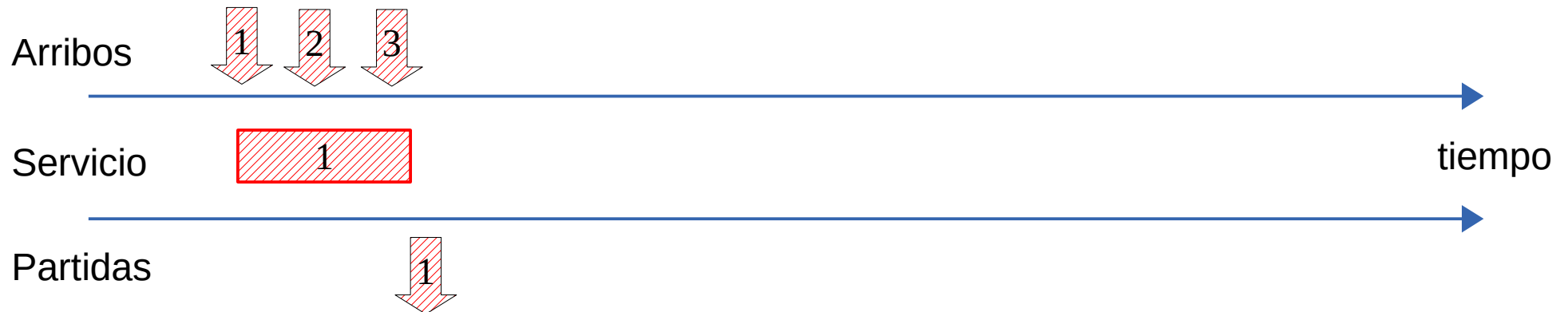
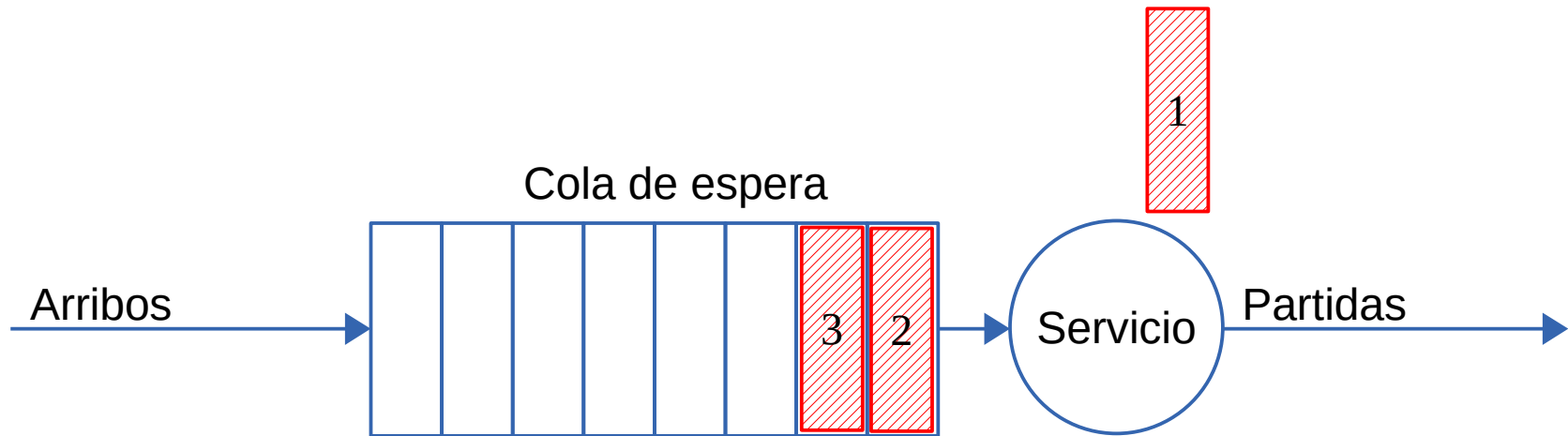
Despacho de paquetes: FIFO



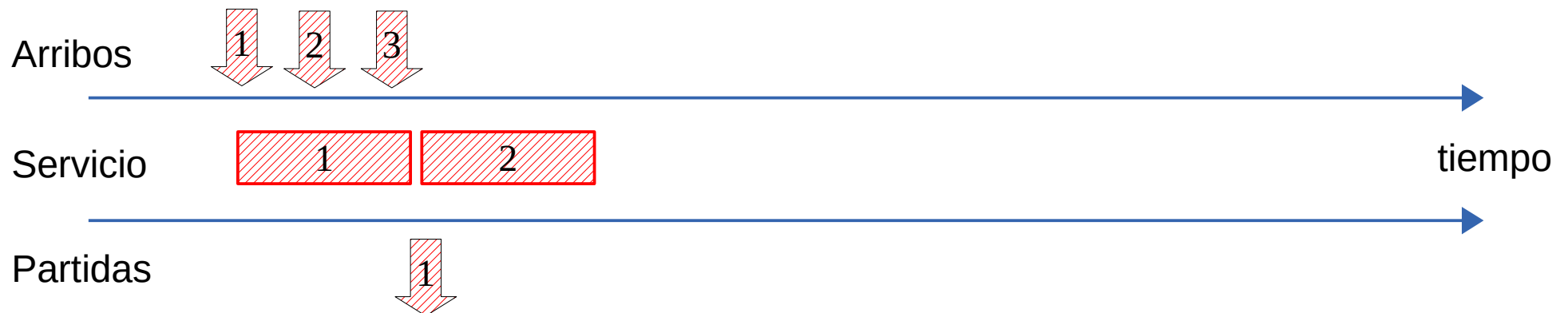
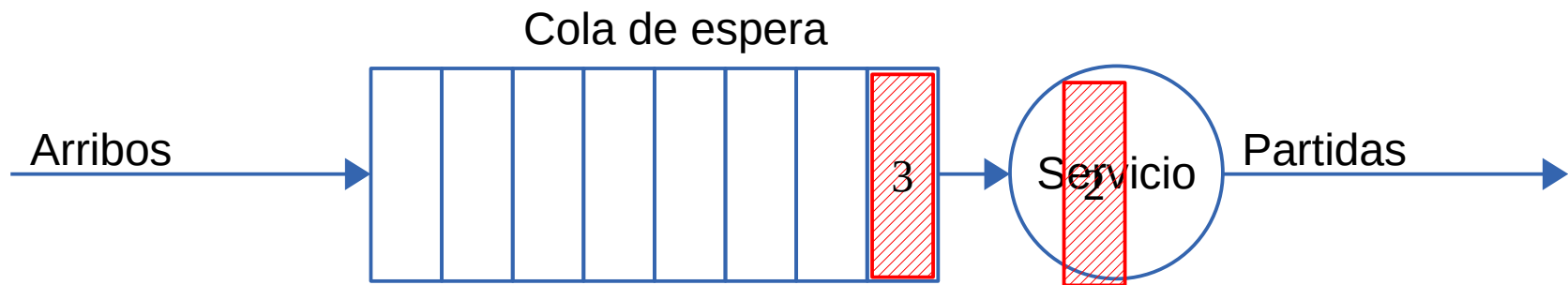
Despacho de paquetes: FIFO



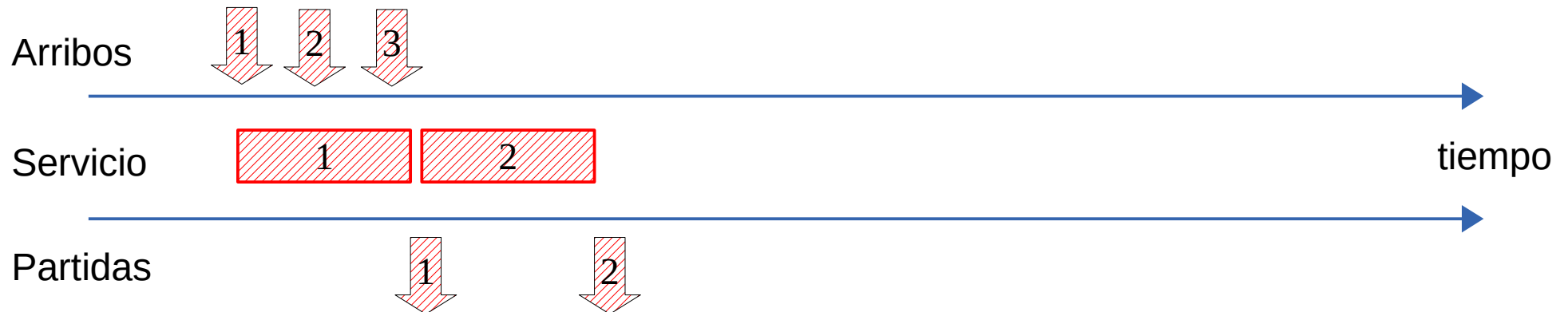
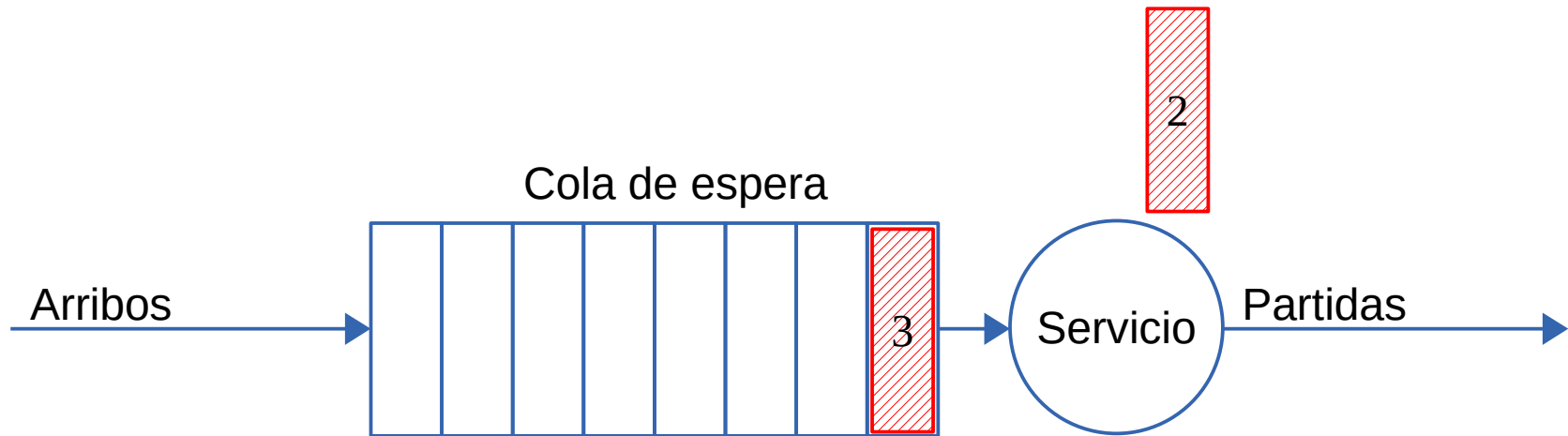
Despacho de paquetes: FIFO



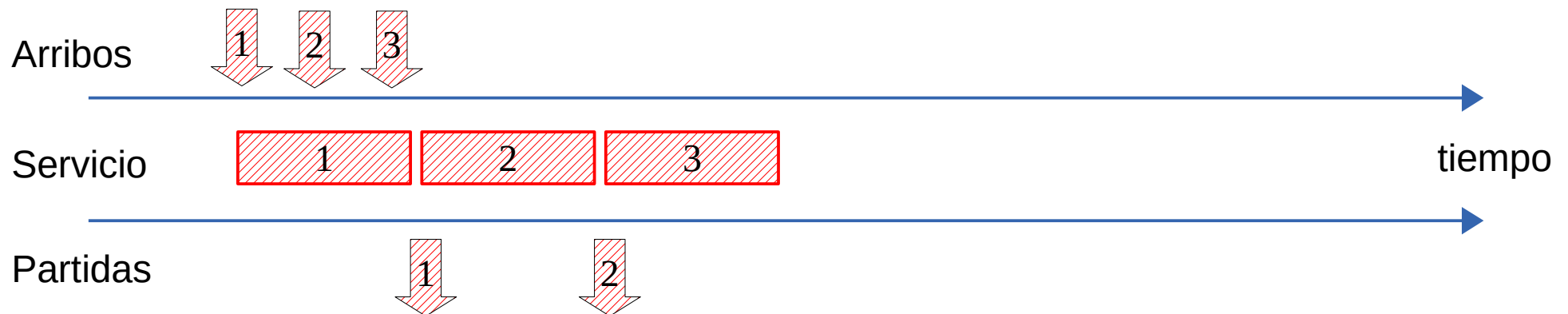
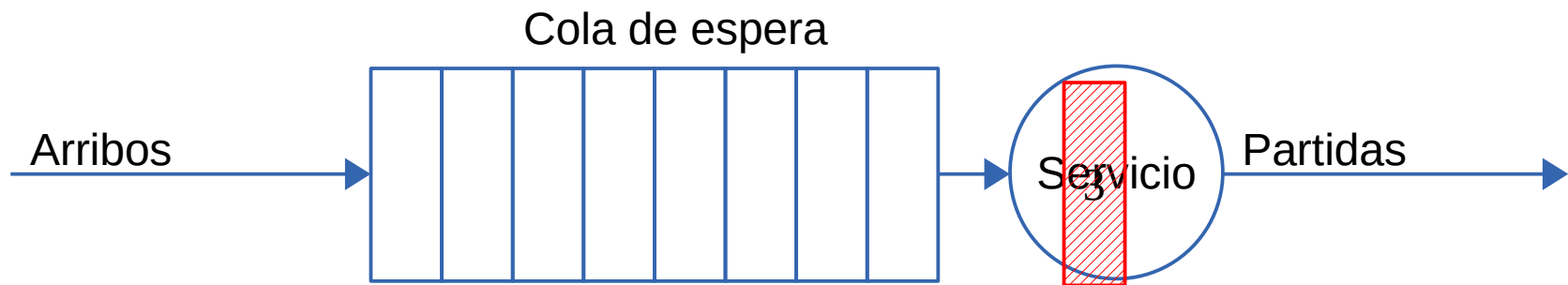
Despacho de paquetes: FIFO



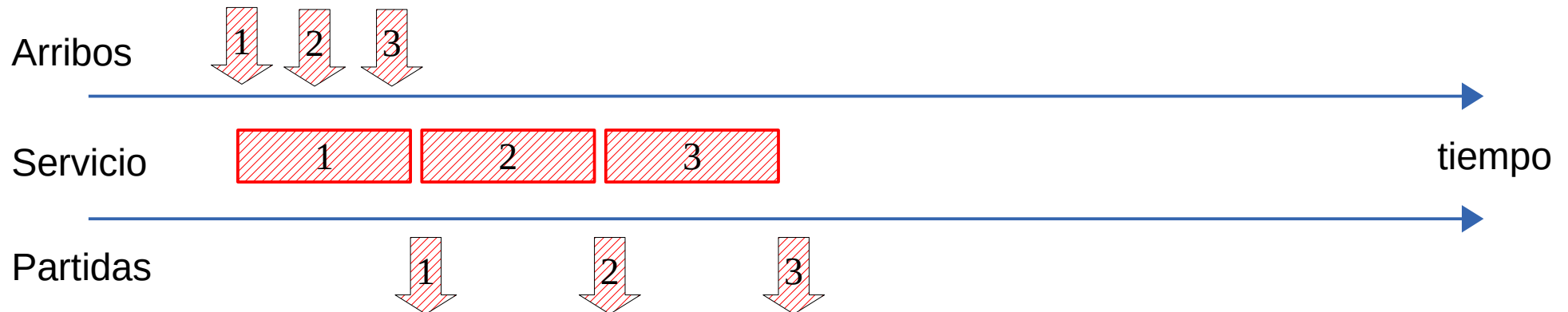
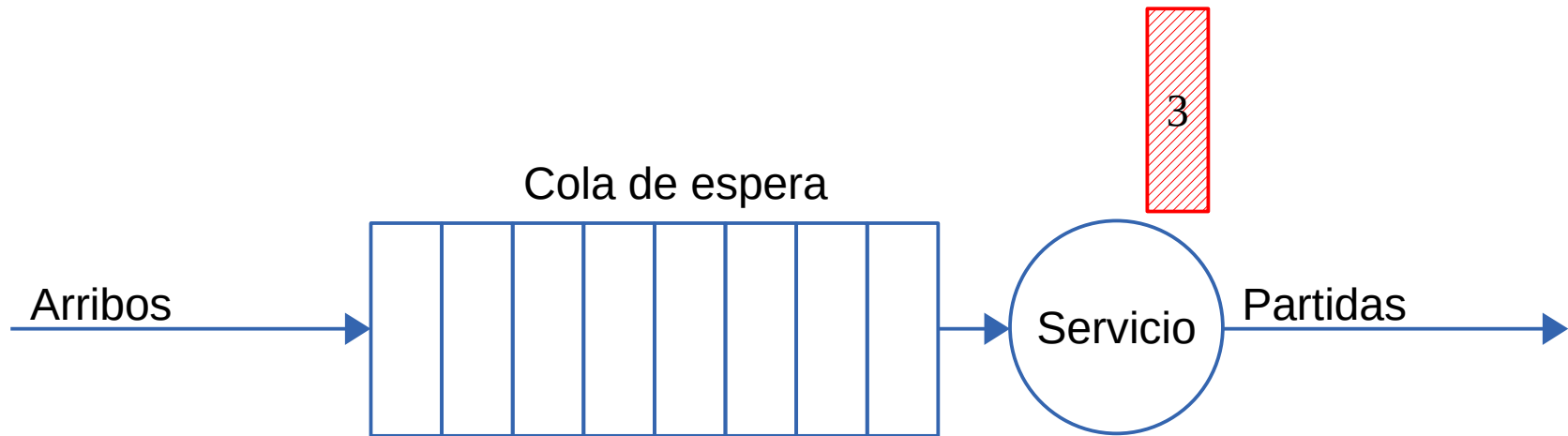
Despacho de paquetes: FIFO



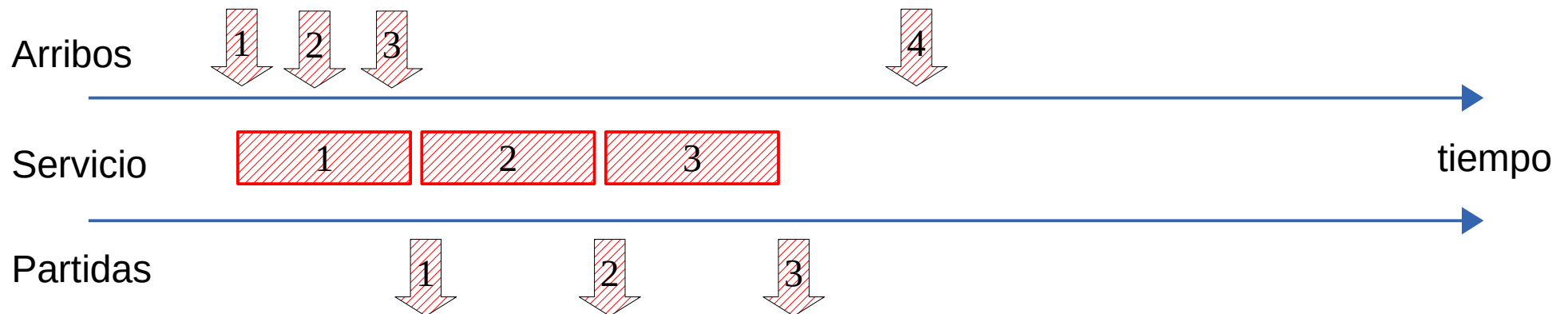
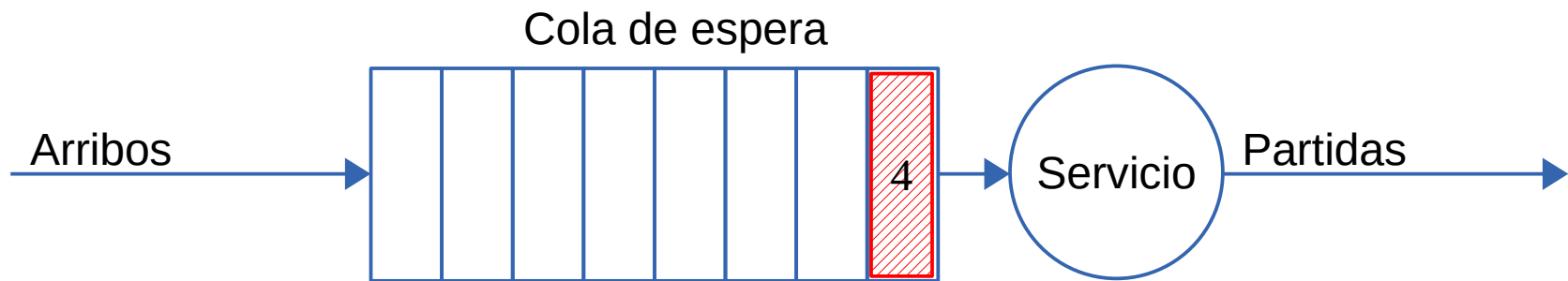
Despacho de paquetes: FIFO



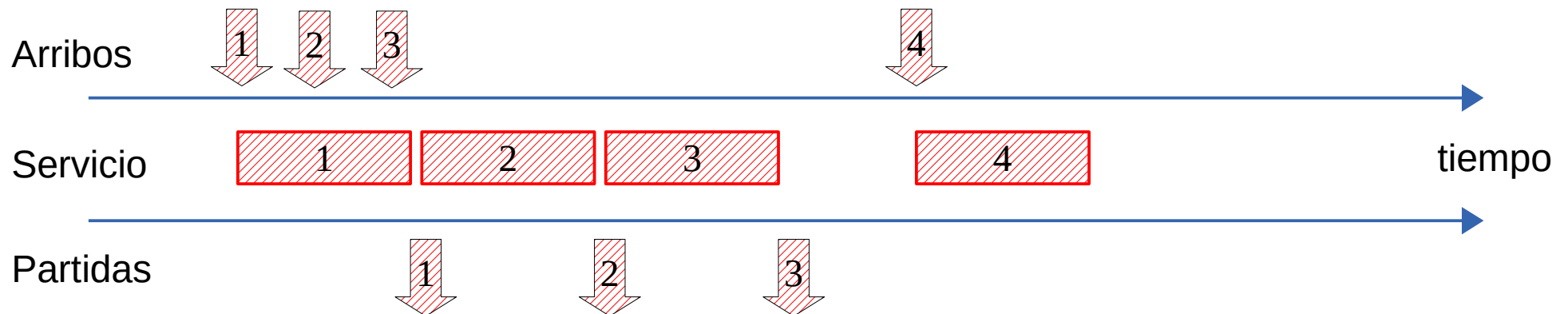
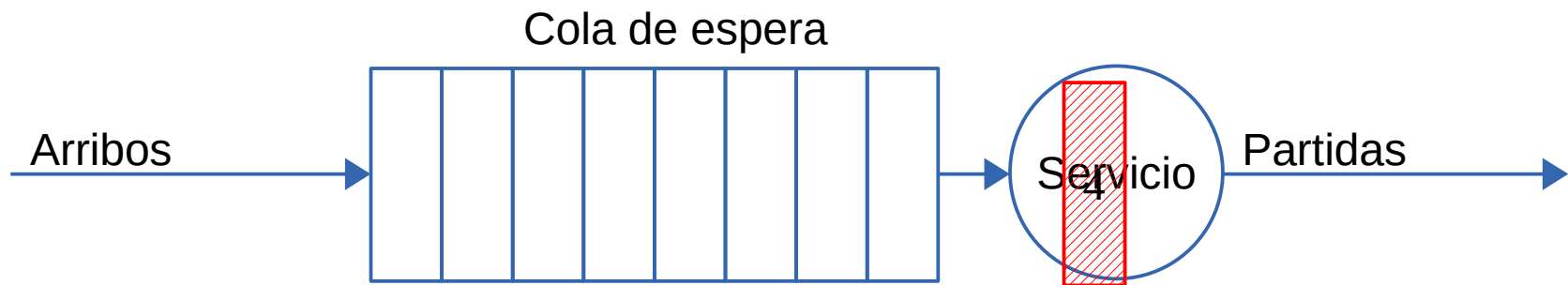
Despacho de paquetes: FIFO



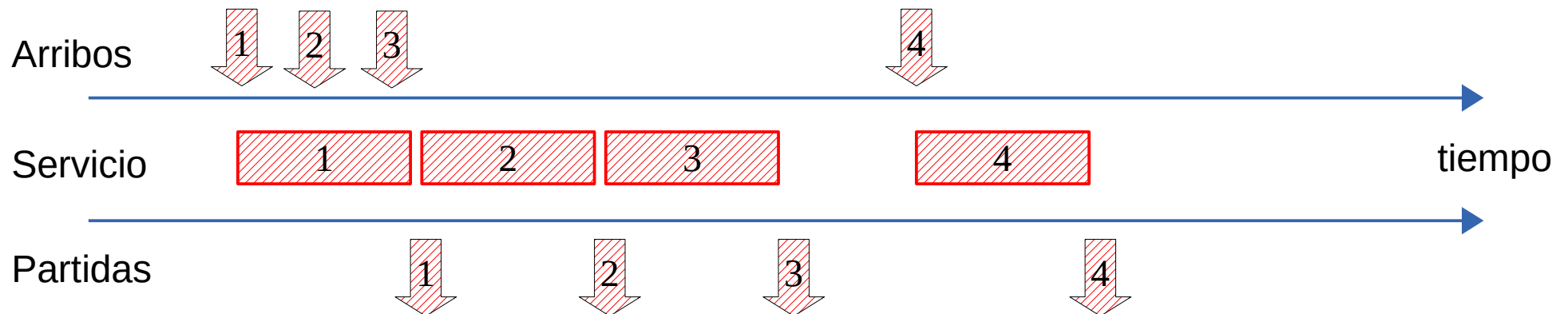
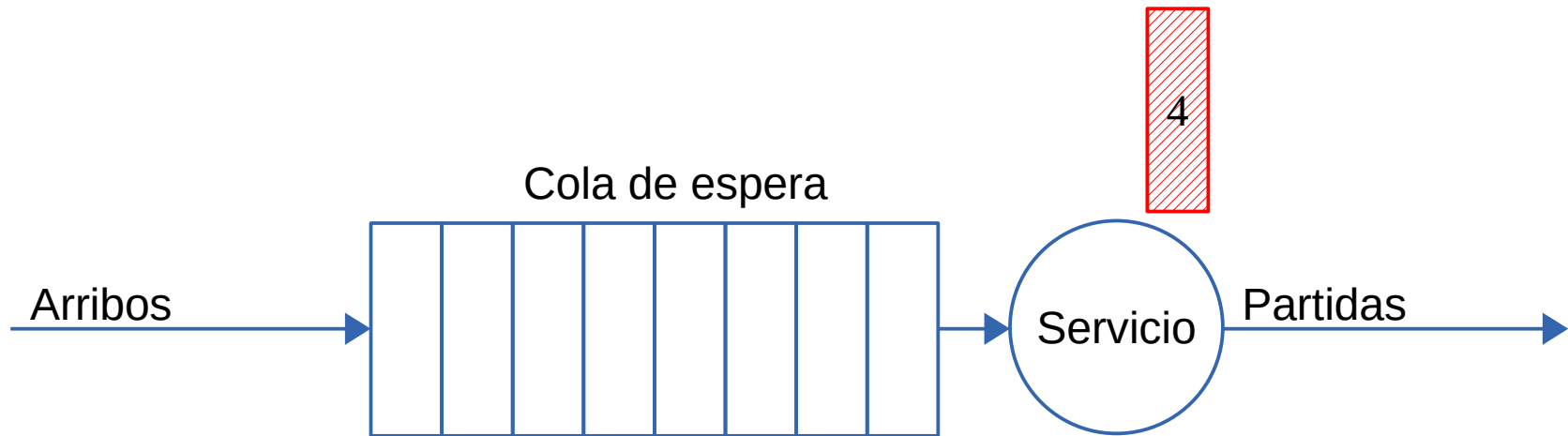
Despacho de paquetes: FIFO



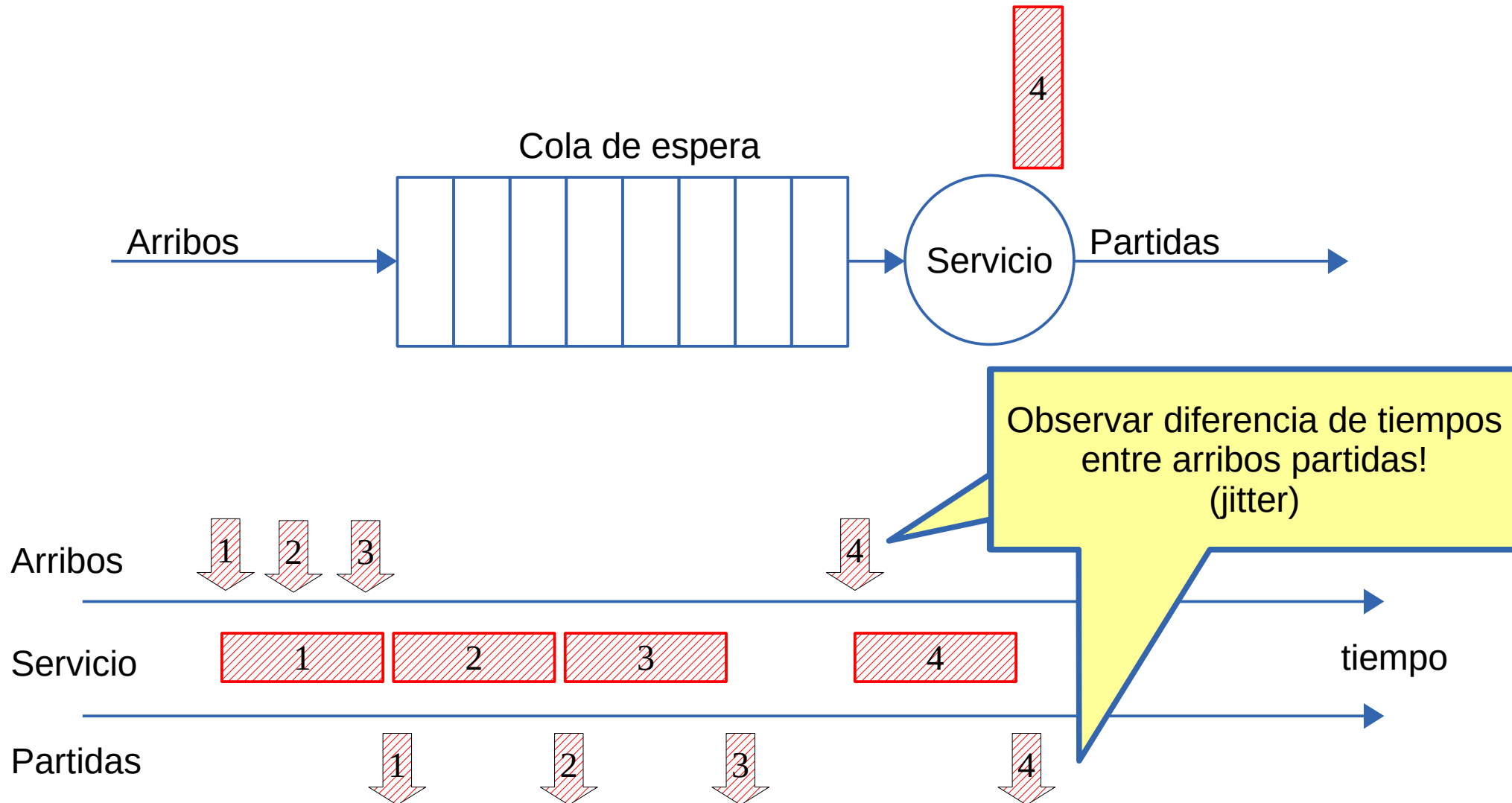
Despacho de paquetes: FIFO



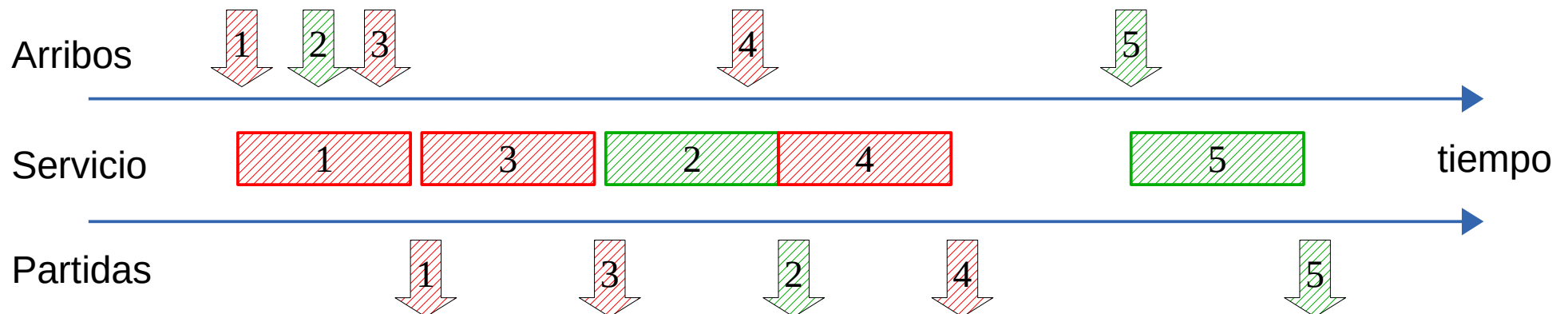
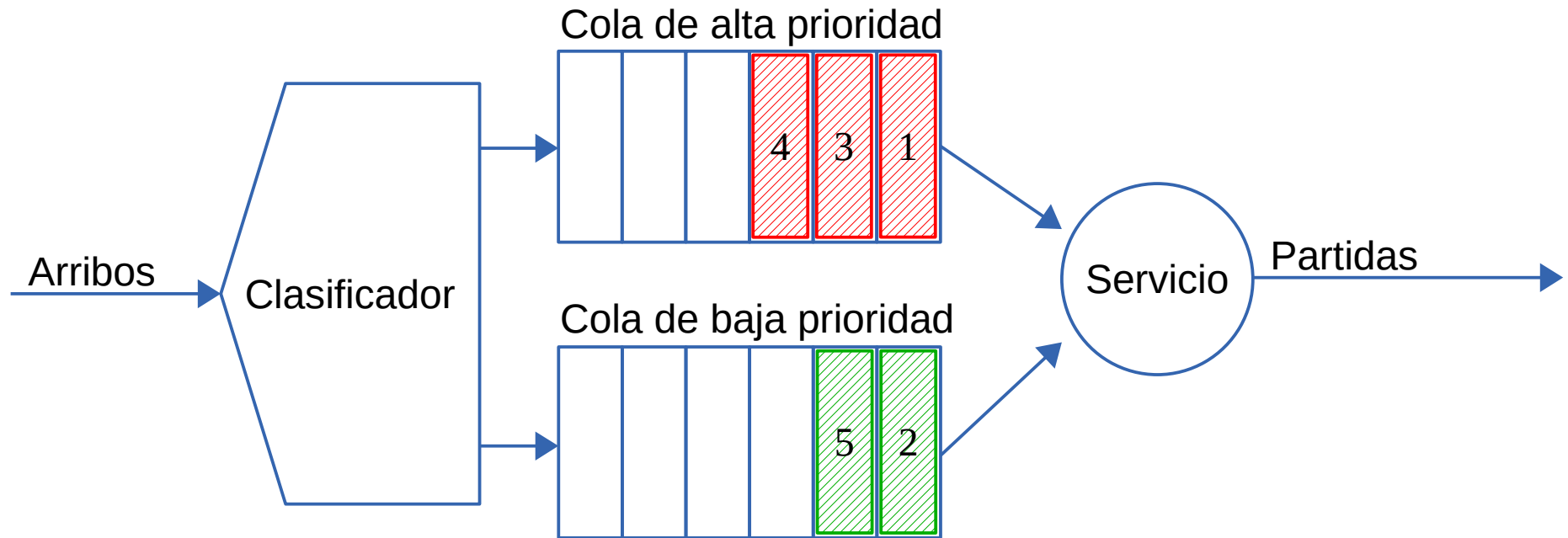
Despacho de paquetes: FIFO



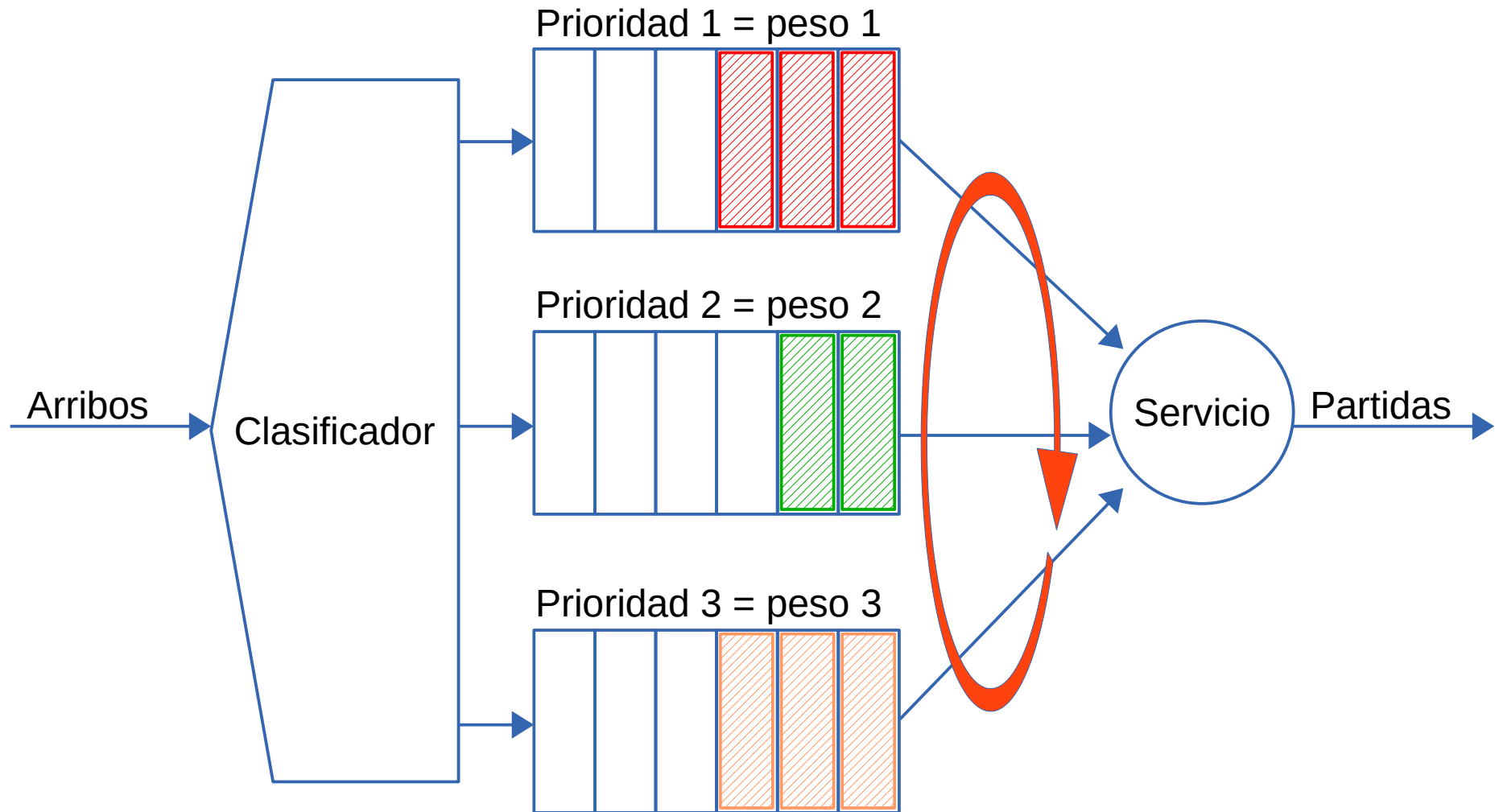
Despacho de paquetes: FIFO



Despacho de paquetes: Encolamiento con prioridades



Despacho de paquetes: WFQ



Redes de datos 1

Capa de red Plano de datos

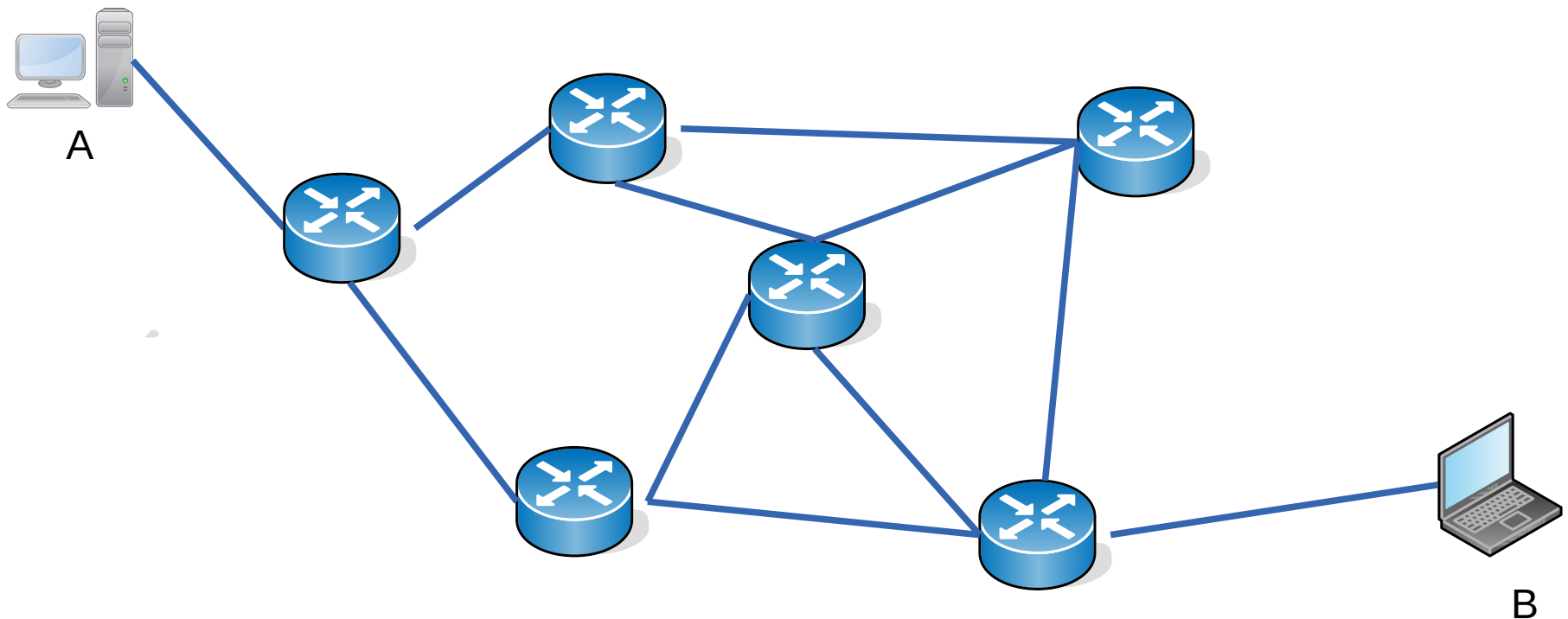
Facultad de Ingeniería – Universidad de la República
Instituto de Ingeniería Eléctrica

Agenda

- Conceptos de capa de red
- Plano de datos
 - Encaminamiento generalizado y encaminamiento clásico
 - Capa de red en Internet
 - Paquete IP versión 4
 - Direcciones IPv4
 - Rangos de direcciones
 - Tablas de forwarding
 - Búsqueda en las tablas de forwarding
 - Asignación de direcciones
 - Escasez de direcciones: NAT
 - IP versión 6
- Plano de control
- Redes de circuitos virtuales

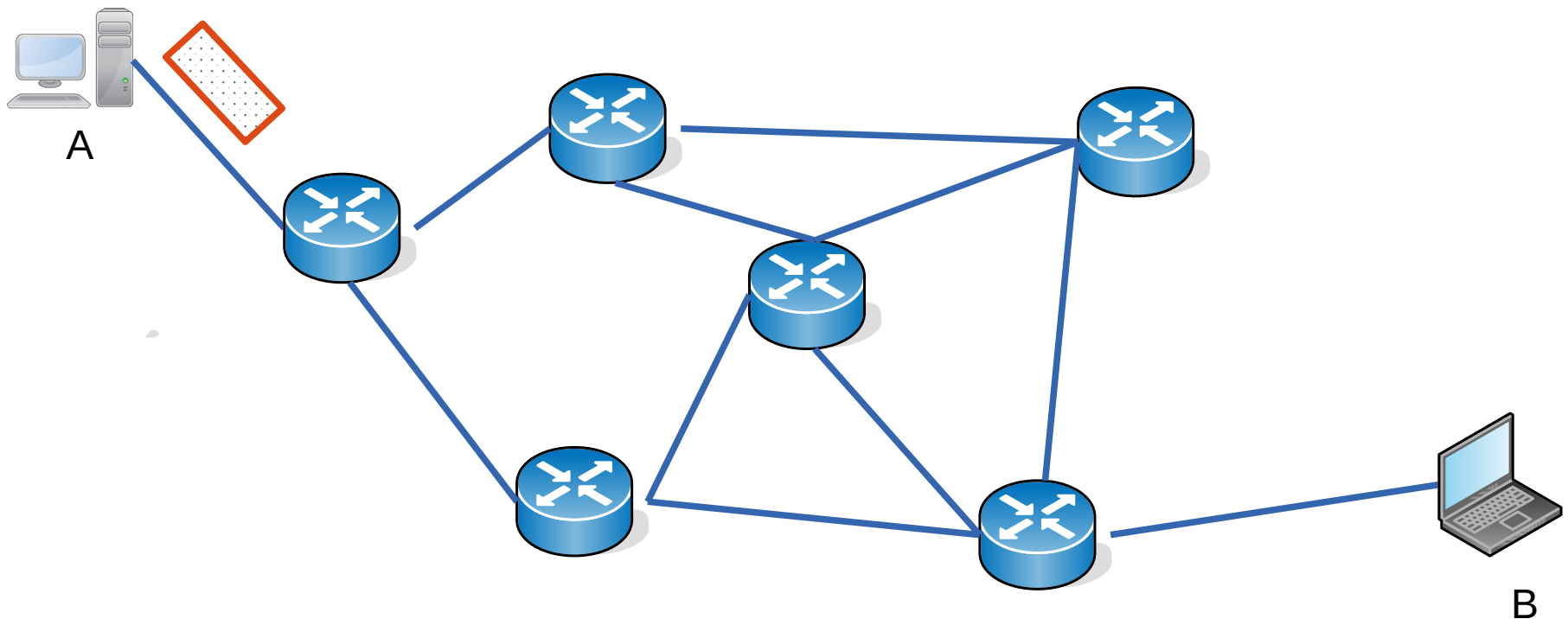
Plano de datos

- **Plano de Datos:** Función de **encaminamiento** (forwarding)
- Cada nodo **recibe** un paquete por una de sus **líneas de entrada** o interfaces, **decide en base a una tabla** cuál es la **línea de salida o próximo salto** más adecuado para ese destino y encamina el paquete hacia allí
- Esas tablas se llaman **tablas de forwarding**



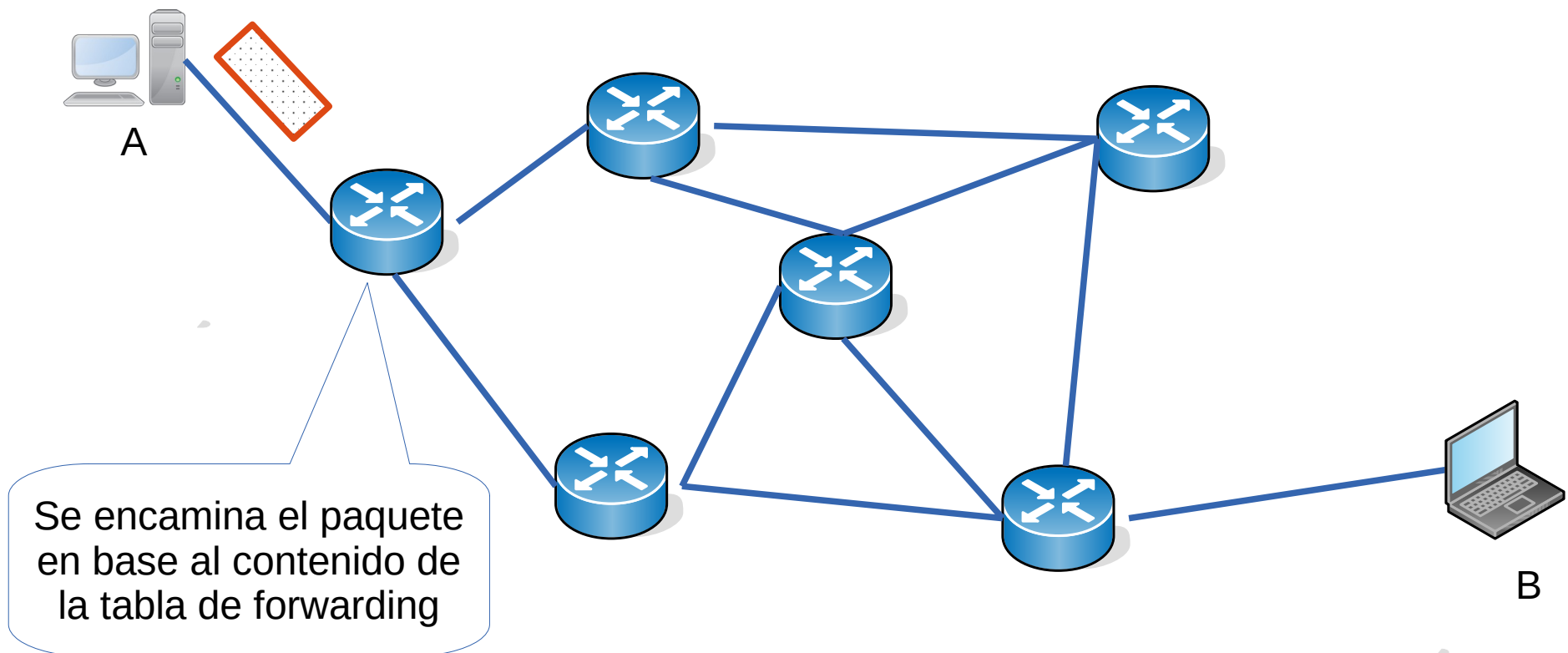
Plano de datos

- **Plano de Datos:** Función de **encaminamiento** (forwarding)
- Cada nodo **recibe** un paquete por una de sus **líneas de entrada** o interfaces, **decide en base a una tabla** cuál es la **línea de salida o próximo salto** más adecuado para ese destino y encamina el paquete hacia allí
- Esas tablas se llaman **tablas de forwarding**



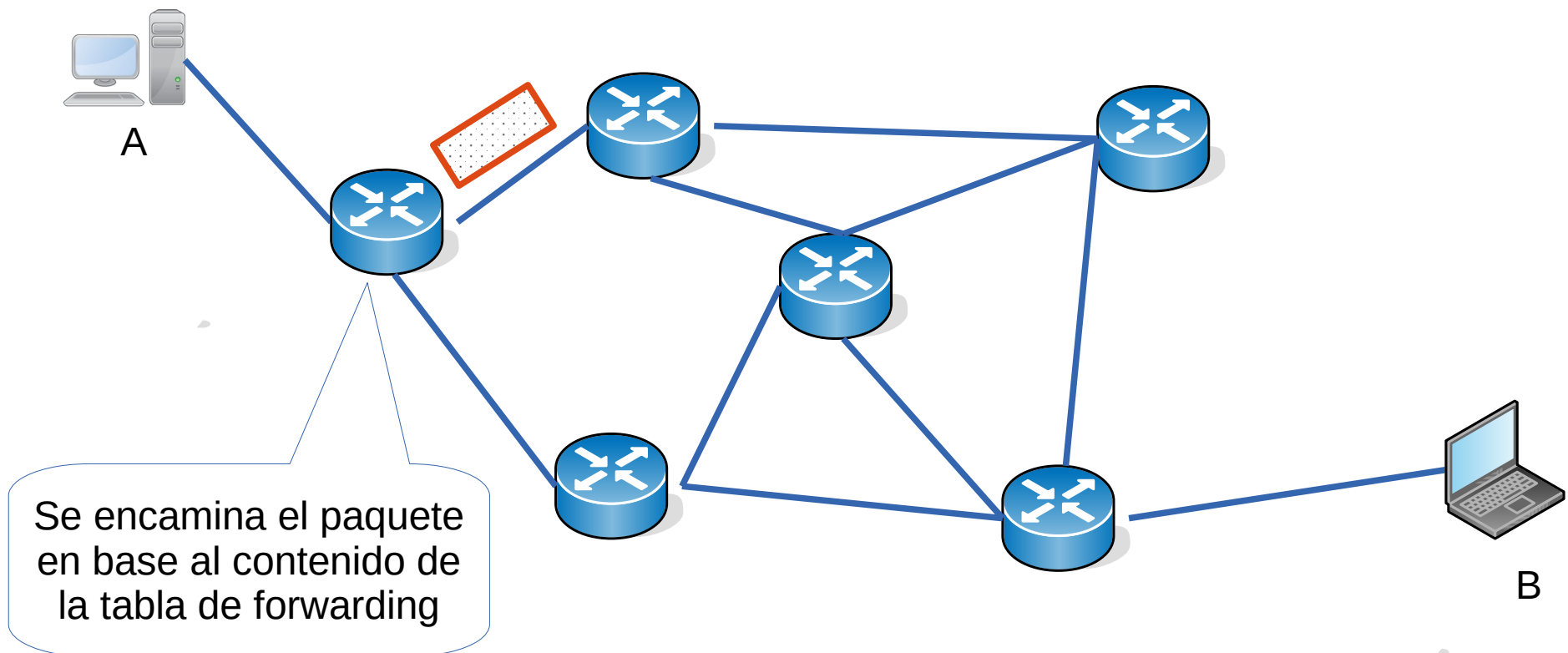
Plano de datos

- **Plano de Datos:** Función de **encaminamiento** (forwarding)
- Cada nodo **recibe** un paquete por una de sus **líneas de entrada** o interfaces, **decide en base a una tabla** cuál es la **línea de salida o próximo salto** más adecuado para ese destino y encamina el paquete hacia allí
- Esas tablas se llaman **tablas de forwarding**



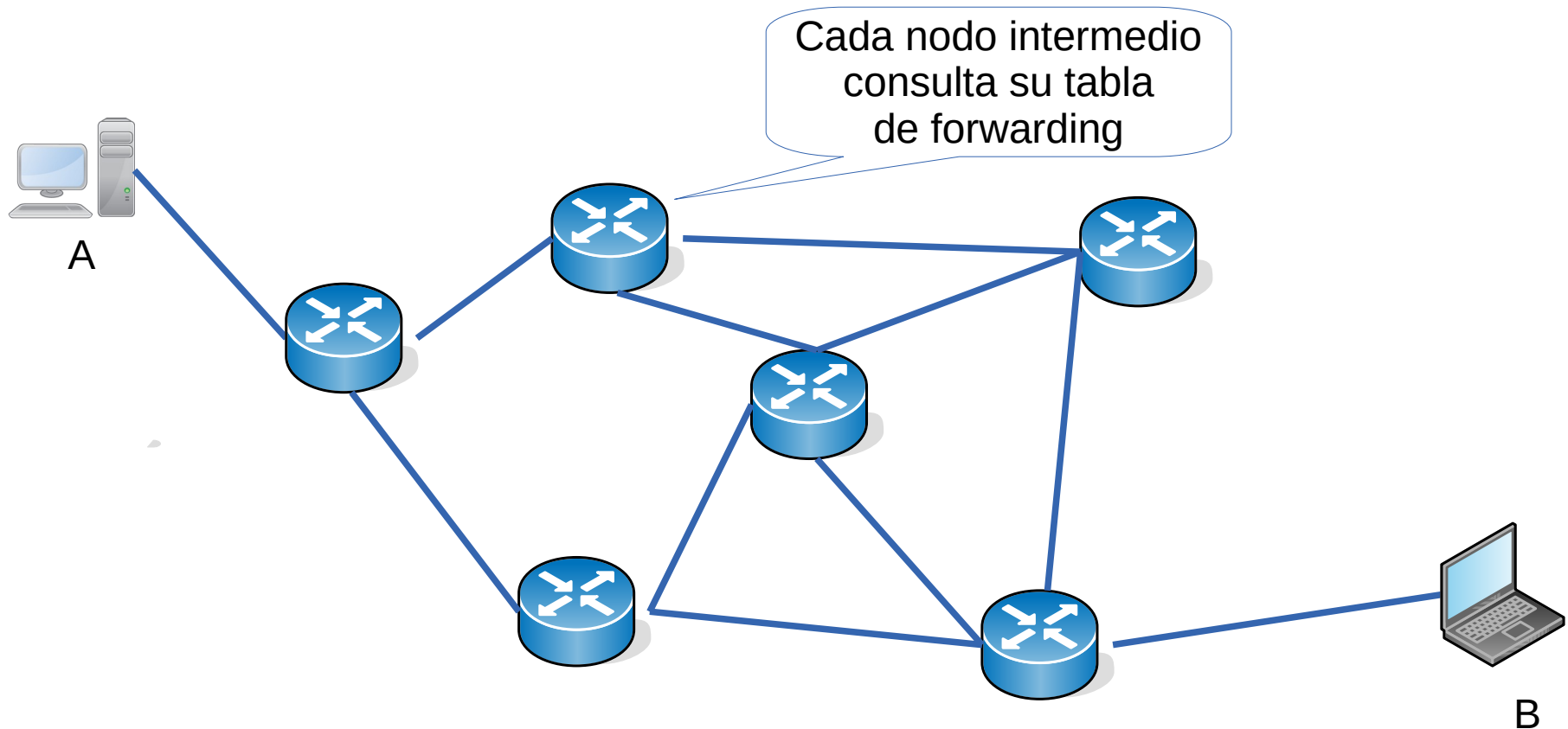
Plano de datos

- **Plano de Datos:** Función de **encaminamiento** (forwarding)
- Cada nodo **recibe** un paquete por una de sus **líneas de entrada** o interfaces, **decide en base a una tabla** cuál es la **línea de salida o próximo salto** más adecuado para ese destino y encamina el paquete hacia allí
- Esas tablas se llaman **tablas de forwarding**



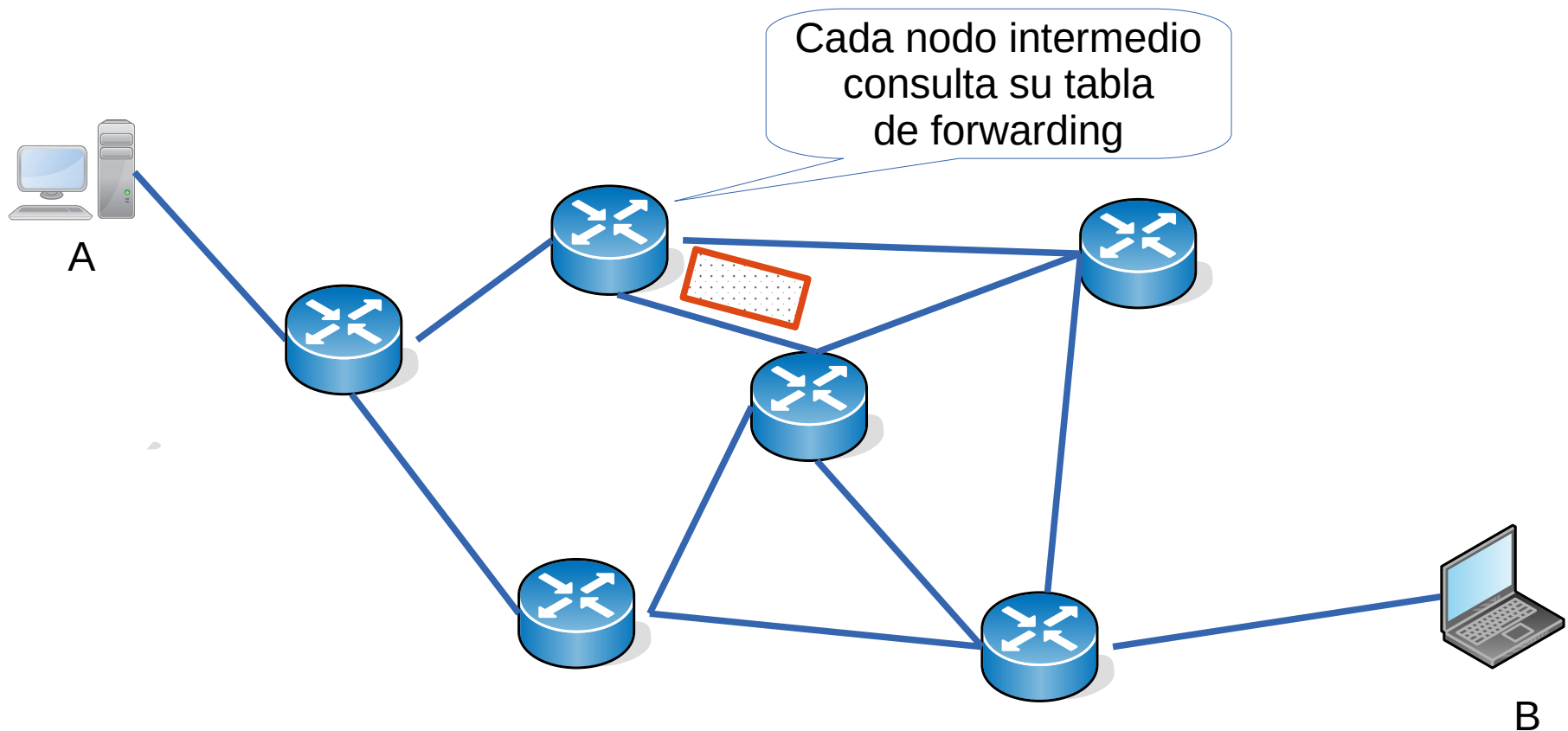
Plano de datos

- **Plano de Datos:** Función de **encaminamiento** (forwarding)
- Cada nodo **recibe** un paquete por una de sus **líneas de entrada** o interfaces, **decide en base a una tabla** cuál es la **línea de salida** o **próximo salto** más adecuado para ese destino y encamina el paquete hacia allí
- Esas tablas se llaman **tablas de forwarding**



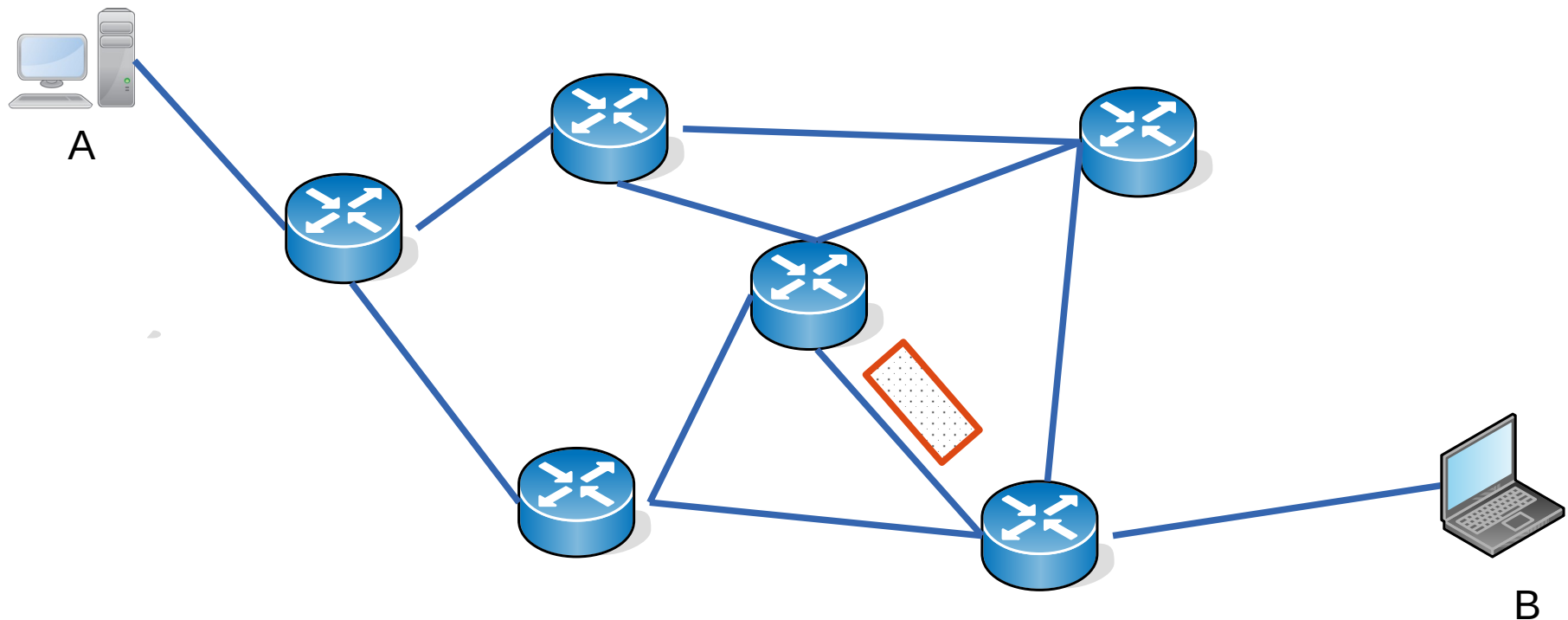
Plano de datos

- **Plano de Datos:** Función de **encaminamiento** (forwarding)
- Cada nodo **recibe** un paquete por una de sus **líneas de entrada** o interfaces, **decide en base a una tabla** cuál es la **línea de salida** o **próximo salto** más adecuado para ese destino y encamina el paquete hacia allí
- Esas tablas se llaman **tablas de forwarding**



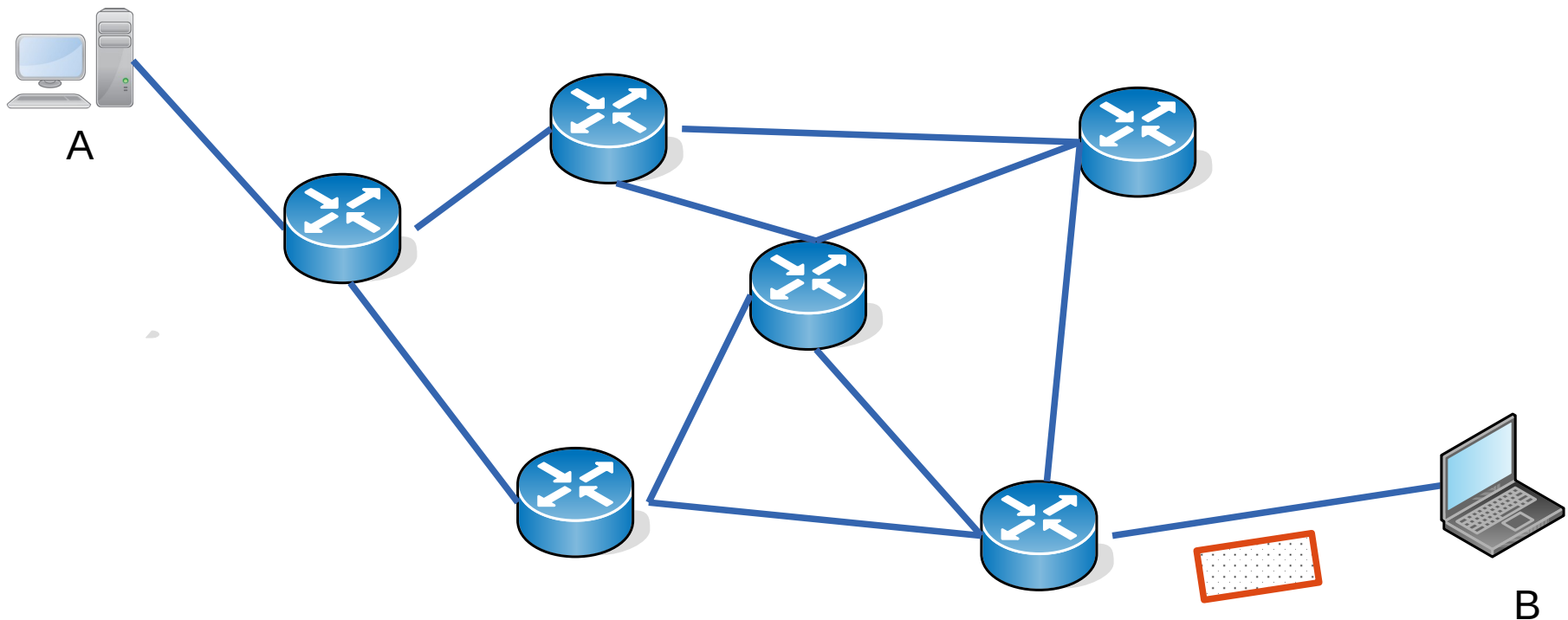
Plano de datos

- **Plano de Datos:** Función de **encaminamiento** (forwarding)
- Cada nodo **recibe** un paquete por una de sus **líneas de entrada** o interfaces, **decide en base a una tabla** cuál es la **línea de salida o próximo salto** más adecuado para ese destino y encamina el paquete hacia allí
- Esas tablas se llaman **tablas de forwarding**



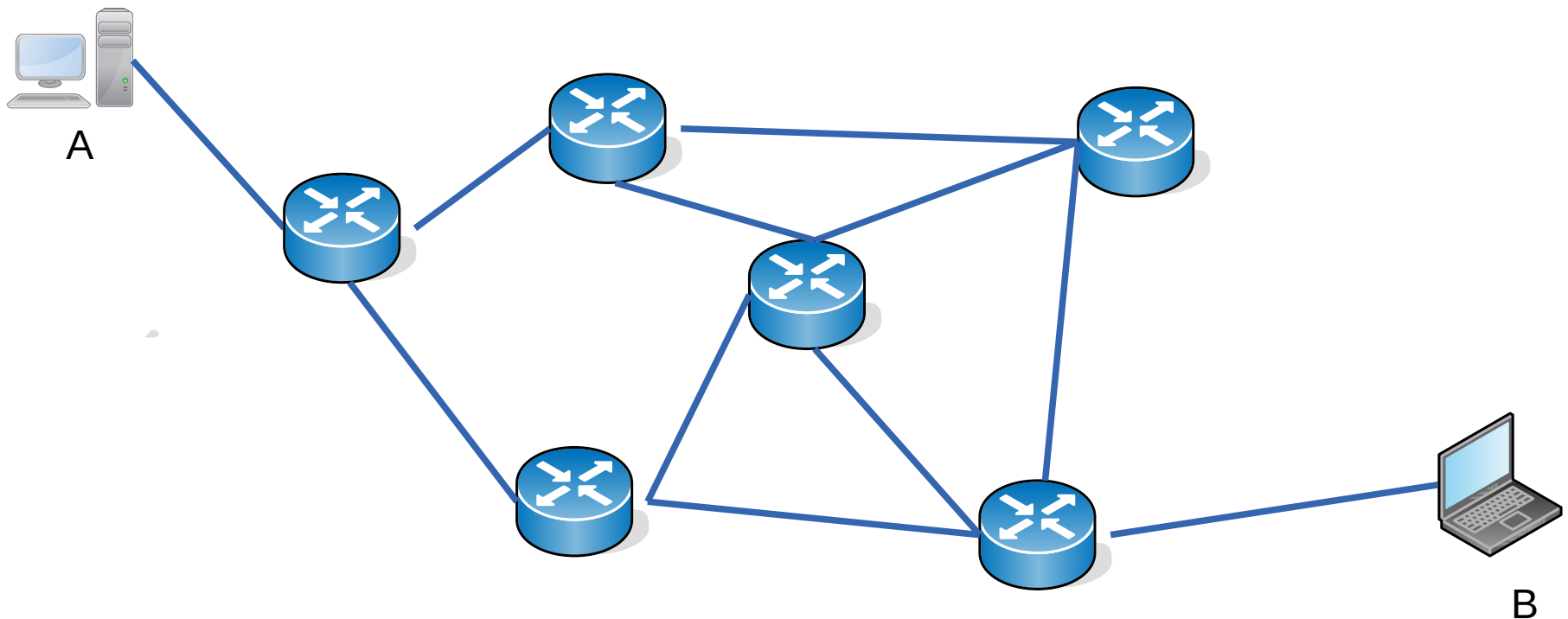
Plano de datos

- **Plano de Datos:** Función de **encaminamiento** (forwarding)
- Cada nodo **recibe** un paquete por una de sus **líneas de entrada** o interfaces, **decide en base a una tabla** cuál es la **línea de salida** o **próximo salto** más adecuado para ese destino y encamina el paquete hacia allí
- Esas tablas se llaman **tablas de forwarding**



Plano de datos

- **Plano de Datos:** Función de **encaminamiento** (forwarding)
- Cada nodo **recibe** un paquete por una de sus **líneas de entrada** o interfaces, **decide en base a una tabla** cuál es la **línea de salida** o **próximo salto** más adecuado para ese destino y encamina el paquete hacia allí
- Esas tablas se llaman **tablas de forwarding**



Encaminamiento generalizado

- La consulta a la tabla de forwarding de forma general consiste en:
 - 1) Buscar coincidencias (**match**) entre algún(os) campos del encabezado del paquete a encaminar y los campos indicados en las entradas de la tabla
 - Por ejemplo: campos del encabezado de capa 3, pero también de capa 4 y capa 2
 - 2) Si se encuentra una coincidencia, la tabla indicará las acciones (**action**) que hay que tomar
 - Por ejemplo: Encaminar el paquete por una salida, modificar campos de los encabezados (de varias capas), descartar el paquete, etc
- Este concepto de encaminamiento generalizado es básico en la arquitectura SDN
- En Internet actualmente se usa un caso particular que es el encaminamiento en base a la dirección de destino del paquete

Encaminamiento clásico básico

- El encaminamiento clásico es en base a la **dirección destino** del paquete
- La dirección de destino del paquete **se busca en la tabla de forwarding** y de esa tabla se obtiene la dirección del **próximo equipo o próximo salto** al que enviarle el paquete

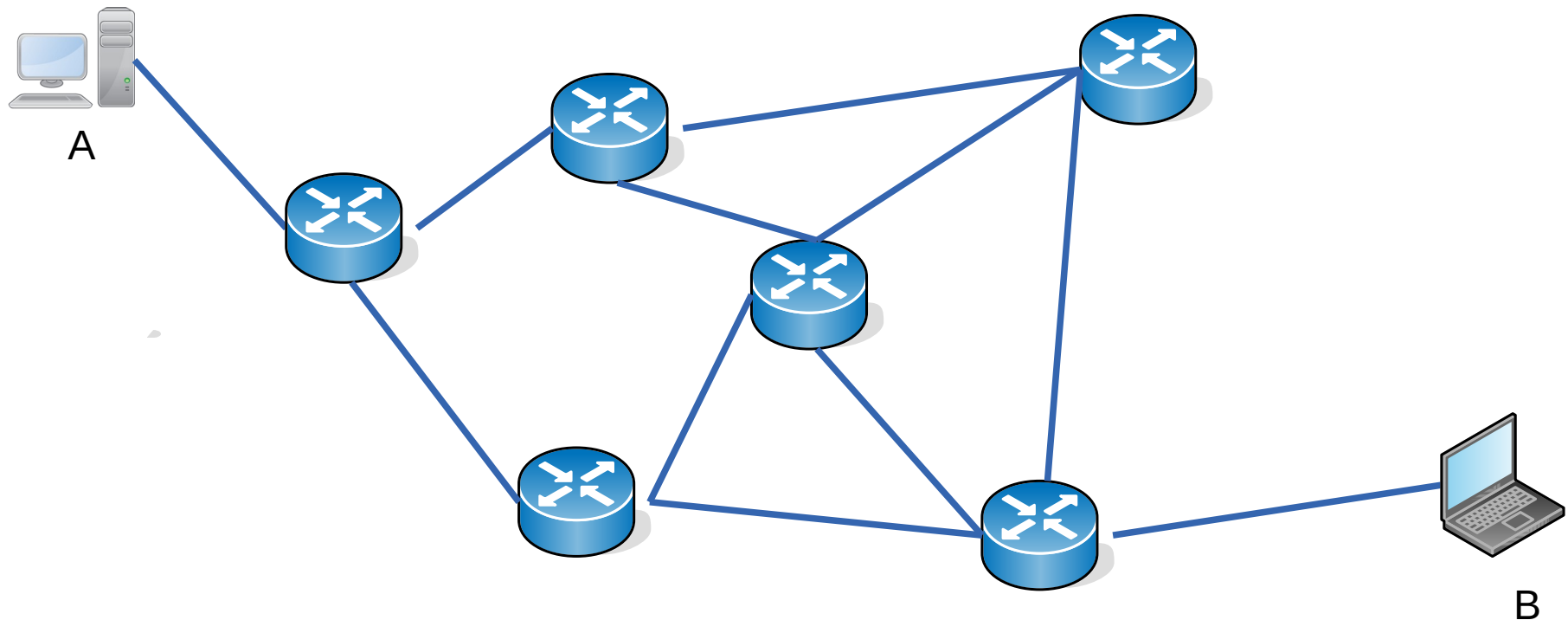
Encaminamiento clásico “avanzado”

- Además de la dirección destino del paquete, se pueden tener en cuenta otras características del paquete, como por ejemplo:
 - Dirección de origen
 - Tipo de servicio
 - Prioridad
- Usando esas otras características se puede por ejemplo:
 - Ofrecer servicios diferenciados a diferentes clientes
 - Calidad de servicio
 - Gestionar el tráfico para tener un mejor uso de los recursos de la red
 - Ingeniería de tráfico

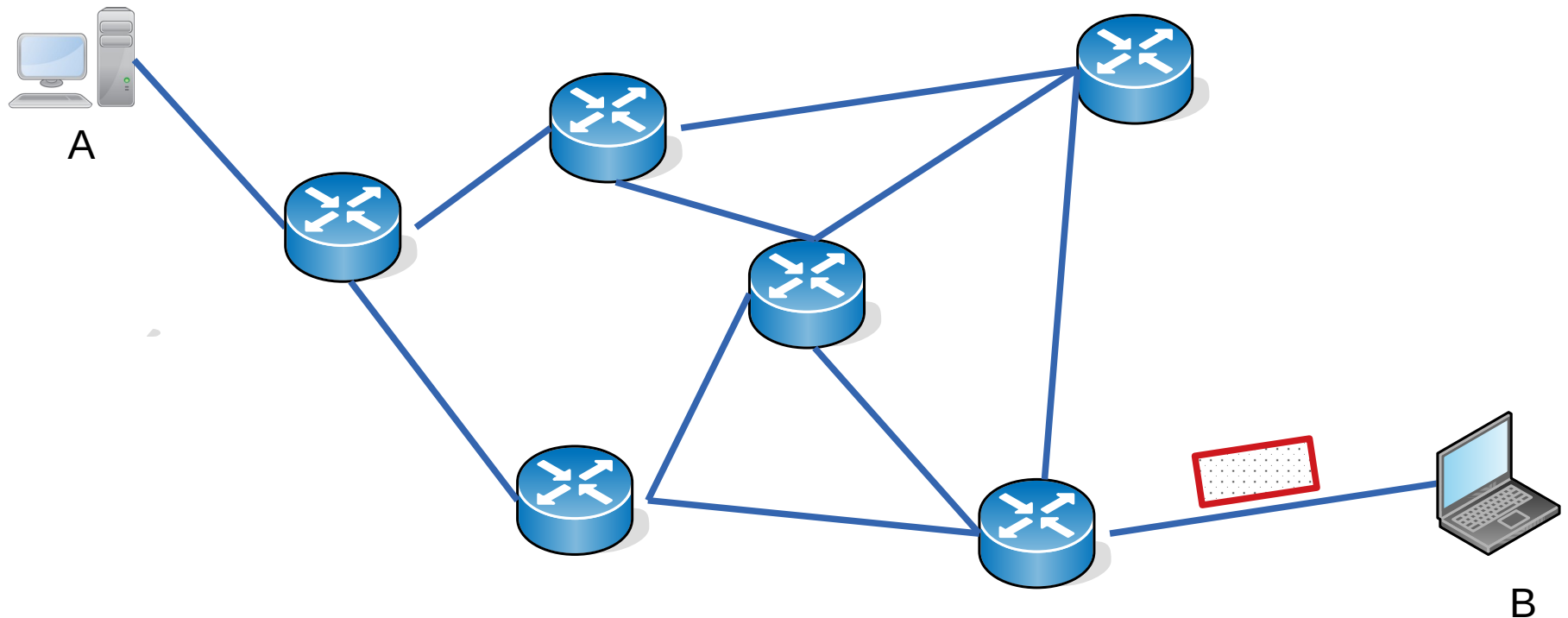
Plano de datos en Internet

- El servicio que brinda la **capa de red en Internet** es un servicio
 - **no orientado a conexión**
 - **no confiable**
- Se define a si mismo como un servicio “**best-effort**”
- **No hay garantía:**
 - Ni de entrega de los paquetes a destino
 - Ni del retardo de los paquetes
 - Ni del orden en que llegan los paquetes a destino
- Cada paquete se encamina **independientemente** de los anteriores
- Los paquetes o datagramas se definen en el Protocolo IP (Internet Protocol)
 - Hay dos versiones de IP en uso:
 - IP versión 4 o **IPv4** (especificado en la RFC 791)
 - IP versión 6 o **IPv6** (RFC 2460 y RFC 4291)
 - En este curso no se verá en detalle IPv6
 - Se espera que en el futuro IPv6 reemplace a IPv4

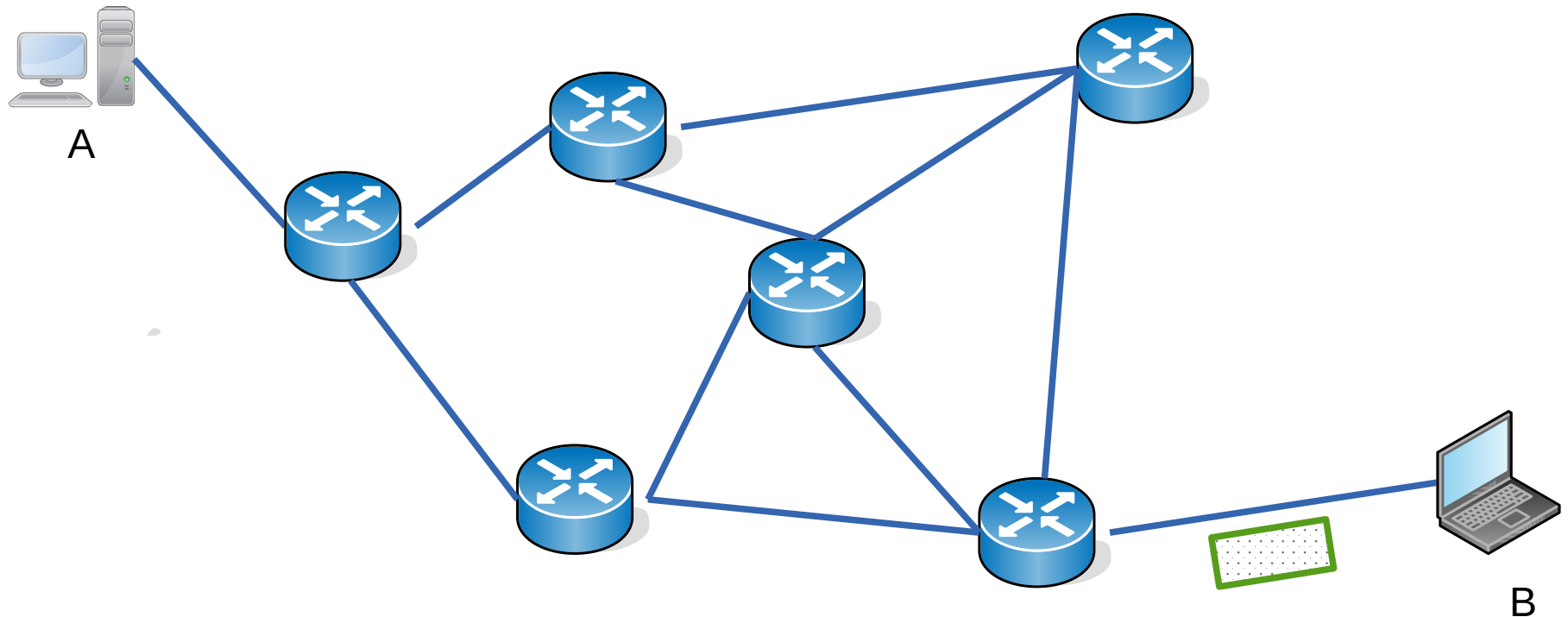
Encaminamiento paquete a paquete



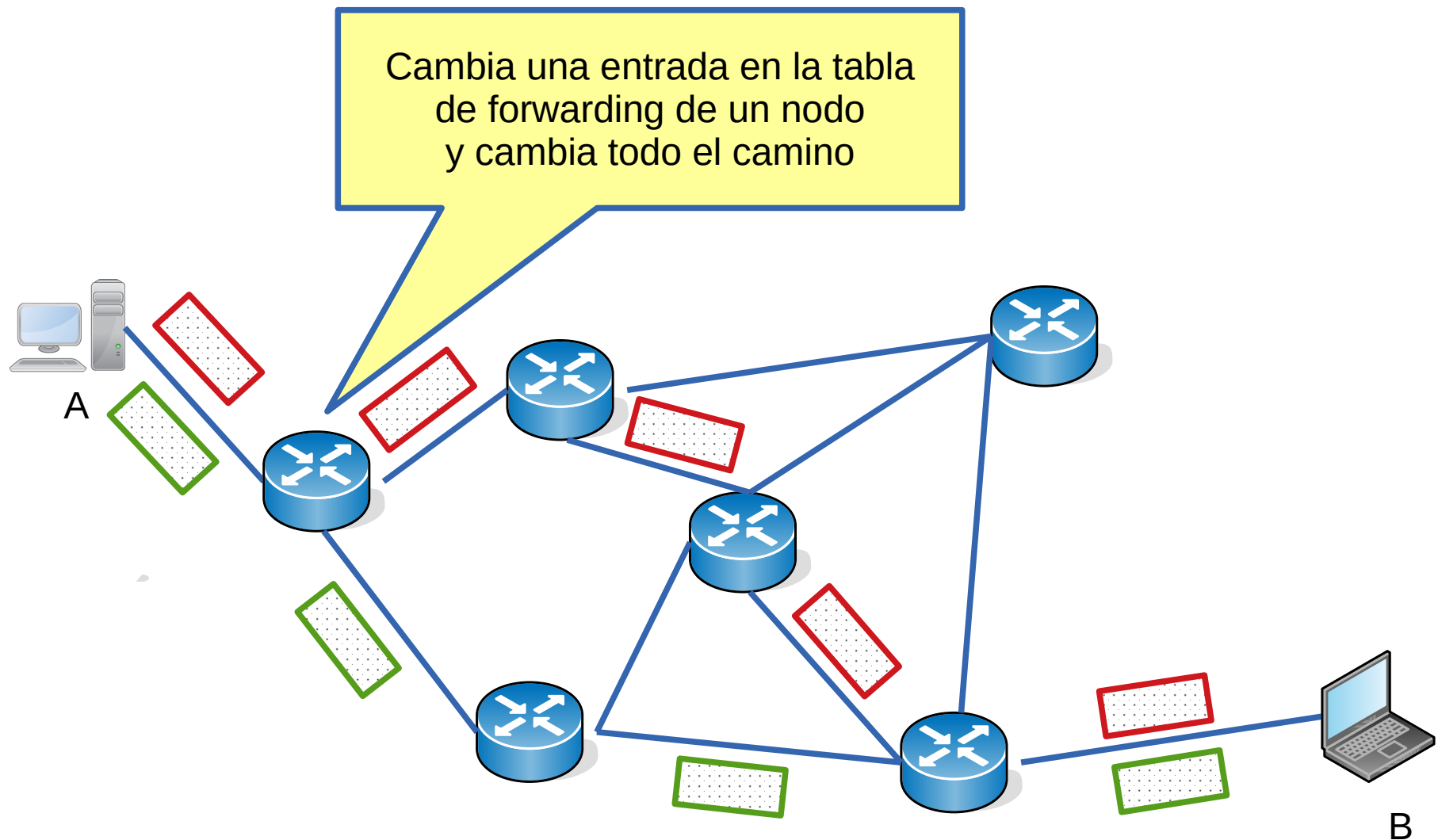
Encaminamiento paquete a paquete



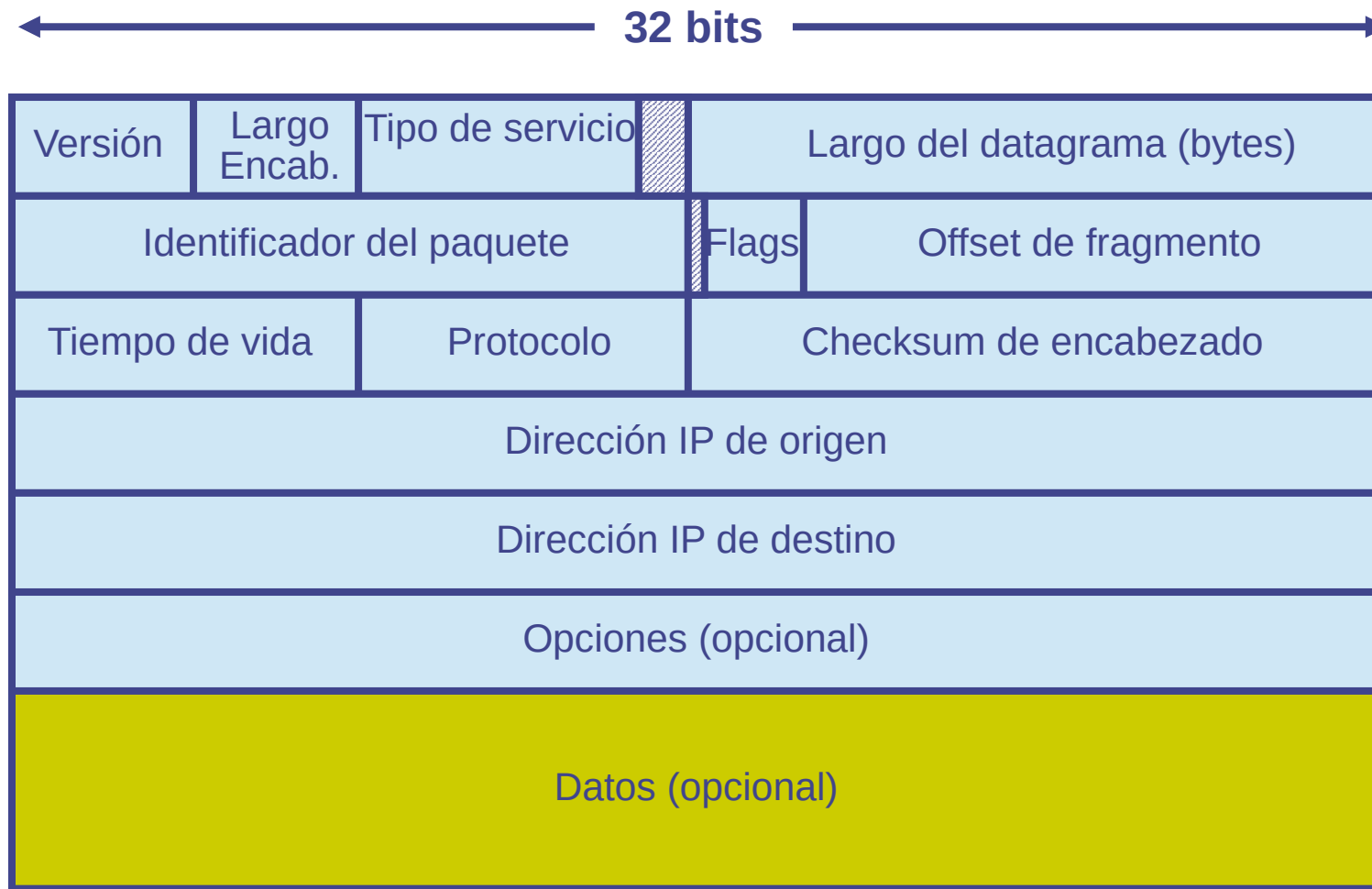
Encaminamiento paquete a paquete



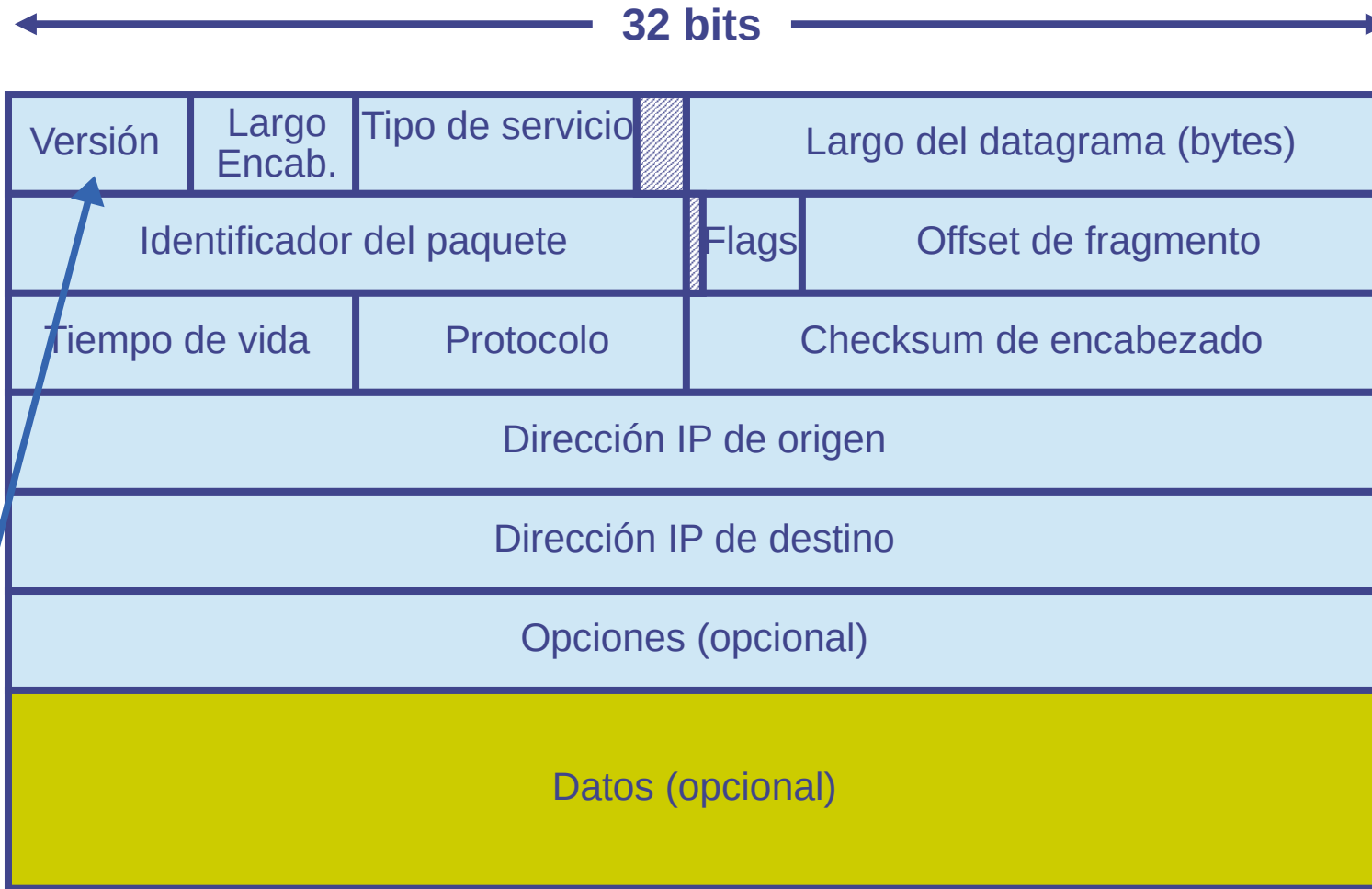
Encaminamiento paquete a paquete



Paquete o datagrama IPv4

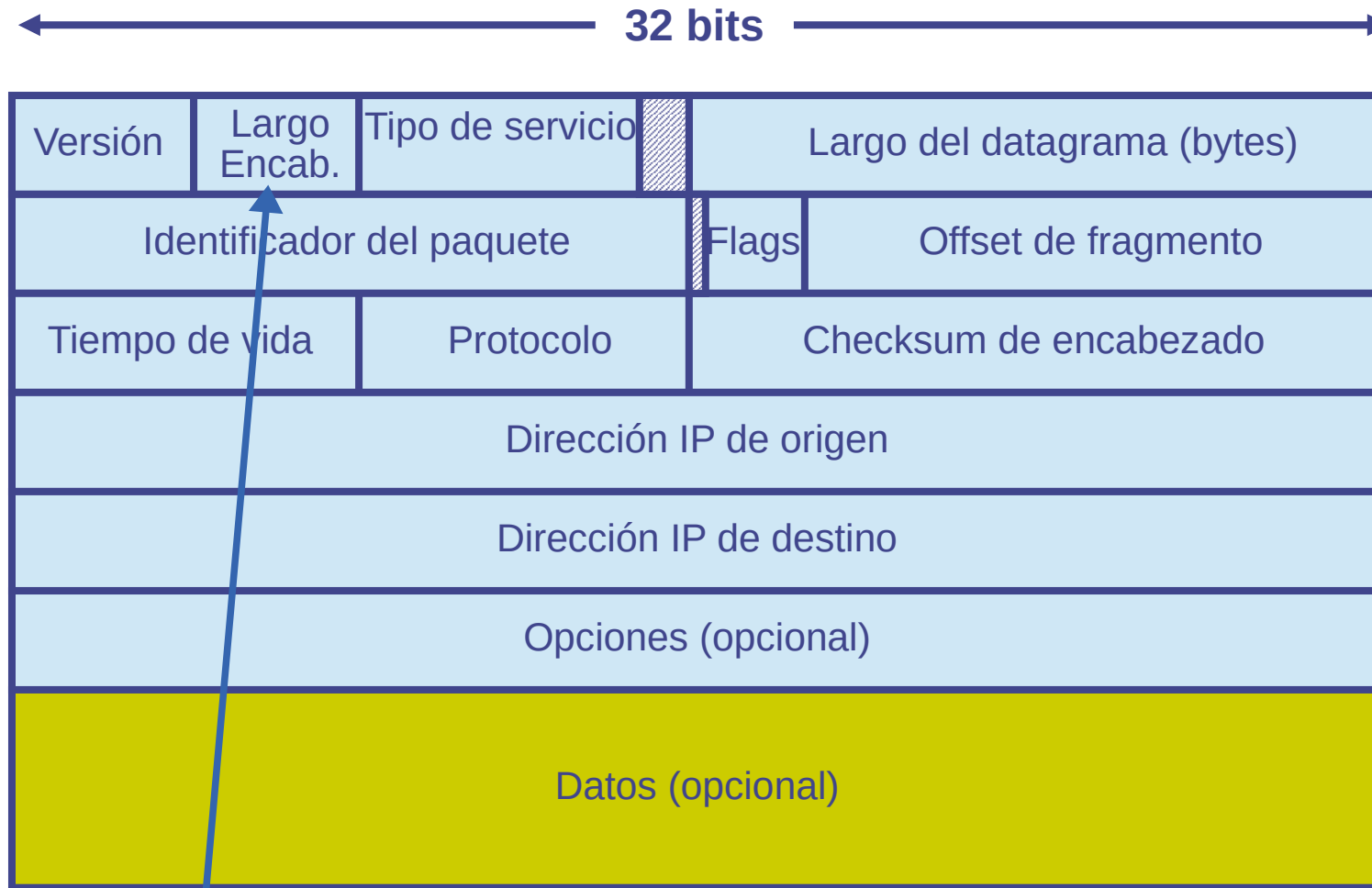


Paquete o datagrama IPv4



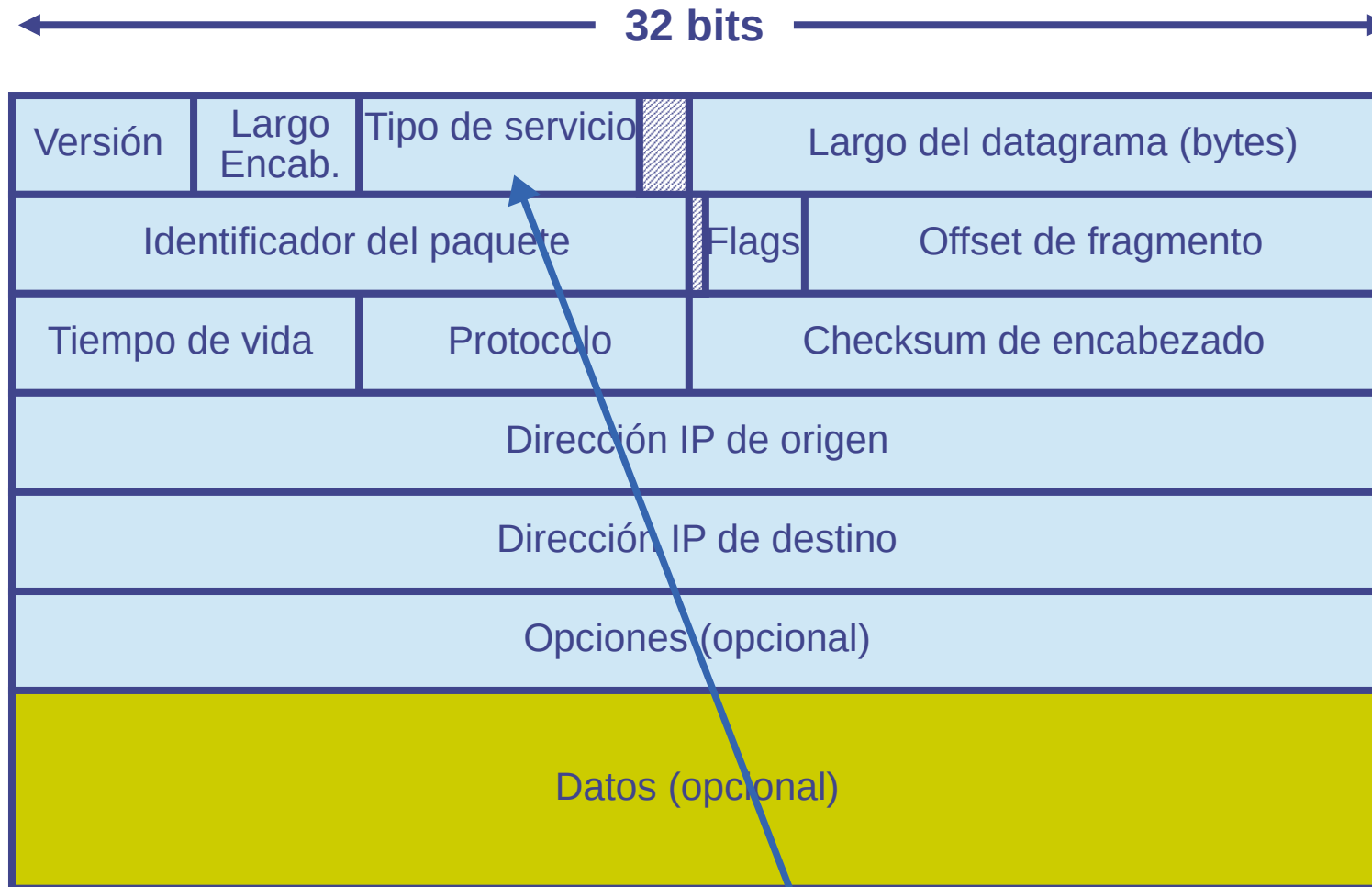
**Versión
(4 bits)**
IPv4 = 0100

Paquete o datagrama IPv4



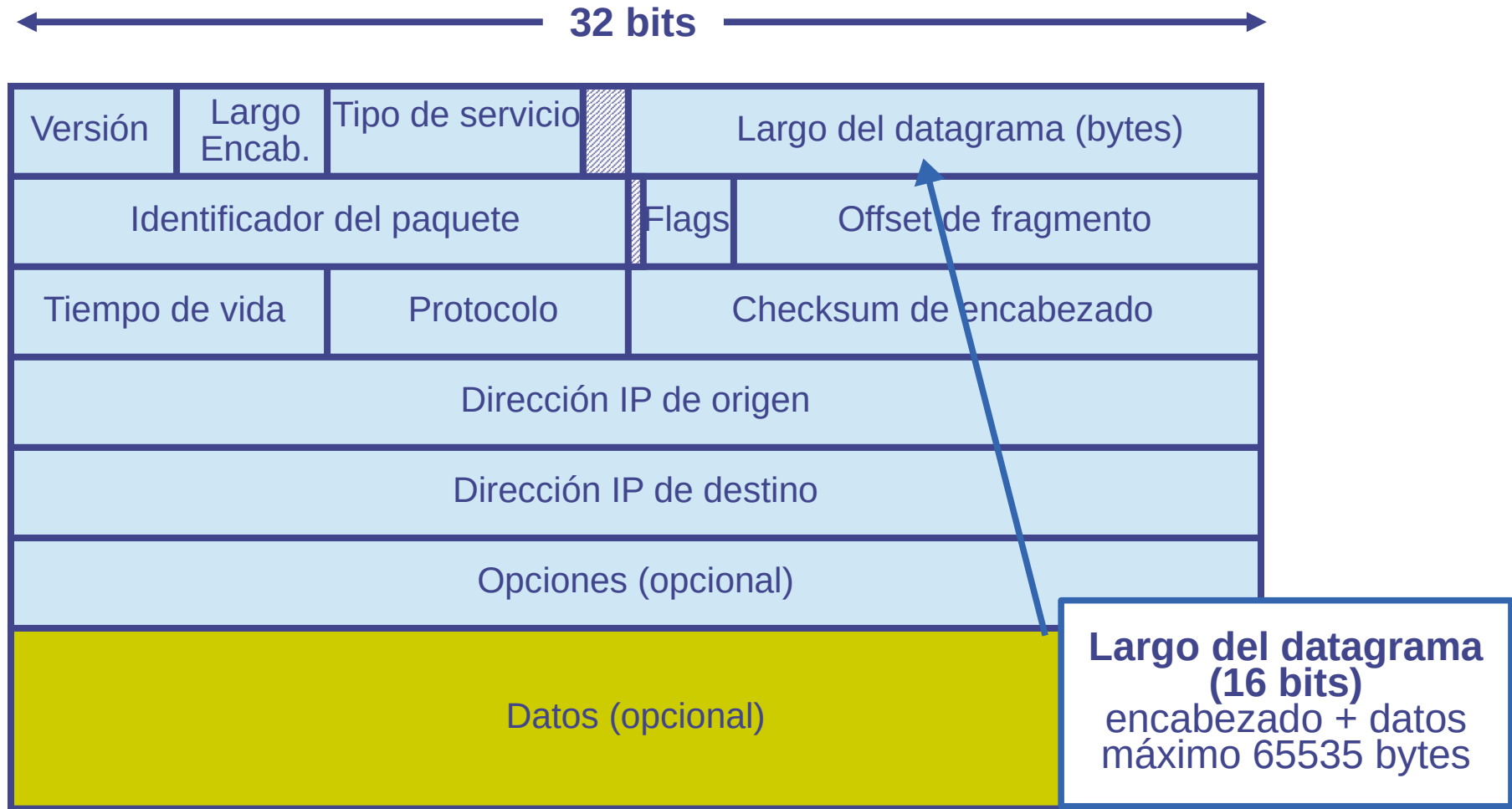
**Largo del encabezado
(4 bits)**
en palabras de 32 bits
Si no hay opciones es = 5

Paquete o datagrama IPv4

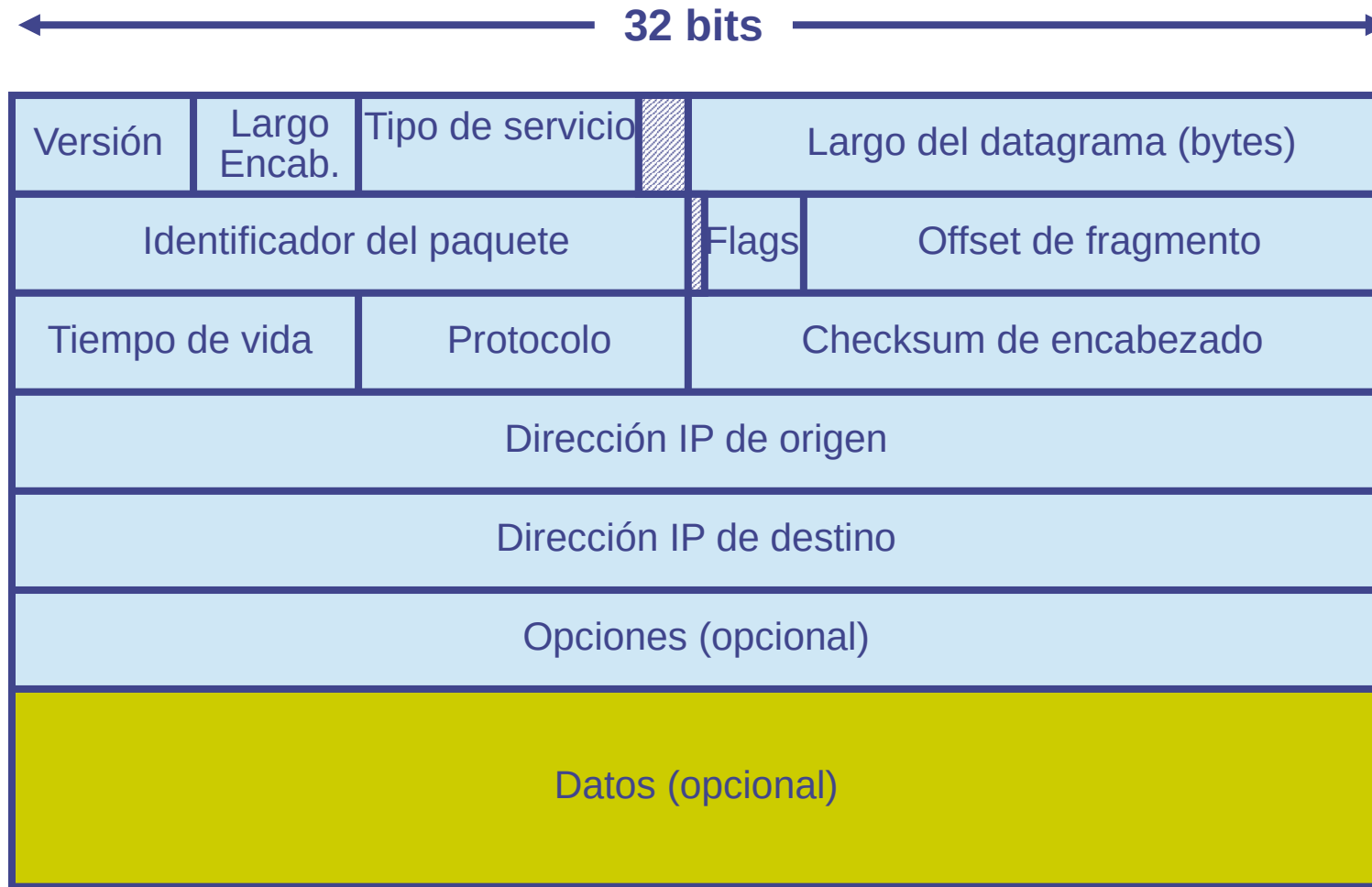


Tipo de servicio (6 bits)
Originalmente 3 bits de precedencia y 3 banderas Delay, Throughput, Reliability
Actualmente se usan para indicar la clase de servicio para ofrecer calidad diferenciada

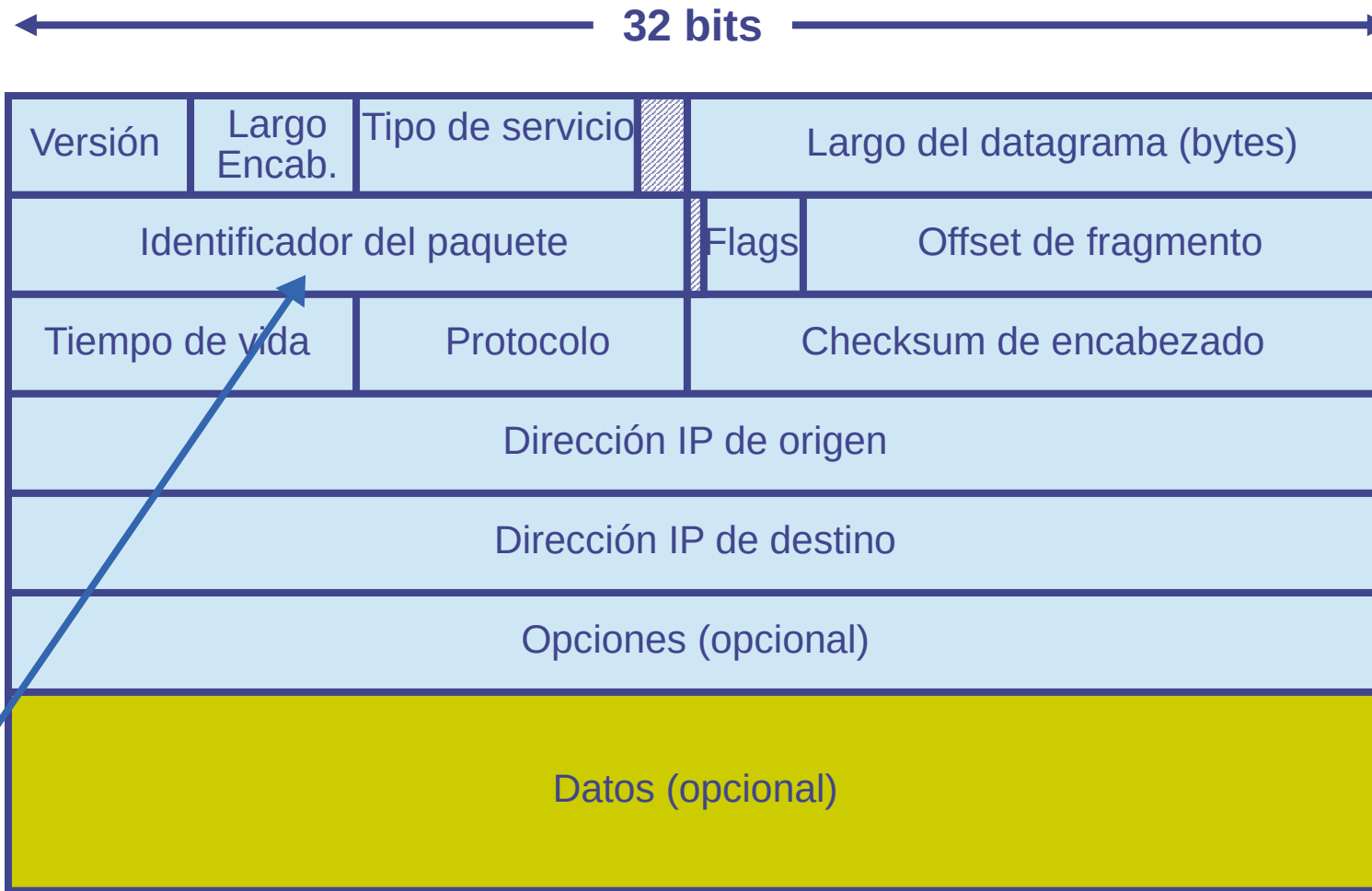
Paquete o datagrama IPv4



Paquete o datagrama IPv4

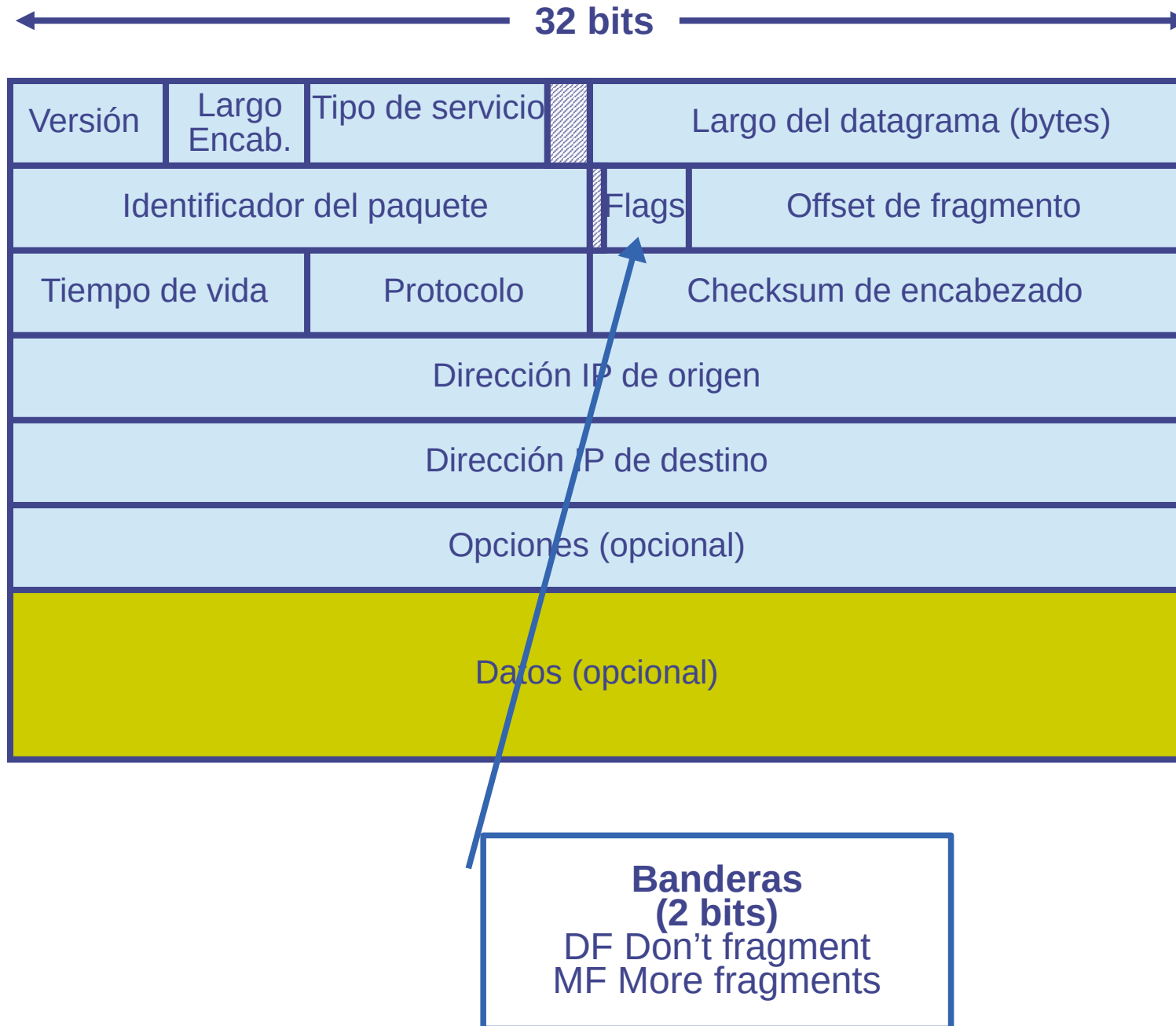


Paquete o datagrama IPv4

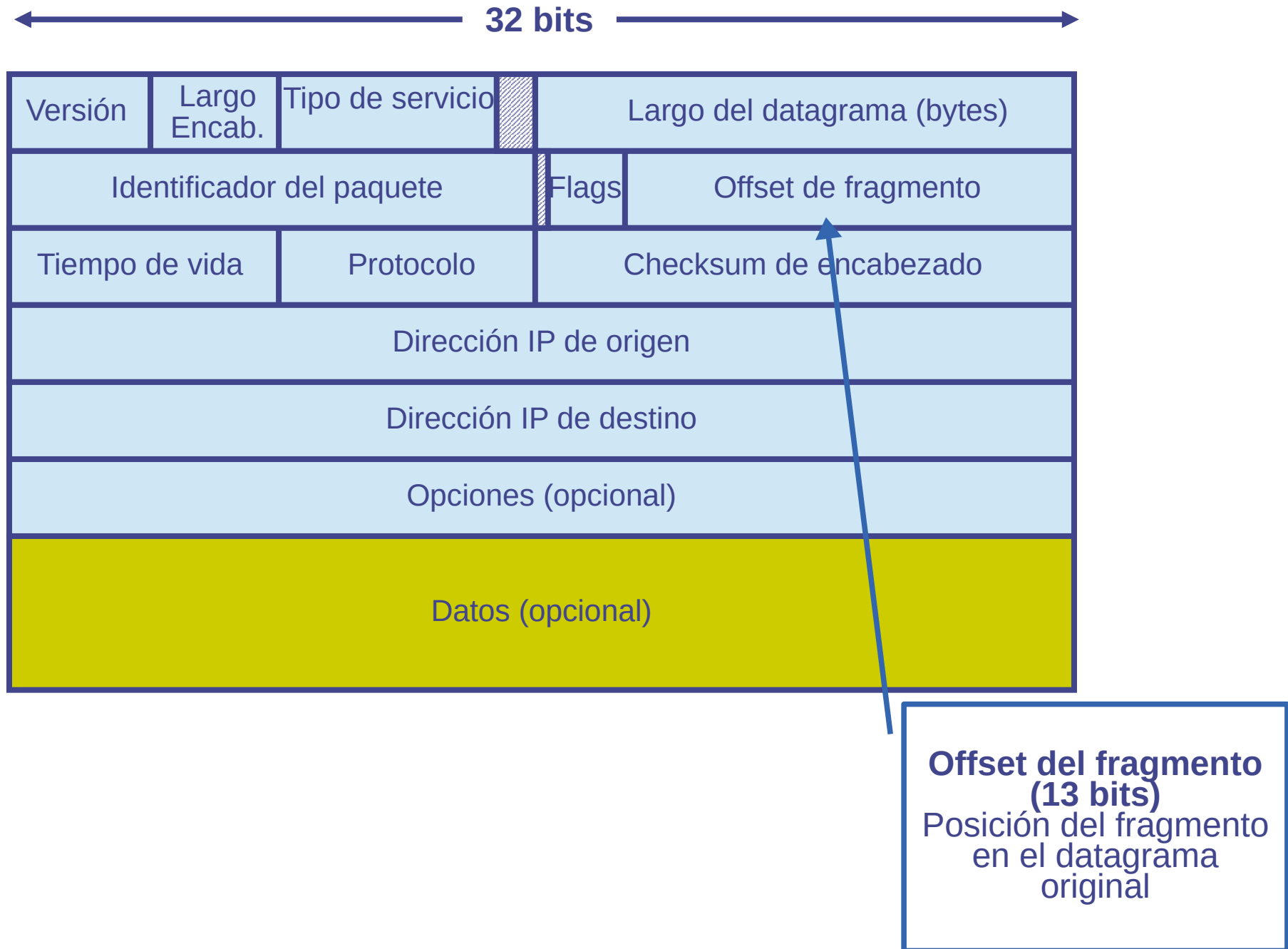


Identificación del paquete (16 bits)
se usa en la fragmentación
todos los fragmentos tienen el mismo identificador

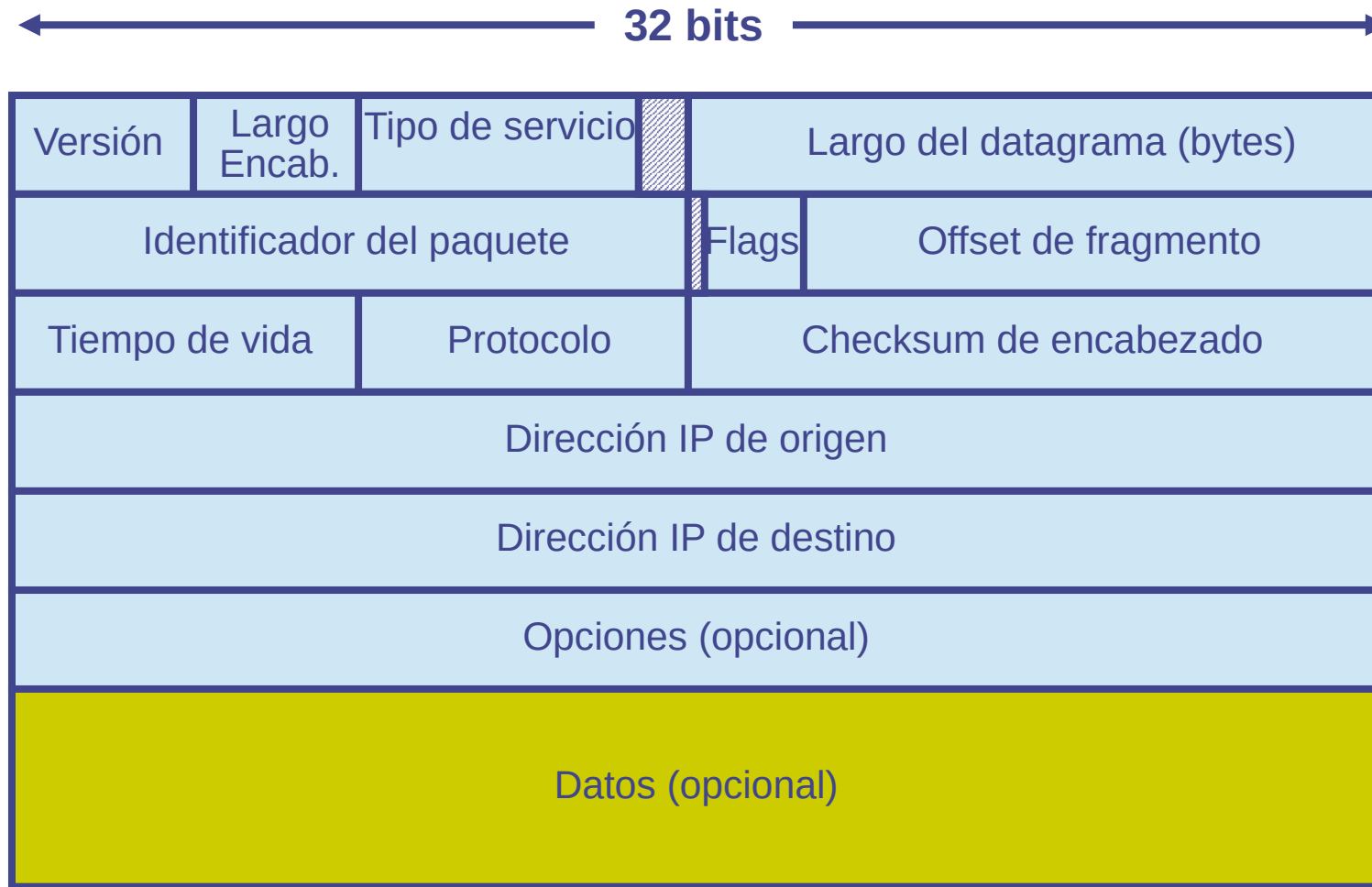
Paquete o datagrama IPv4



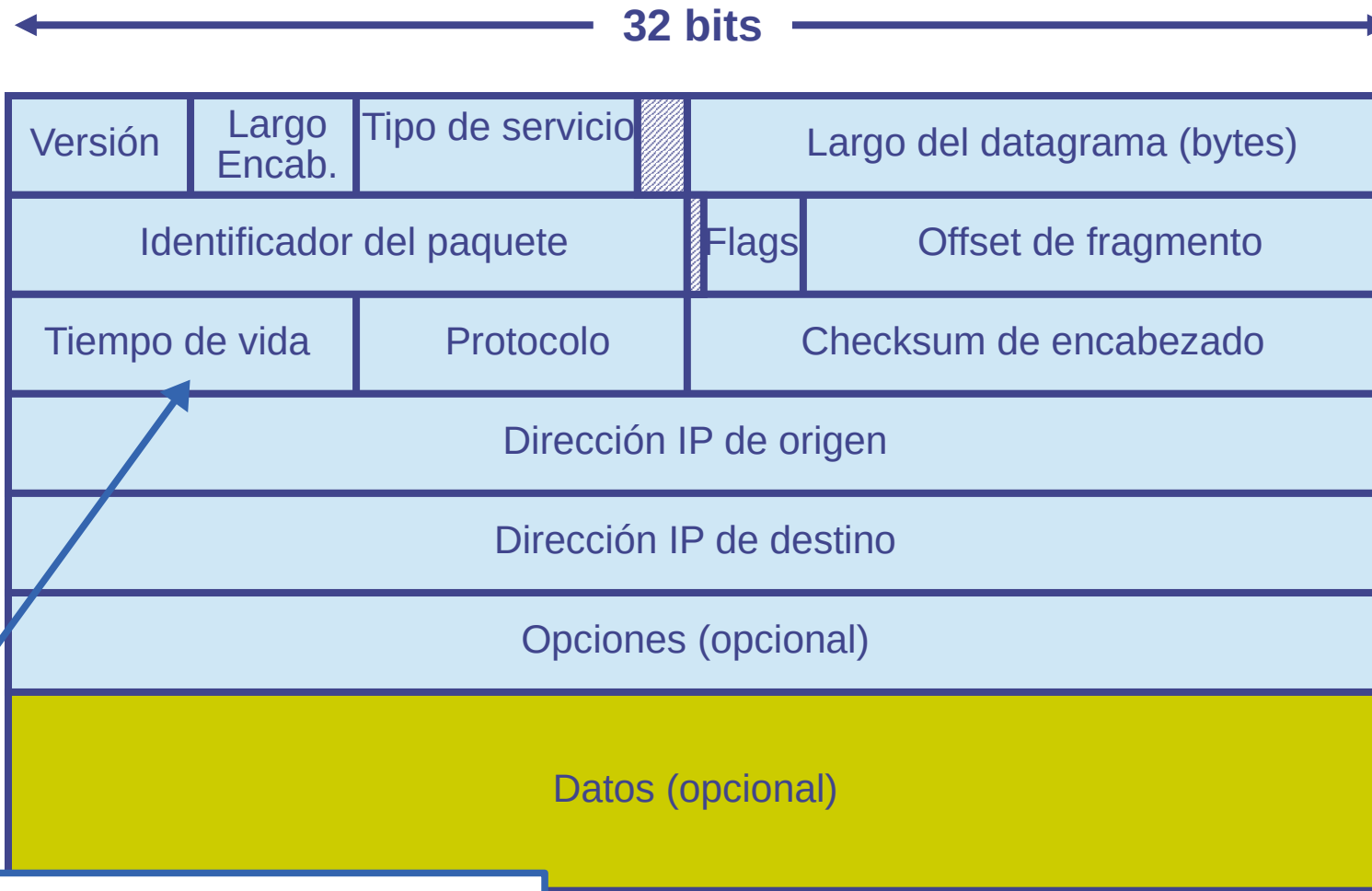
Paquete o datagrama IPv4



Paquete o datagrama IPv4



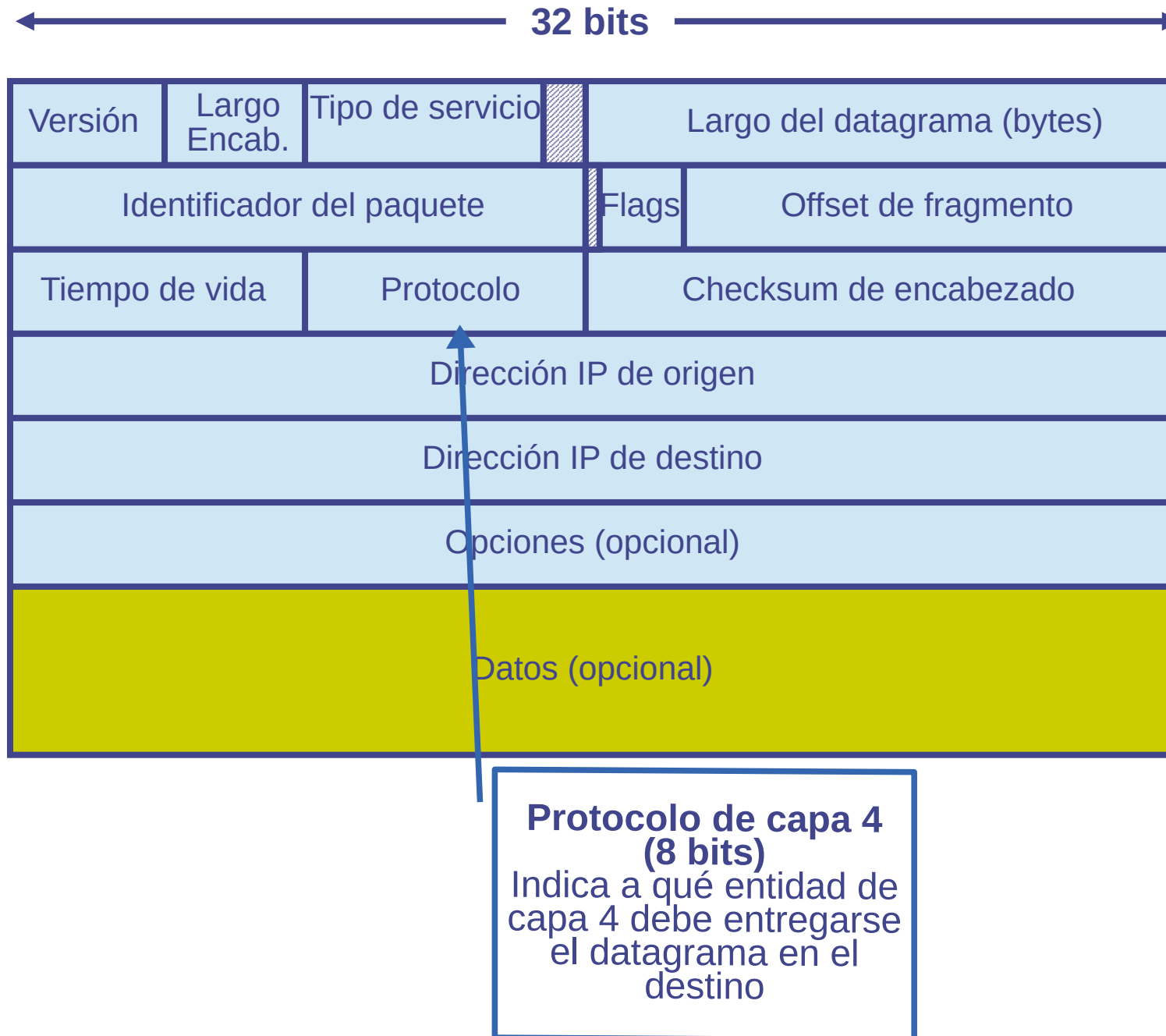
Paquete o datagrama IPv4



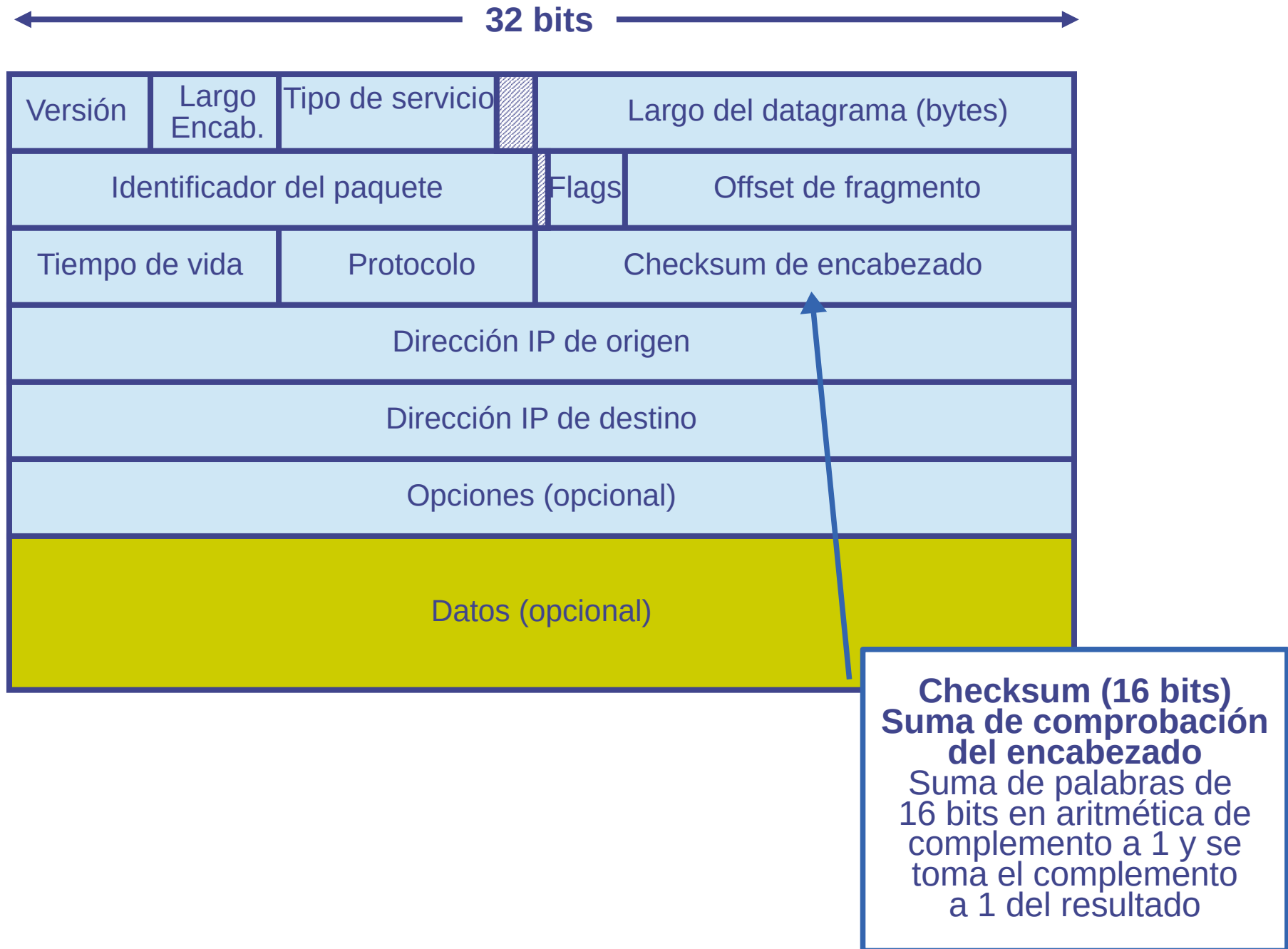
Tiempo de vida (8 bits) (time-to-live o TTL)

Contador de saltos para garantizar un máximo de vida de los paquetes
Cada router lo decrementa y cuando le da 0 lo debe descartar

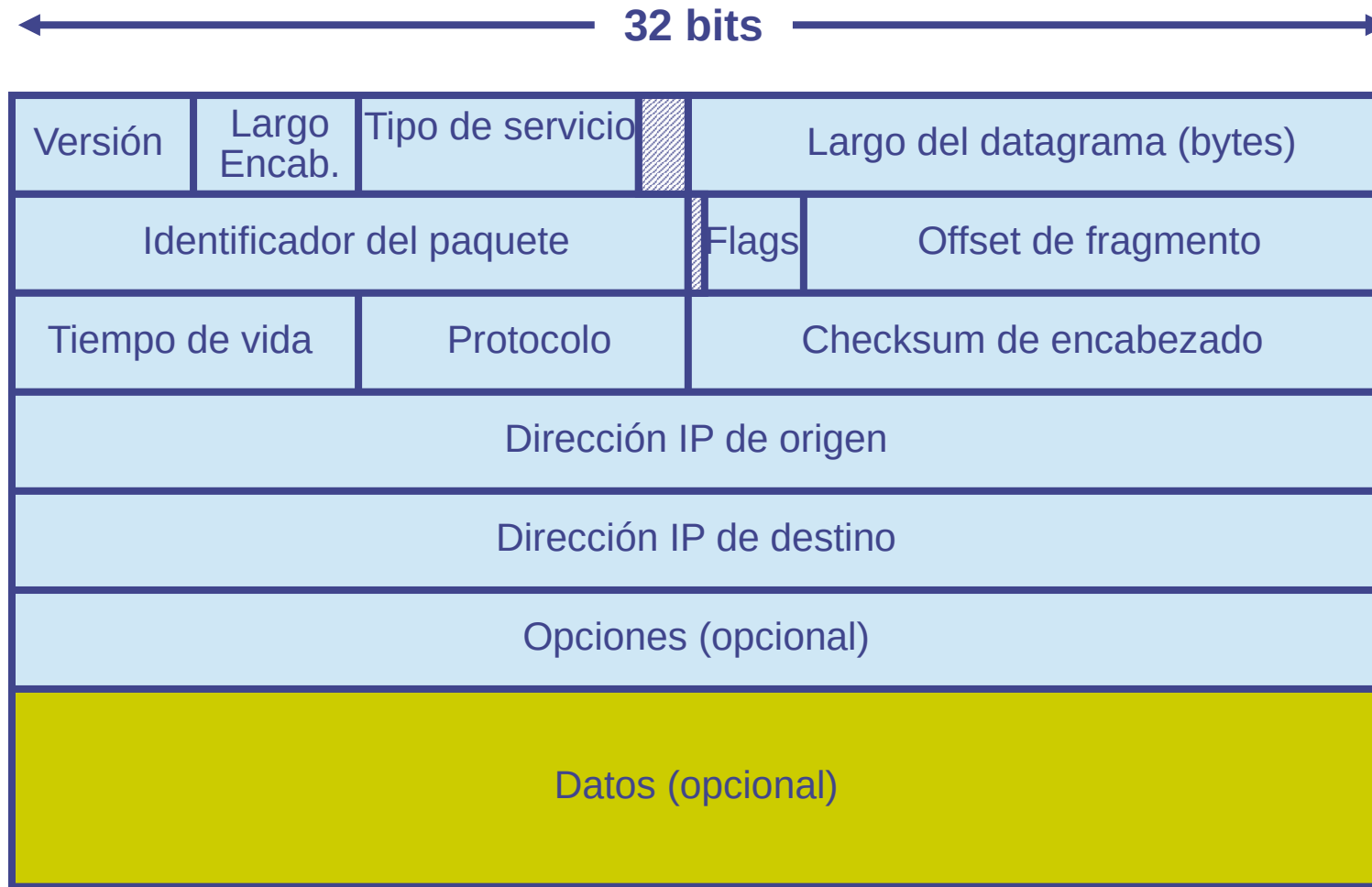
Paquete o datagrama IPv4



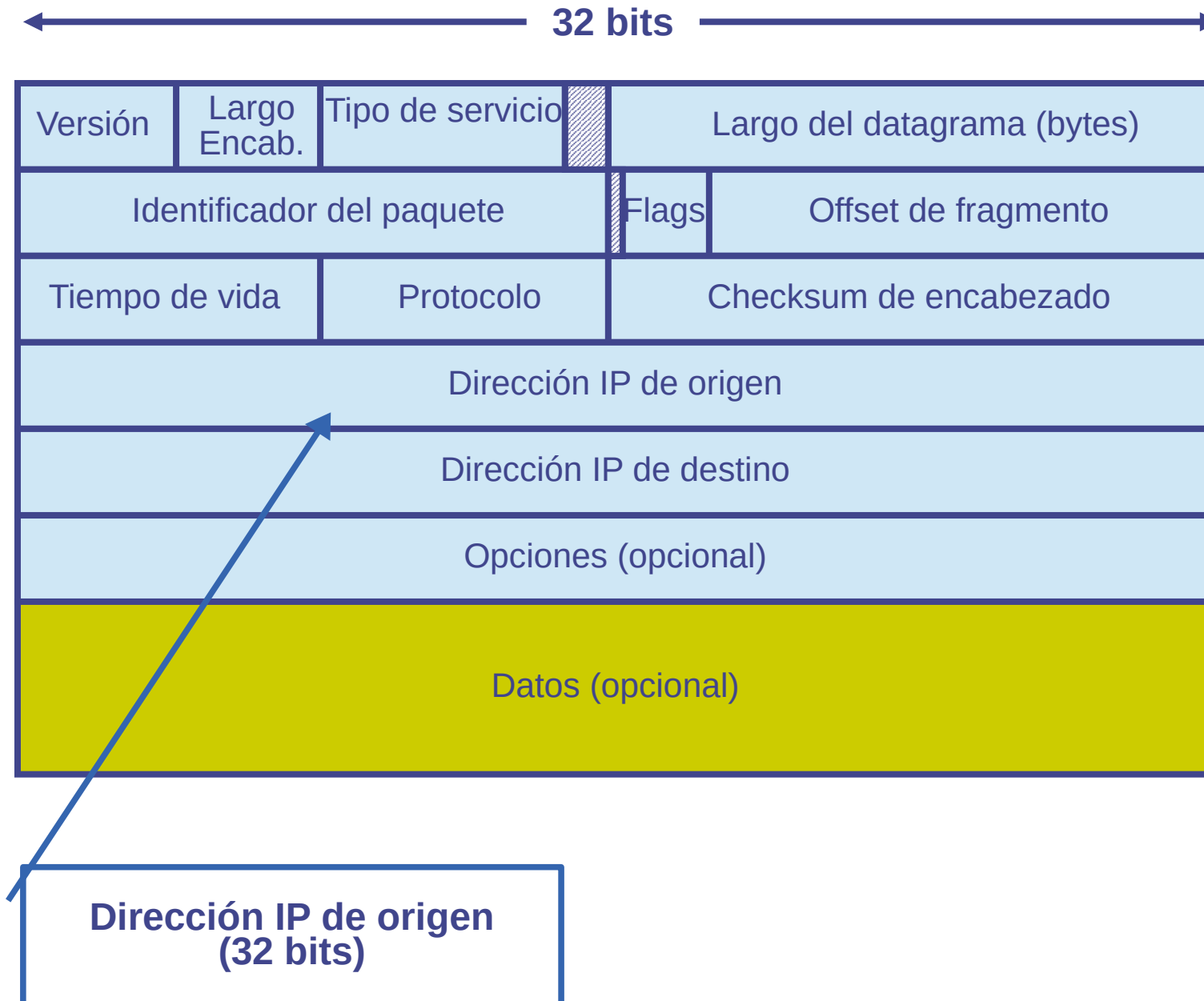
Paquete o datagrama IPv4



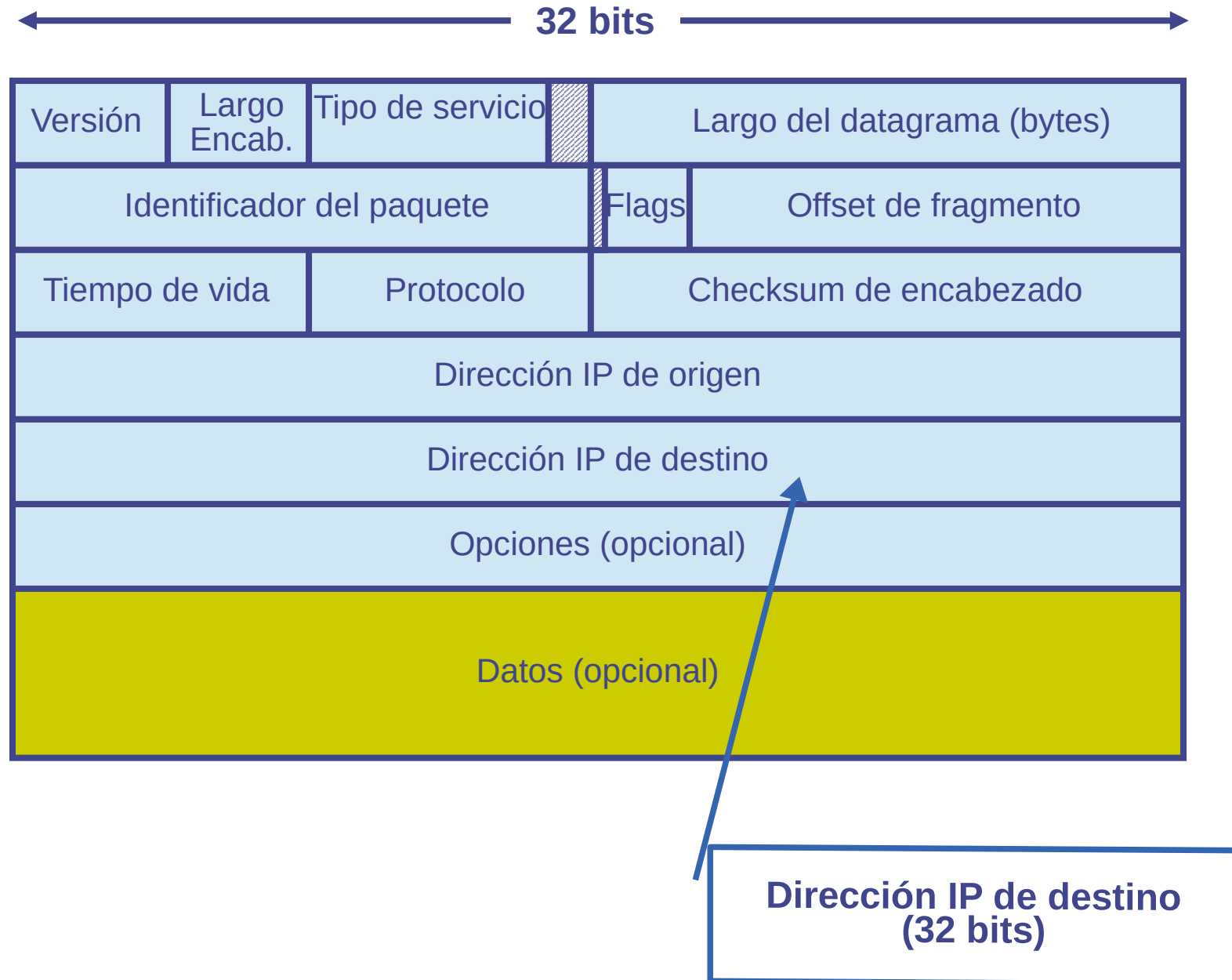
Paquete o datagrama IPv4



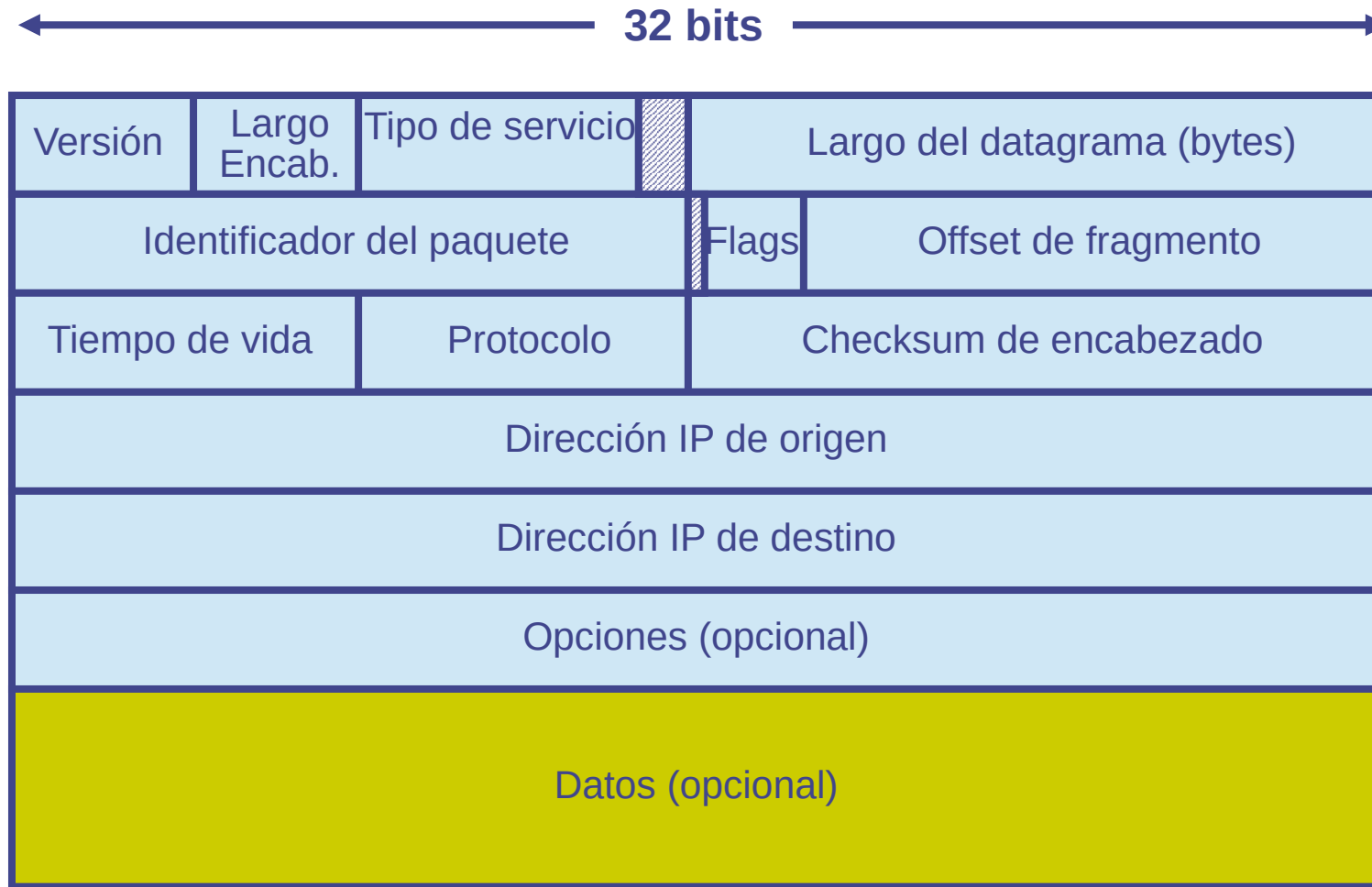
Paquete o datagrama IPv4



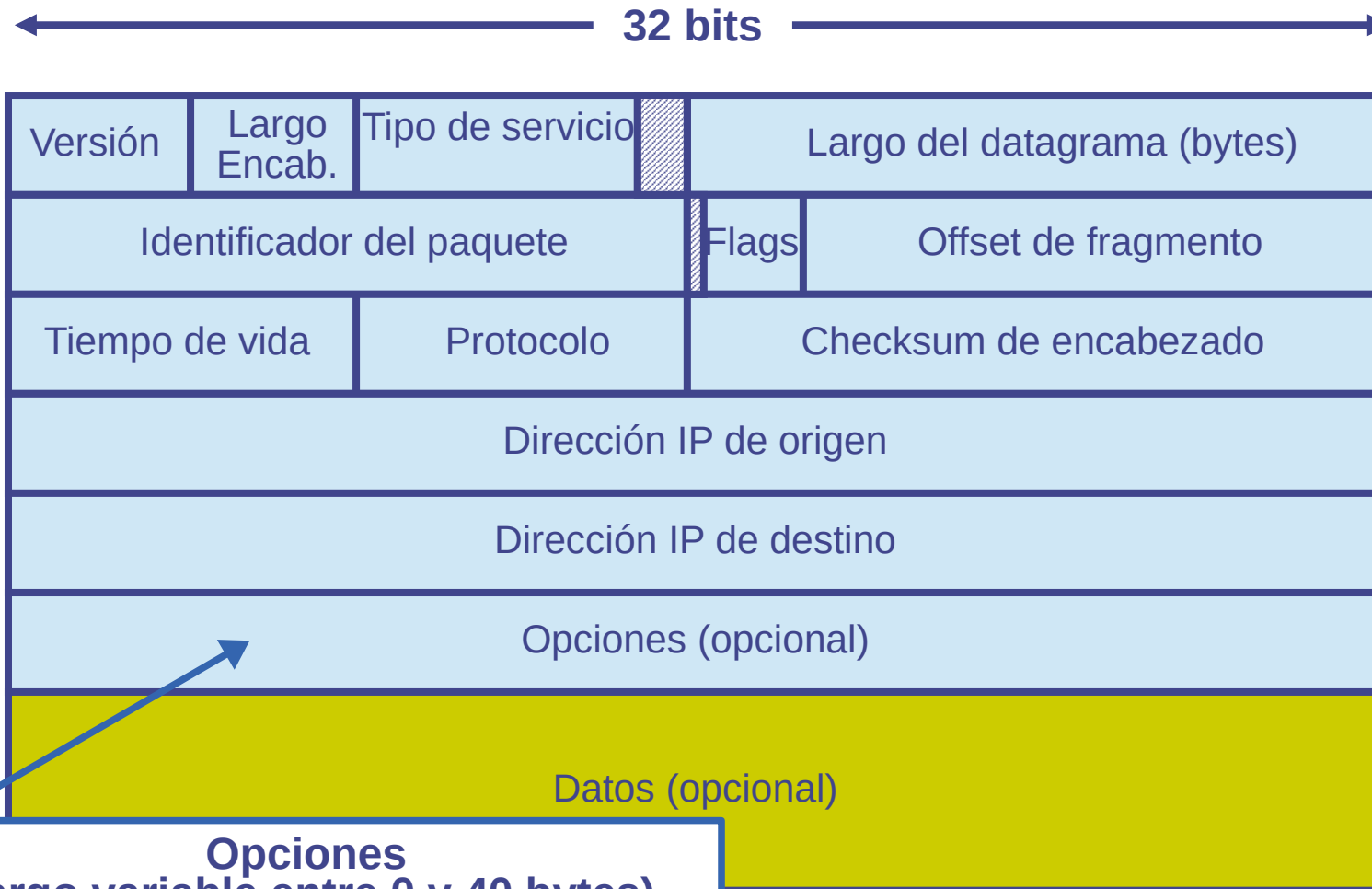
Paquete o datagrama IPv4



Paquete o datagrama IPv4



Paquete o datagrama IPv4

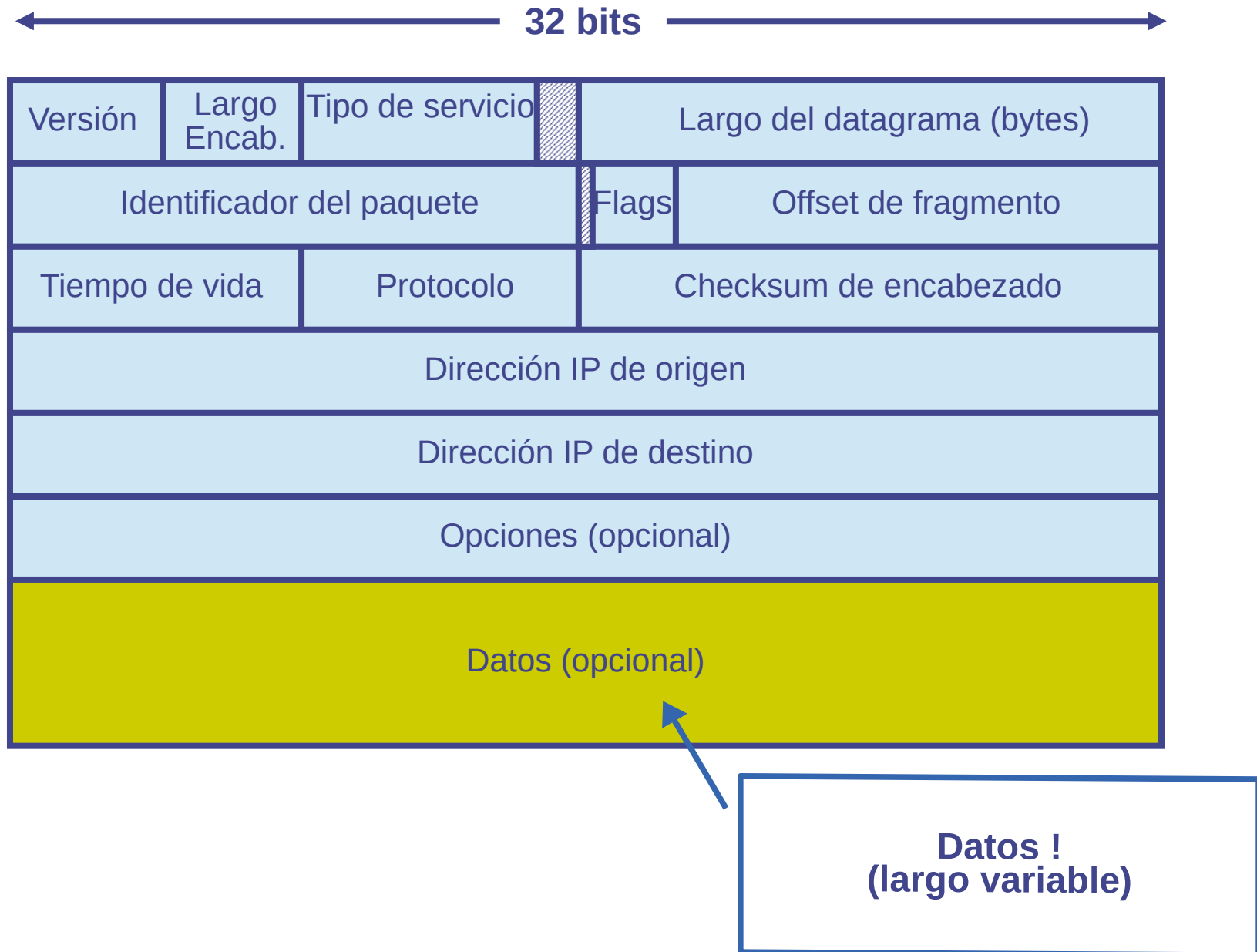


Opciones
(largo variable entre 0 y 40 bytes)

Ejemplos:

Security (SEC, E-SEC, CIPSO)
Strict source routing (SSR)
Loose source routing (LSR)
Record route (RR)
Timestamp (TS)
....

Paquete o datagrama IPv4



Paquete IPv4

- **Versión:** 4 bits
 - Versión del protocolo IP
 - El enrutador puede mirar esos primeros 4 bits del datagrama y saber cómo tratar el resto del paquete
- **Largo del encabezado:** 4 bits
 - Tamaño del encabezado expresado en palabras de 32 bits
 - Es necesario porque el encabezado tiene tamaño variable en caso que se use el campo de opciones
 - Si el paquete no tiene opciones (caso típico) el largo del encabezado es de **20 bytes** y el campo tendrá el valor 5
- **Tipo de servicio:** 8 bits (en general ignorado en Internet)
 - Especificación original:
 - Precedencia (3 bits) (Prioridad 0..7)
 - DTR (Delay, Throughput, Reliability)
 - 2 bits no usados
 - “Nuevo” uso: DSCP (6 bits, 2 no usados) (RFC 2474) (hay más propuestas)

Paquete IPv4

- **Largo del datagrama:** 16 bits
 - Largo total incluyendo encabezado y datos, medido en bytes
 - Máximo teórico 65535 bytes
 - Normalmente 1500 bytes por ser la carga útil del protocolo de capa 2 más usado (Ethernet)
- **Identificador del paquete** (16 bits), **Banderas** (2 bits) (DF y MF) y **Offset de fragmento** (13 bits)
 - Campos usados para la **fragmentación** de paquetes
- **Tiempo de vida** (Time-to-live, TTL): 8 bits
 - Es para limitar el tiempo de vida de los paquetes en la red
 - Se implementa como un contador de saltos
 - Cada enrutador por el que pasa el paquete debe decrementar el TTL en 1 y si llega a 0 descartar el paquete
- **Protocolo:** 8 bits
 - Protocolo de capa superior del contenido del paquete: TCP, UDP, otros

Paquete IPv4

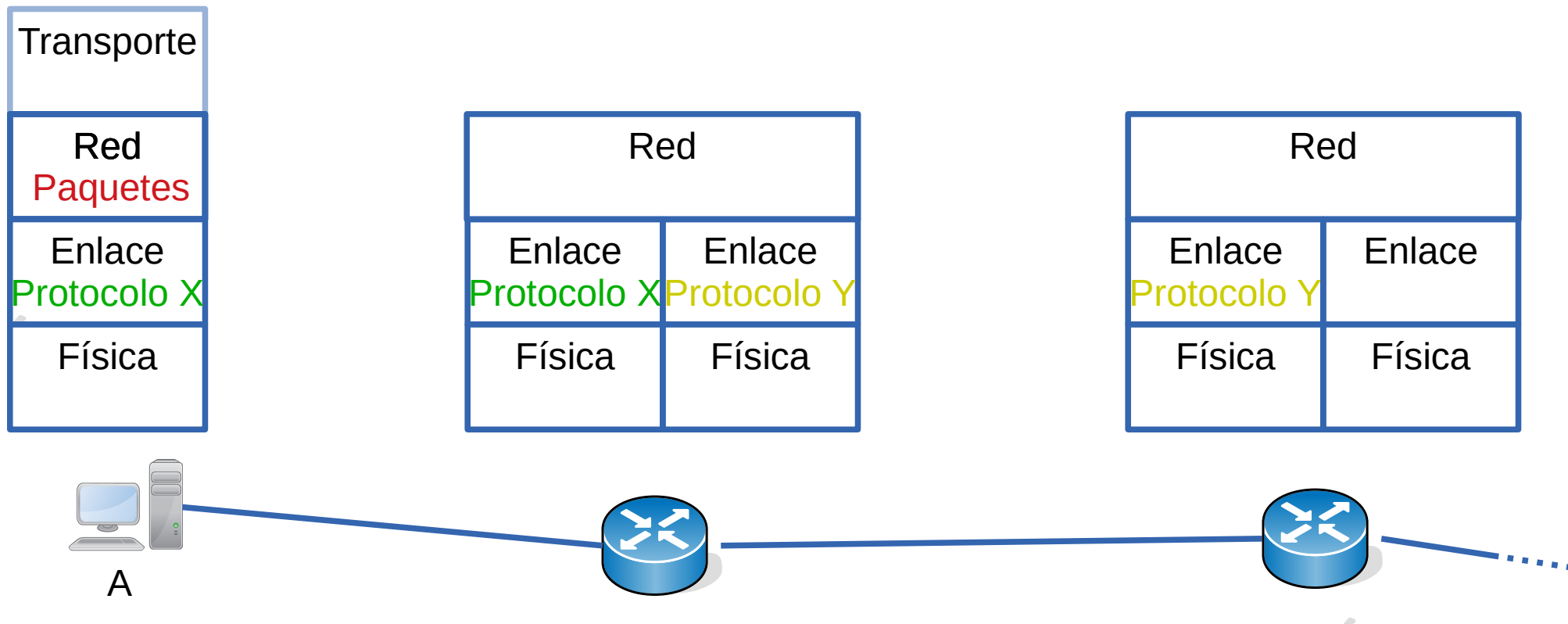
- **Checksum** del encabezado: 16 bits
 - Sirve para control de errores del encabezado
 - No incluye el campo de datos del paquete! (a diferencia de TCP/UDP)
 - Forma de cálculo:
 - en el transmisor:
 - Se inicializa el campo checksum con 0's
 - Se suman las palabras de 16 bits del encabezado usando aritmética de complemento a 1
 - Se toma el complemento a 1 de ese resultado
 - Este resultado se usa como valor del campo checksum
 - En recepción se calcula el checksum con la misma regla y si no hay errores el resultado es todos 1's
 - En cada enrutador por el que pasa el paquete hay que recalcular el checksum, ya que al menos el campo TTL cambia en cada salto
- **Dirección IP de origen**: 32 bits / **Dirección IP de destino**: 32 bits
 - Identifican el equipo origen y destino de cada paquete

Paquete IPv4

- **Opciones:** largo variable según los parámetros de cada opción
 - Permiten extender funcionalidades:
 - Hay varias definidas:
 - Enrutamiento desde el origen (source routing)
 - Registro de enrutadores por los que pasa el paquete (record route)
 - etc
 - Hacen que el procesamiento de los paquetes sea más lento (procesamiento por software)
 - Actualmente tienen uso casi nulo en Internet
 - A veces los paquetes con opciones son descartados por algunos equipos intermedios (por consideraciones de seguridad)
- **Datos:** largo variable
 - Contenido o carga útil del paquete (**payload**)
 - Típicamente el contenido de un segmento TCP o UDP, pero hay otros protocolos

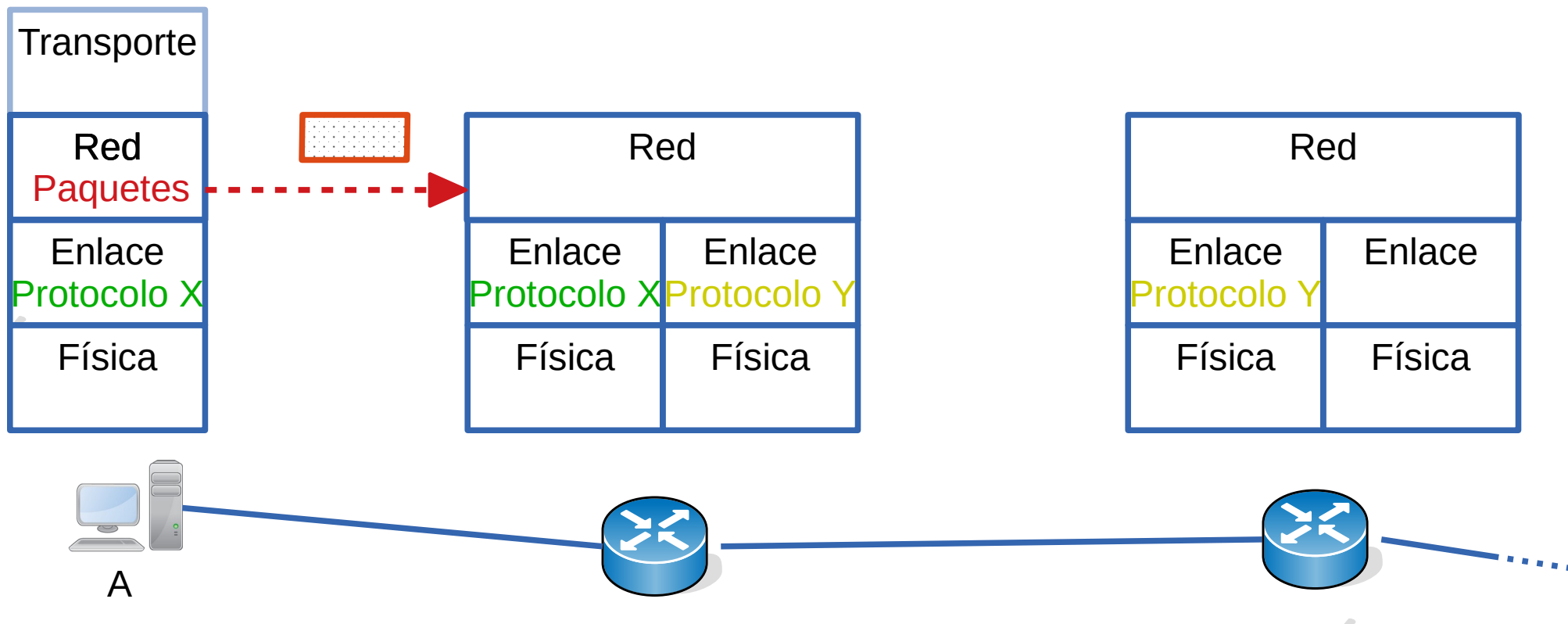
Fragmentación

- En cada salto, los paquetes viajan como carga útil del protocolo de capa de enlace
- Cada enlace podrá tener diferente tecnología y por tanto diferente protocolo de capa de enlace, con su correspondiente formato de trama
- En particular la máxima capacidad de carga útil de la capa de enlace se conoce como **MTU** (Maximum Transmission Unit)



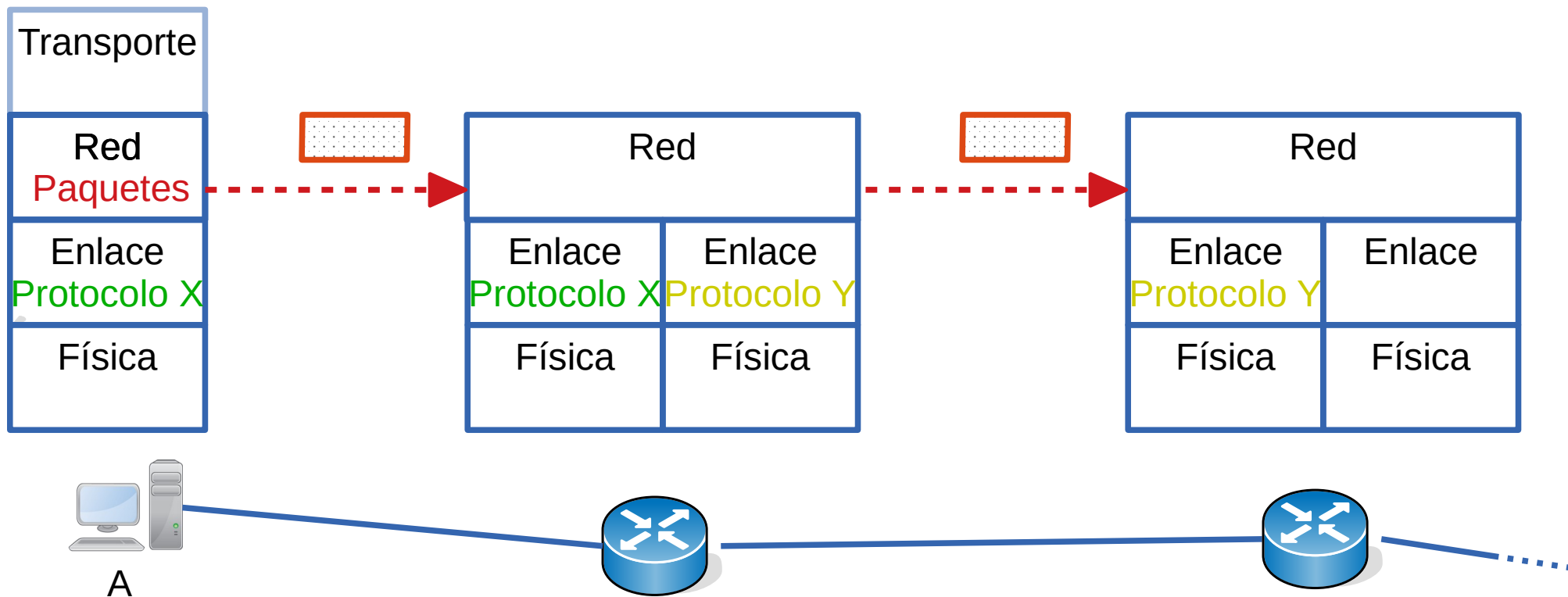
Fragmentación

- En cada salto, los paquetes viajan como carga útil del protocolo de capa de enlace
- Cada enlace podrá tener diferente tecnología y por tanto diferente protocolo de capa de enlace, con su correspondiente formato de trama
- En particular la máxima capacidad de carga útil de la capa de enlace se conoce como **MTU** (Maximum Transmission Unit)



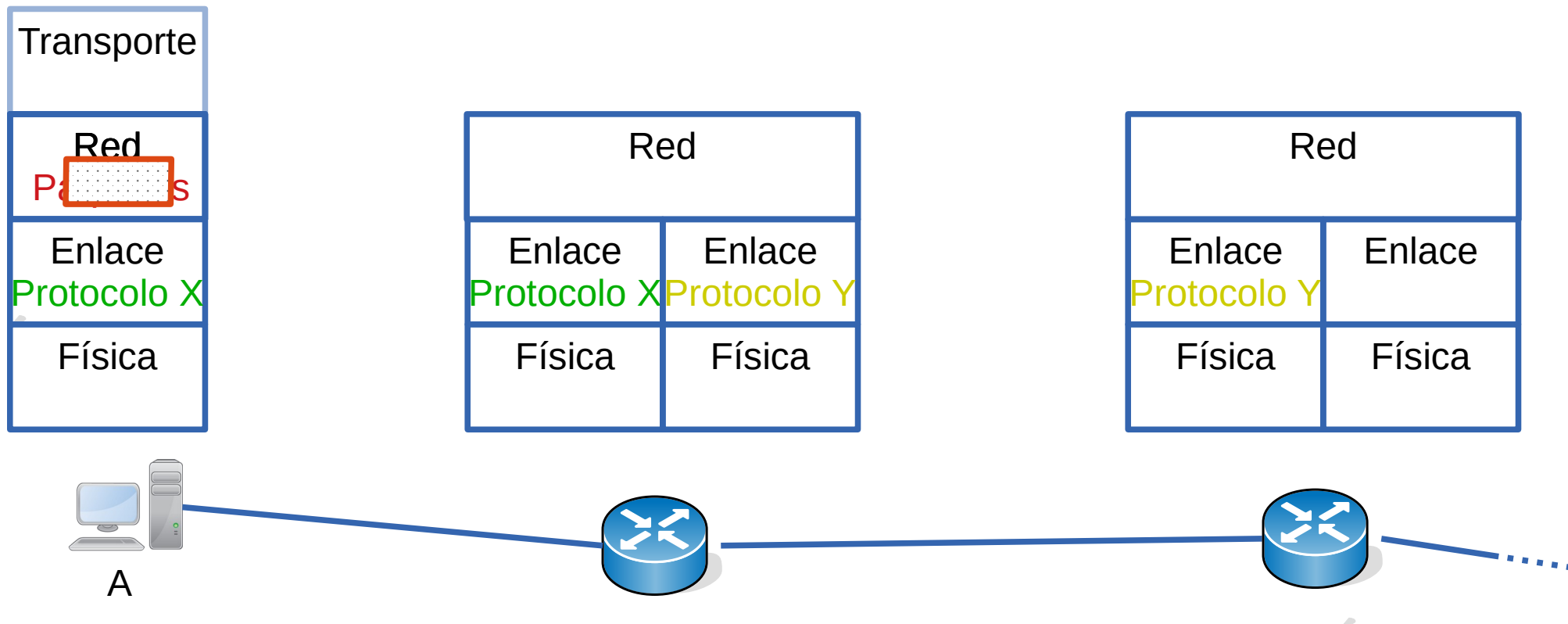
Fragmentación

- En cada salto, los paquetes viajan como carga útil del protocolo de capa de enlace
- Cada enlace podrá tener diferente tecnología y por tanto diferente protocolo de capa de enlace, con su correspondiente formato de trama
- En particular la máxima capacidad de carga útil de la capa de enlace se conoce como **MTU** (Maximum Transmission Unit)



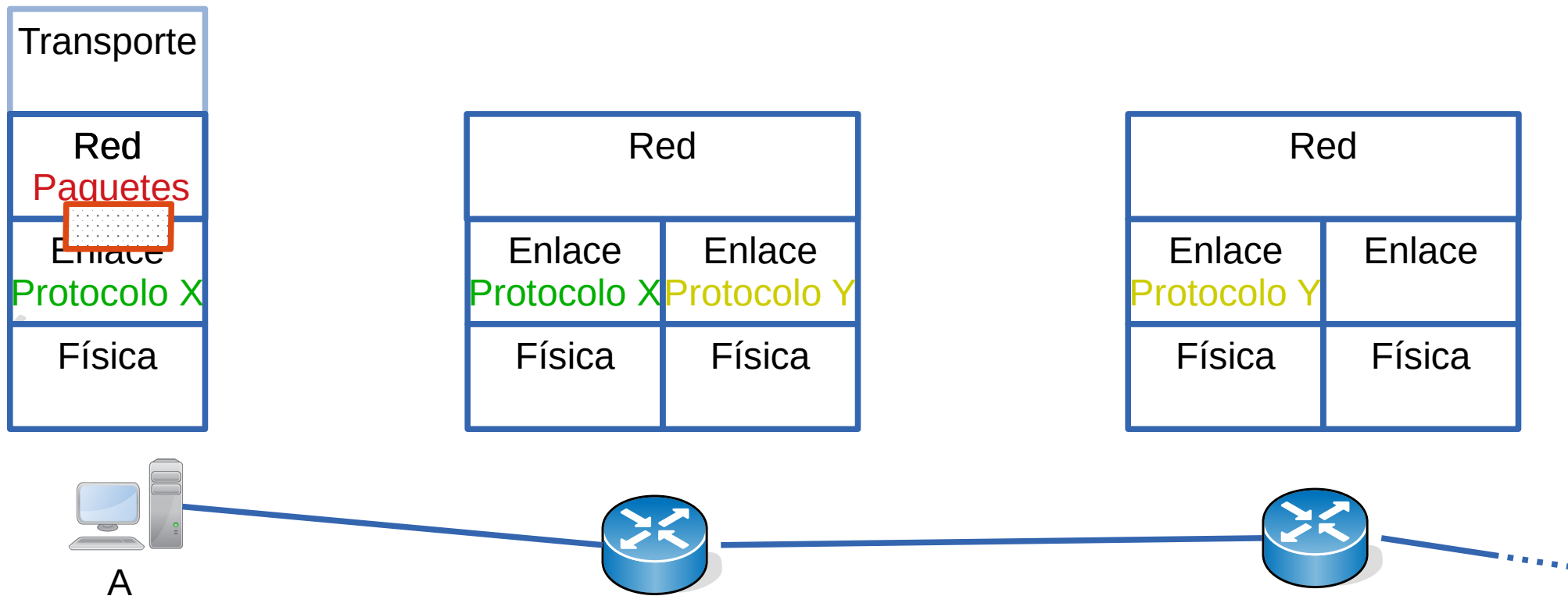
Fragmentación

- En cada salto, los paquetes viajan como carga útil del protocolo de capa de enlace
- Cada enlace podrá tener diferente tecnología y por tanto diferente protocolo de capa de enlace, con su correspondiente formato de trama
- En particular la máxima capacidad de carga útil de la capa de enlace se conoce como **MTU** (Maximum Transmission Unit)



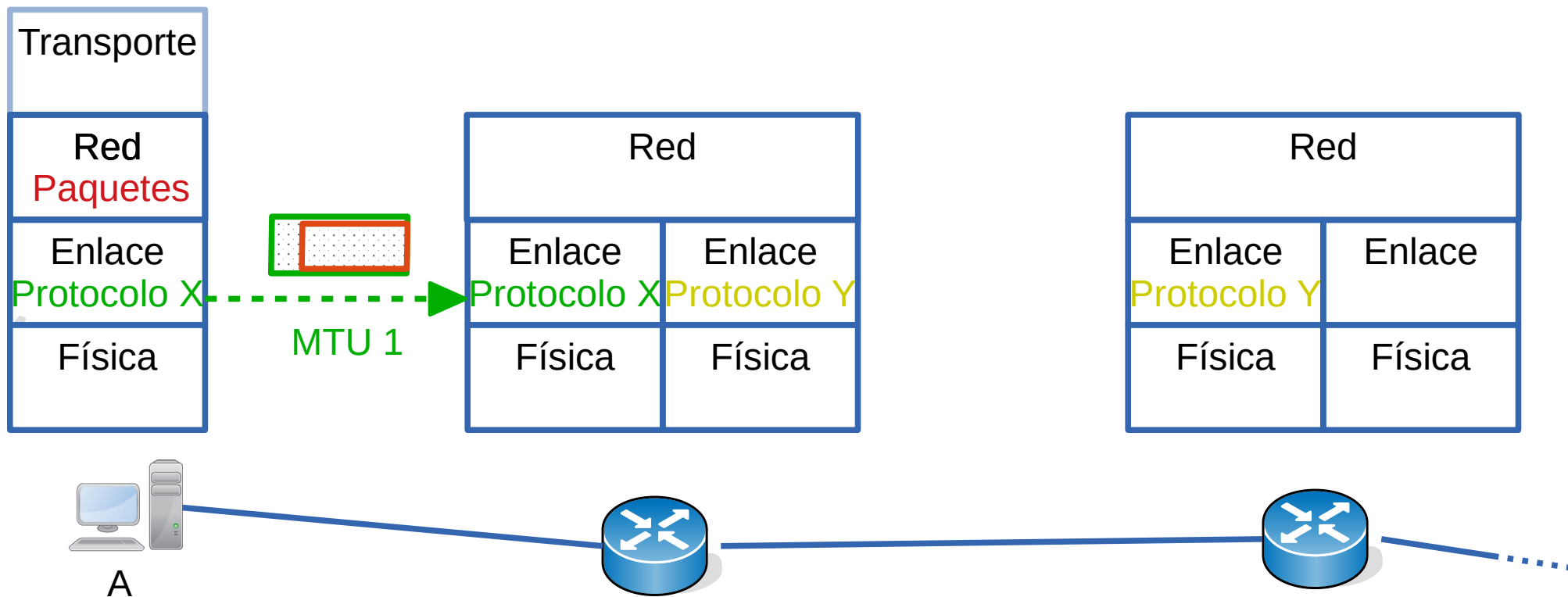
Fragmentación

- En cada salto, los paquetes viajan como carga útil del protocolo de capa de enlace
- Cada enlace podrá tener diferente tecnología y por tanto diferente protocolo de capa de enlace, con su correspondiente formato de trama
- En particular la máxima capacidad de carga útil de la capa de enlace se conoce como **MTU** (Maximum Transmission Unit)



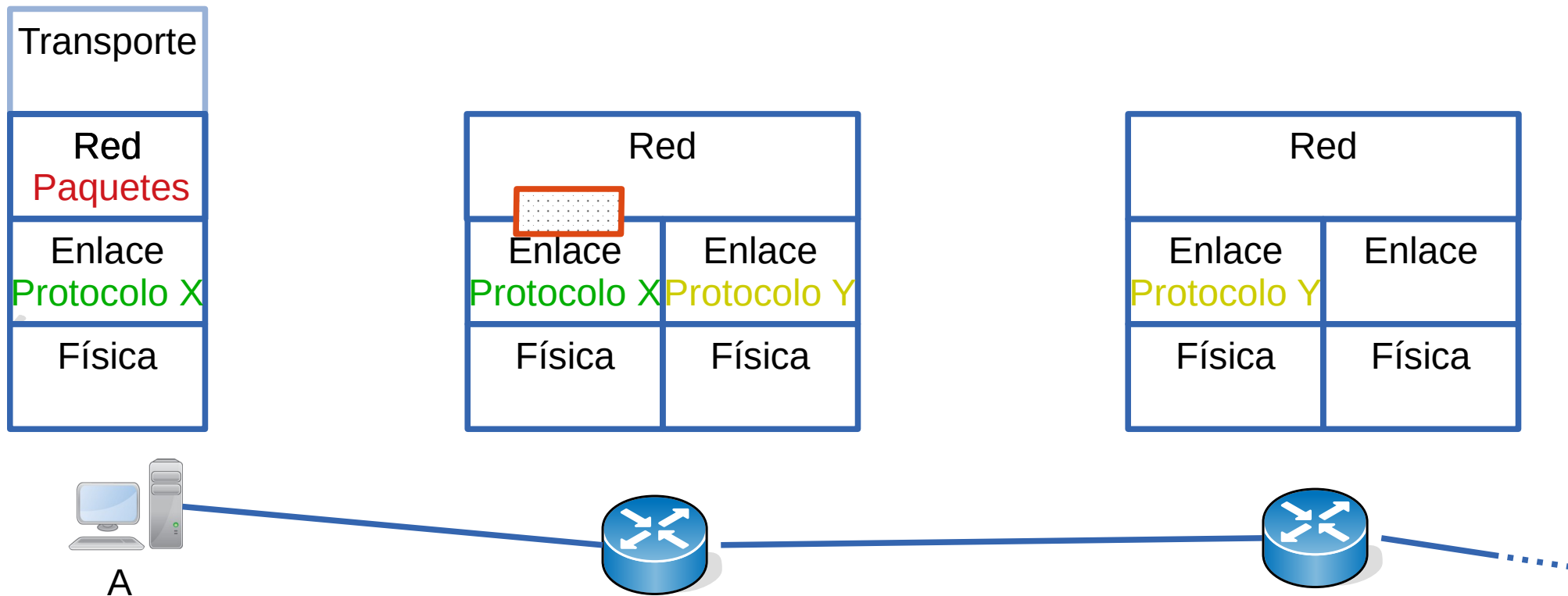
Fragmentación

- En cada salto, los paquetes viajan como carga útil del protocolo de capa de enlace
- Cada enlace podrá tener diferente tecnología y por tanto diferente protocolo de capa de enlace, con su correspondiente formato de trama
- En particular la máxima capacidad de carga útil de la capa de enlace se conoce como **MTU** (Maximum Transmission Unit)



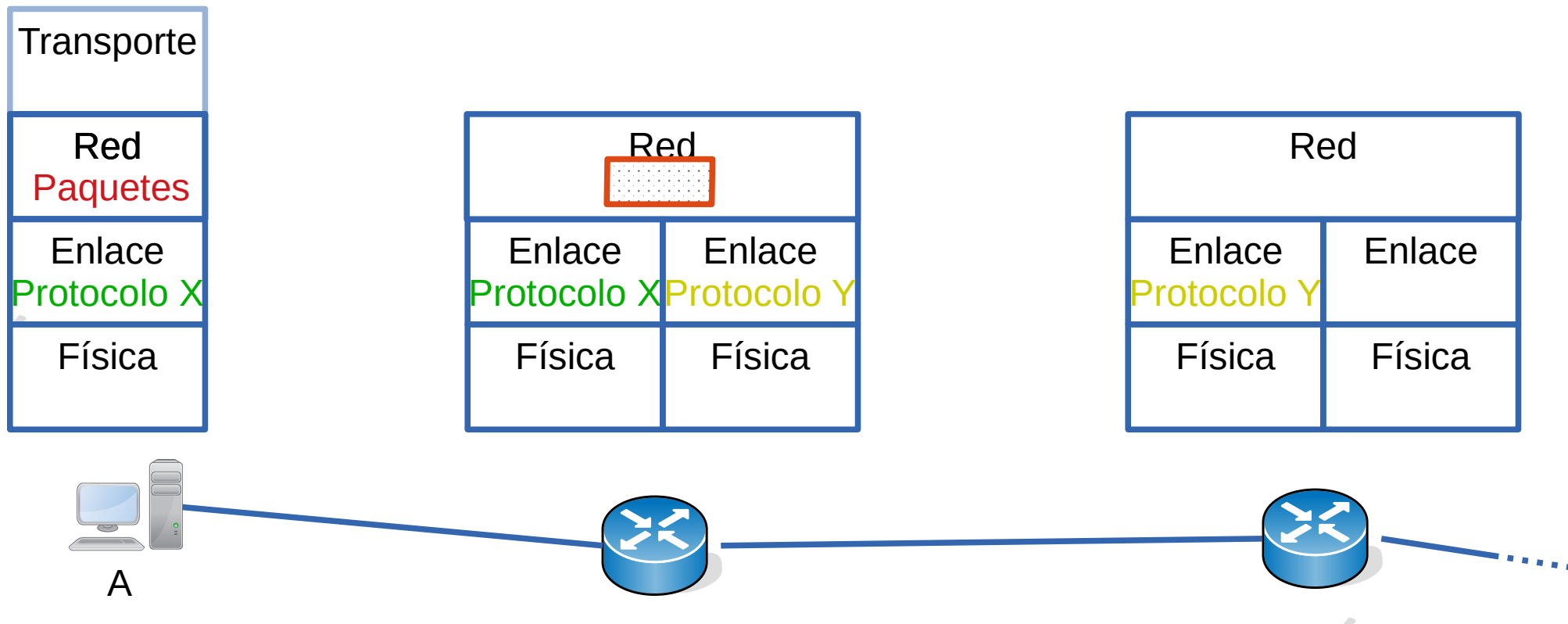
Fragmentación

- En cada salto, los paquetes viajan como carga útil del protocolo de capa de enlace
- Cada enlace podrá tener diferente tecnología y por tanto diferente protocolo de capa de enlace, con su correspondiente formato de trama
- En particular la máxima capacidad de carga útil de la capa de enlace se conoce como **MTU** (Maximum Transmission Unit)



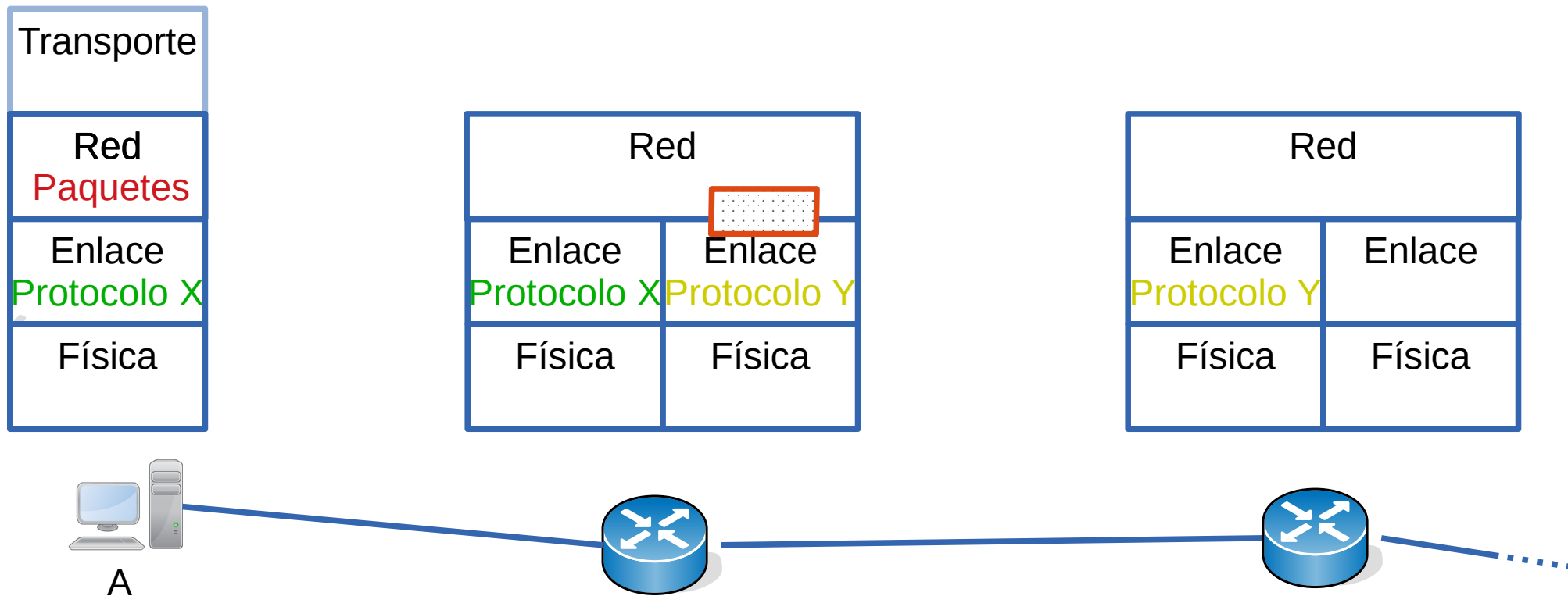
Fragmentación

- En cada salto, los paquetes viajan como carga útil del protocolo de capa de enlace
- Cada enlace podrá tener diferente tecnología y por tanto diferente protocolo de capa de enlace, con su correspondiente formato de trama
- En particular la máxima capacidad de carga útil de la capa de enlace se conoce como **MTU** (Maximum Transmission Unit)



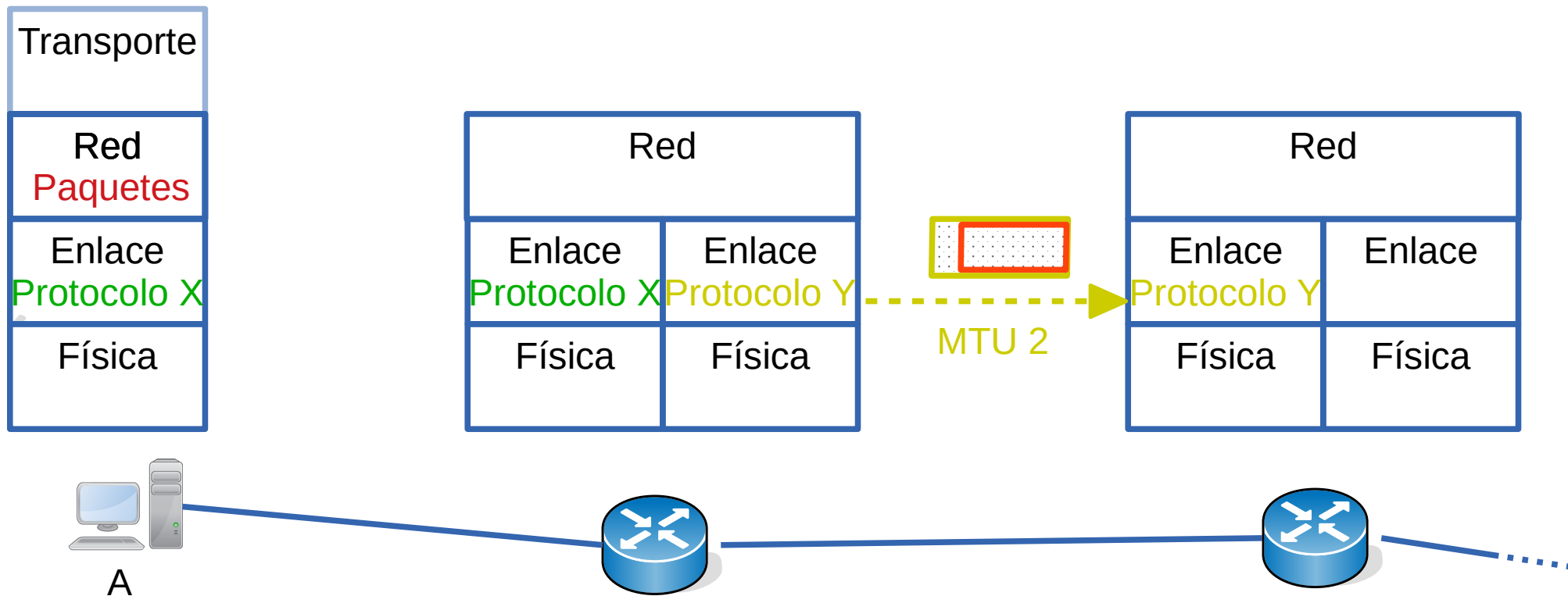
Fragmentación

- En cada salto, los paquetes viajan como carga útil del protocolo de capa de enlace
- Cada enlace podrá tener diferente tecnología y por tanto diferente protocolo de capa de enlace, con su correspondiente formato de trama
- En particular la máxima capacidad de carga útil de la capa de enlace se conoce como **MTU** (Maximum Transmission Unit)



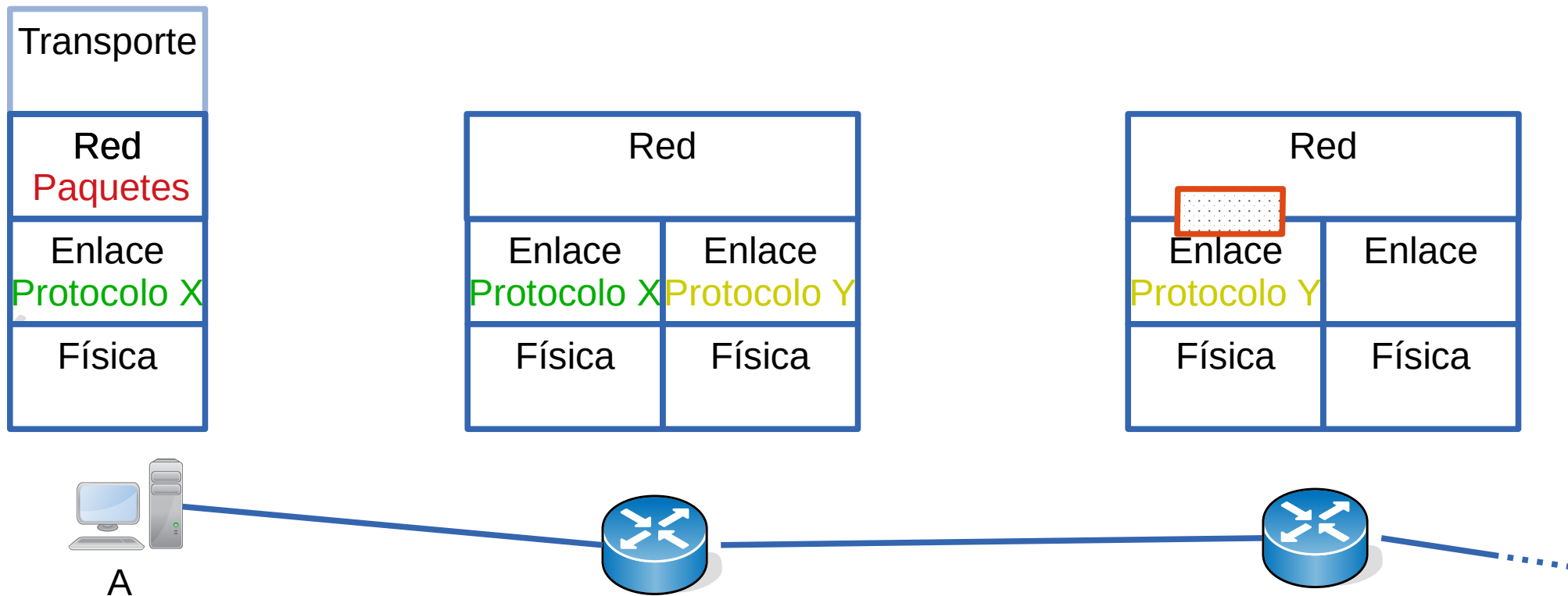
Fragmentación

- En cada salto, los paquetes viajan como carga útil del protocolo de capa de enlace
- Cada enlace podrá tener diferente tecnología y por tanto diferente protocolo de capa de enlace, con su correspondiente formato de trama
- En particular la máxima capacidad de carga útil de la capa de enlace se conoce como **MTU** (Maximum Transmission Unit)



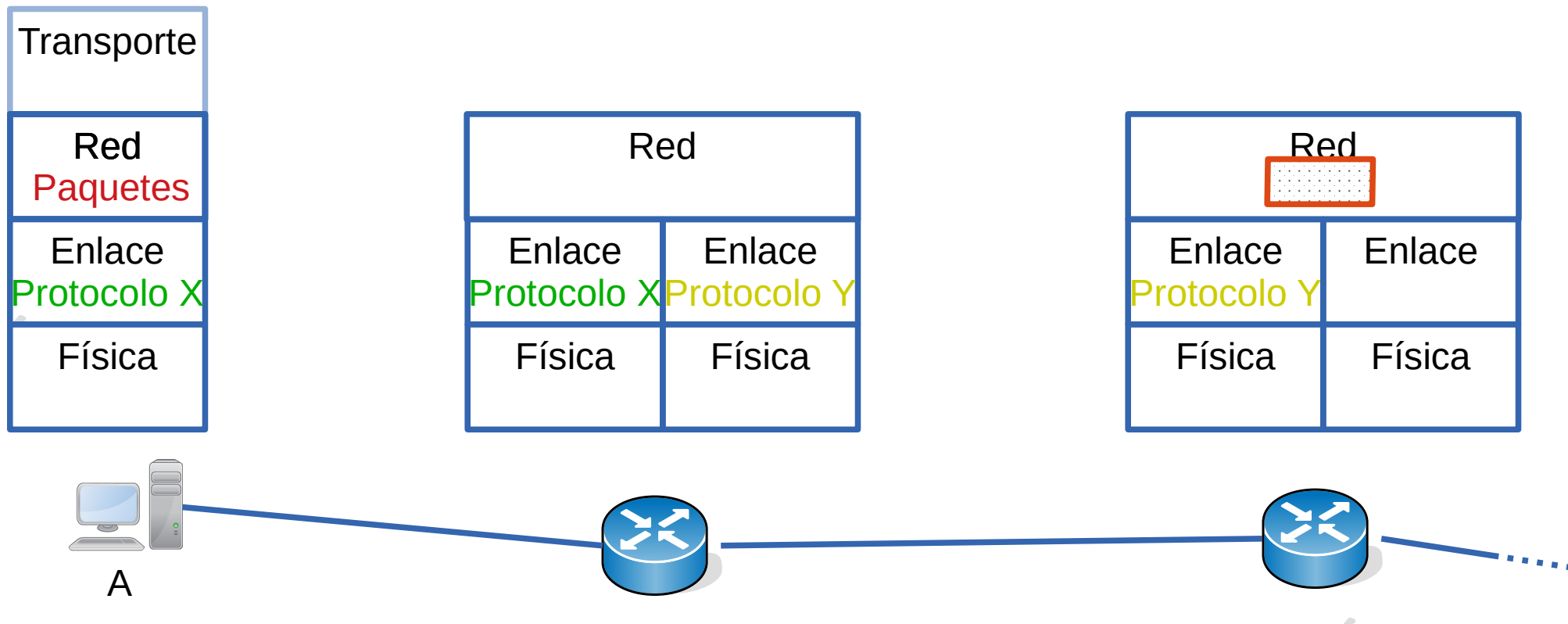
Fragmentación

- En cada salto, los paquetes viajan como carga útil del protocolo de capa de enlace
- Cada enlace podrá tener diferente tecnología y por tanto diferente protocolo de capa de enlace, con su correspondiente formato de trama
- En particular la máxima capacidad de carga útil de la capa de enlace se conoce como **MTU** (Maximum Transmission Unit)



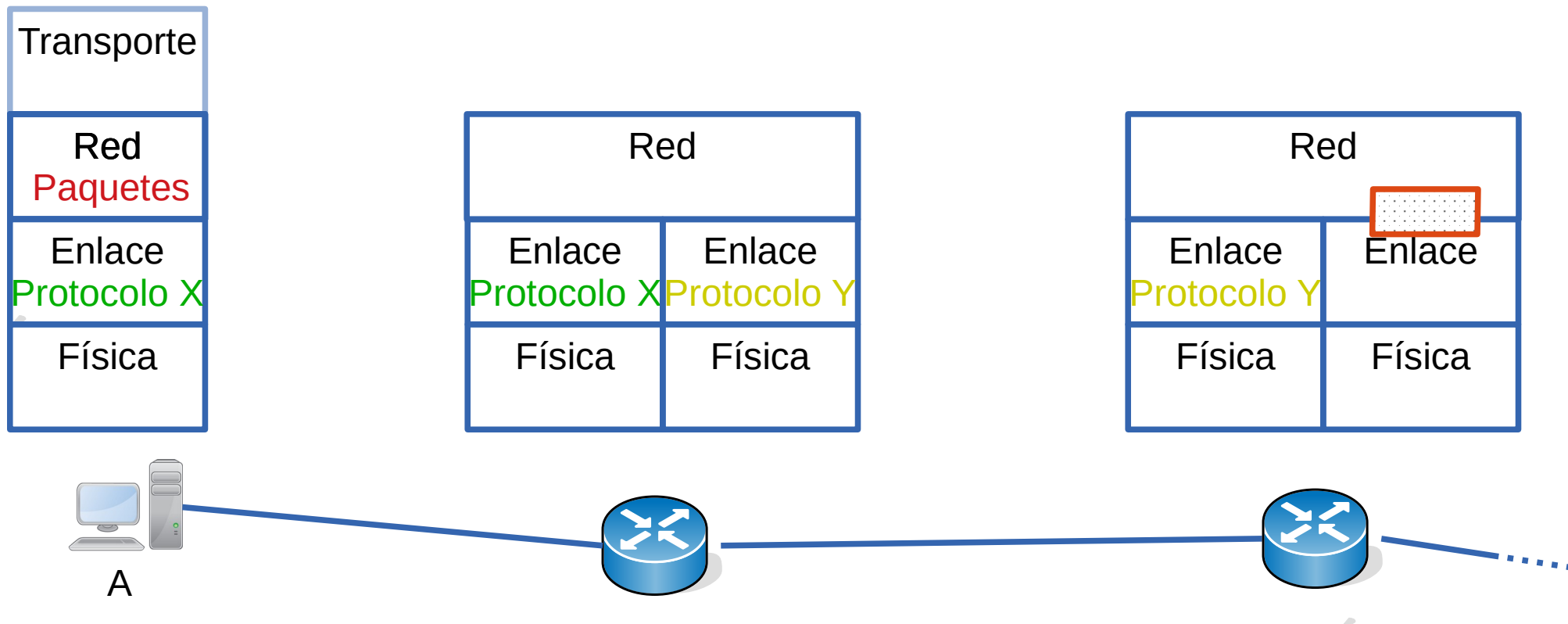
Fragmentación

- En cada salto, los paquetes viajan como carga útil del protocolo de capa de enlace
- Cada enlace podrá tener diferente tecnología y por tanto diferente protocolo de capa de enlace, con su correspondiente formato de trama
- En particular la máxima capacidad de carga útil de la capa de enlace se conoce como **MTU** (Maximum Transmission Unit)



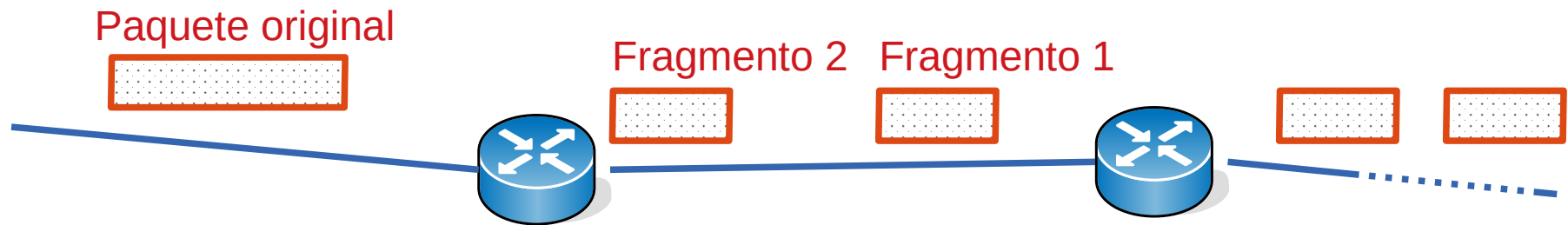
Fragmentación

- En cada salto, los paquetes viajan como carga útil del protocolo de capa de enlace
- Cada enlace podrá tener diferente tecnología y por tanto diferente protocolo de capa de enlace, con su correspondiente formato de trama
- En particular la máxima capacidad de carga útil de la capa de enlace se conoce como **MTU** (Maximum Transmission Unit)



Fragmentación

- ¿Qué sucede si la MTU de un enlace es menor que la del enlace anterior?
- Es necesario dividir el paquete en paquetes más pequeños: **fragmentar**
- Como el trabajo de fragmentar y reensamblar es una tarea costosa, una vez que el paquete IP se fragmenta, el reensamblaje se realiza en el destino



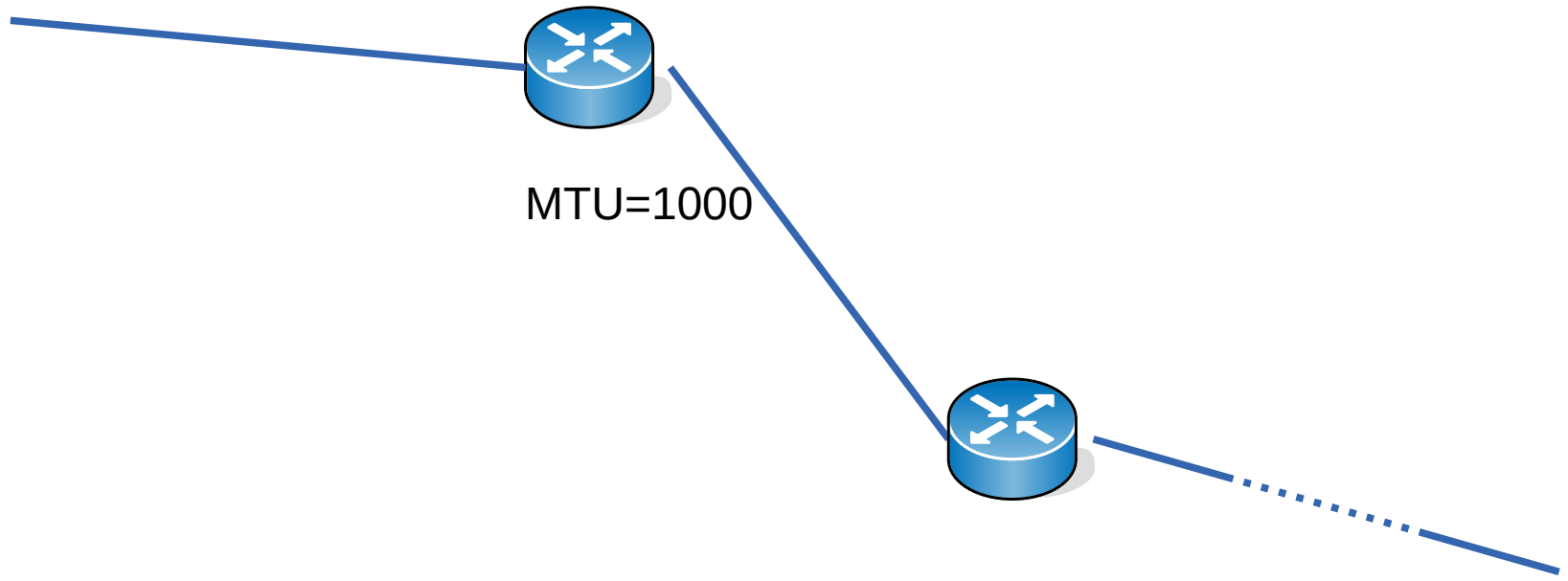
- En recepción es necesario:
 - Saber si un fragmento pertenece a un paquete de mayor tamaño
 - El **identificador de paquete** es común a todos los fragmentos
 - Saber ordenar los fragmentos para armar el paquete original
 - Campo **offset de fragmento**
 - Saber si llegaron todos los fragmentos para armar el paquete original
 - La bandera **MF** (More Fragments) es 1 en todos los fragmentos menos el último
 - Si no se quiere que un paquete se fragmente se le establece la bandera **DF** (Don't fragment) en 1

Ejemplo: Fragmentación

Paquete original

Id=15, off=0, MF=0

1500 bytes

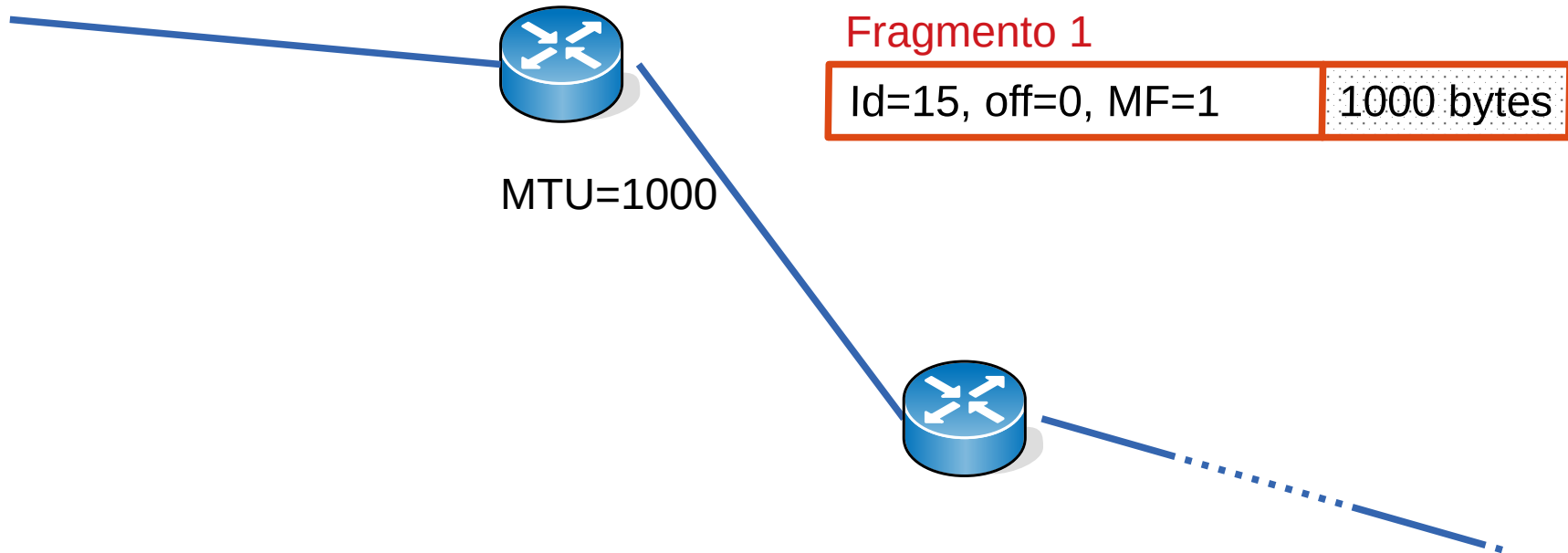


Ejemplo: Fragmentación

Paquete original

Id=15, off=0, MF=0

1500 bytes

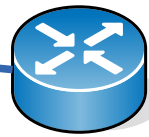


Ejemplo: Fragmentación

Paquete original

Id=15, off=0, MF=0

1500 bytes



MTU=1000

Fragmento 1

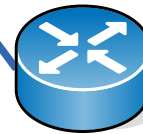
Id=15, off=0, MF=1

1000 bytes

Fragmento 2

Id=15, off=1000, MF=0

500 bytes

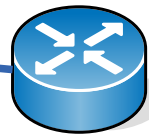


Ejemplo: Fragmentación

Paquete original

Id=15, off=0, MF=0

1500 bytes



MTU=1000

Fragmento 1

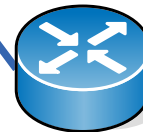
Id=15, off=0, MF=1

1000 bytes

Fragmento 2

Id=15, off=1000, MF=0

500 bytes



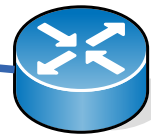
MTU=500

Ejemplo: Fragmentación

Paquete original

Id=15, off=0, MF=0

1500 bytes



MTU=1000

Fragmento 1

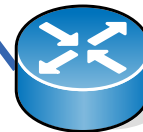
Id=15, off=0, MF=1

1000 bytes

Fragmento 2

Id=15, off=1000, MF=0

500 bytes



MTU=500

Fragmento 1.1

Id=15, off=0, MF=1

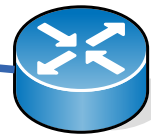
500 bytes

Ejemplo: Fragmentación

Paquete original

Id=15, off=0, MF=0

1500 bytes



MTU=1000

Fragmento 1

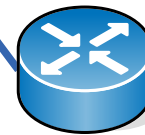
Id=15, off=0, MF=1

1000 bytes

Fragmento 2

Id=15, off=1000, MF=0

500 bytes



MTU=500

Fragmento 1.1

Id=15, off=0, MF=1

500 bytes

Fragmento 1.2

Id=15, off=500, MF=1

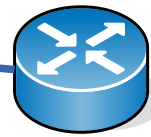
500 bytes

Ejemplo: Fragmentación

Paquete original

Id=15, off=0, MF=0

1500 bytes



MTU=1000

Fragmento 1

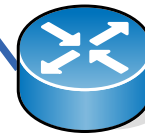
Id=15, off=0, MF=1

1000 bytes

Fragmento 2

Id=15, off=1000, MF=0

500 bytes



MTU=500

Fragmento 1.1

Id=15, off=0, MF=1

500 bytes

Fragmento 1.2

Id=15, off=500, MF=1

500 bytes

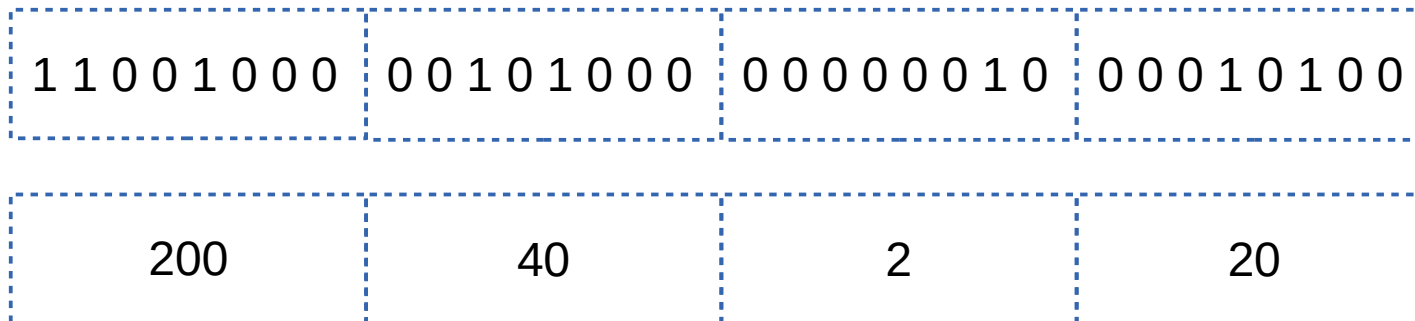
Fragmento 2

Id=15, off=1000, MF=0

500 bytes

Direcciones IPv4

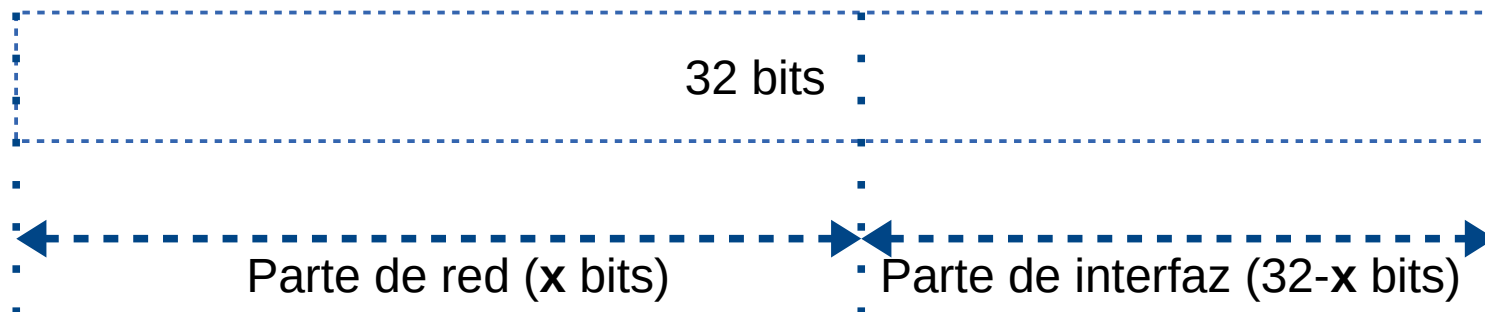
- Números de **32 bits** (4 bytes) que permiten identificar y localizar un dispositivo en la red
- Estrictamente las direcciones están asociadas a una interfaz de red
- Un equipo puede tener más de una interfaz si está conectado a varias redes
 - En particular los enrutadores normalmente tienen más de una interfaz porque interconectan al menos dos segmentos de red
- En general las direcciones IP deben ser únicas en la red (salvo en presencia del mecanismo de NAT que veremos luego)
- Se representan en el formato llamado: **dotted-decimal notation**



- Se escribe como: 200.40.2.20

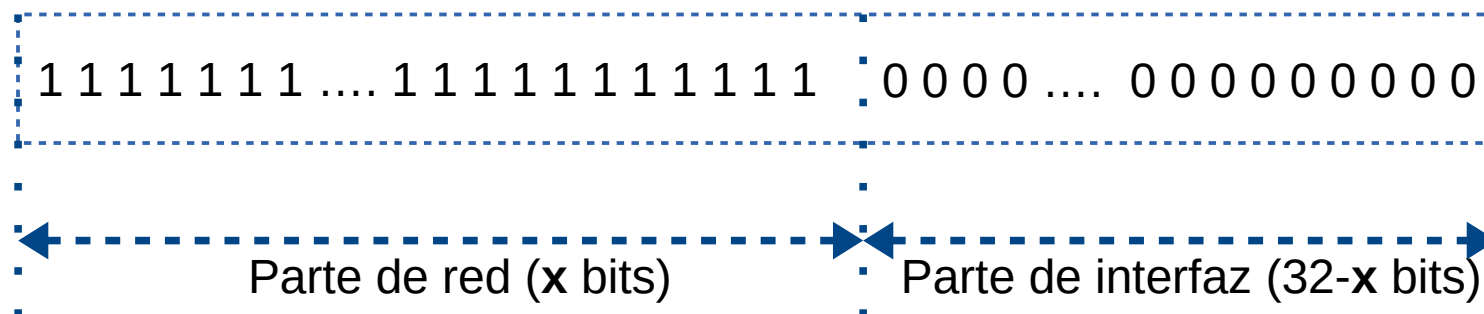
Rangos de direcciones

- Las direcciones se asignan por **rangos**
 - Si se asignaran direcciones individualmente, direcciones contiguas podrían ser asignadas a equipos de distintas partes del mundo y por tanto los enrutadores necesitarían tablas de forwarding más grandes
 - Tablas más grandes, implican más tiempo para buscar en ellas, más memoria
 - En las tablas de forwarding se utilizan esos rangos como destinos
- Un rango de direcciones está compuesto por el conjunto de direcciones que tienen un **prefijo** (los x primeros bits) en común
 - x es el largo del prefijo
- Determina la **parte de red** y la **parte de interfaz (o de host)**



Máscara de la red

- La división entre parte de red y parte de interfaz de los rangos de direcciones, también se suele especificar como una palabra de 32 bits que tiene 1's en la parte de red y 0's en la parte de interfaz



- Como son 32 bits, se puede usar la “dotted-decimal notation”
- Por ejemplo un largo de prefijo de 24 bits, se corresponde con una palabra de 32 bits donde
 - los primeros 24 están en 1, señalando la parte de red y
 - los últimos 8 en 0, señalando la parte de interfaz
- En dotted-decimal notation sería: 255.255.255.0
 - Van a ver esta notación en algunos sistemas operativos u otros equipos

Ejemplo de rangos de direcciones

- Las direcciones IP:
 - 223.1.1.0
 - 223.1.1.1
 - 223.1.1.2
 - ...
 - 223.1.1.255
- Tienen los primeros 3 bytes (24 bits) en común (223.1.1.x)
- Se puede referir este rango de direcciones como 223.1.1.0/24
 - Conjunto de direcciones consecutivas que empiezan en la 223.1.1.0 y mantienen los primeros 24 bits iguales
- Las direcciones con todos los bits de la parte de interfaz en 0 no se pueden asignar a interfaces ya que se reservan para identificar la red
 - En el ejemplo 223.1.1.0
- Las direcciones con todos los bits de la parte de interfaz en 1 no se pueden asignar a interfaces ya que se usan como dirección de difusión de la red
 - En el ejemplo 223.1.1.255

Classless Interdomain Routing (CIDR)

- La estrategia de definición de rangos de direcciones utilizando la especificación del largo del prefijo se conoce como **CIDR**
- Los rangos se especifican como **a.b.c.d/x** donde **x** es el largo del prefijo
- Antes de CIDR, las direcciones se dividían en clases de largos de prefijo preestablecidos en 8, 16 y 24: **Classfull Addressing**
- El esquema era muy rígido y comenzaron a escasear las clases B
- Vamos a hablar de clases A, B y C para prefijos de largo 8, 16 y 24

A	0	Red		Interfaz		Prefijo de largo 8
B	10	Red		Interfaz		Prefijo de largo 16
C	110	Red		Interfaz		Prefijo de largo 24
D	1110		Multicast			
E	1111		Reservadas			

División de rangos

- Supongamos que tenemos para nuestra red el rango [223.1.1.0/24](#) (direcciones útiles de la 223.1.1.1 a la 223.1.1.254)
- La función de ruteo resolverá que desde el resto de Internet sepan cómo llegar a ese rango de direcciones
- ¿Cómo usamos esas direcciones?
 - Depende de cómo queremos diseñar la red en nuestra institución
- Ejemplo:
 - Una sola red o subred
 - Asignamos a nuestros equipos direcciones dentro del rango
 - Dividimos el rango asignado en rangos menores
 - Por ejemplo, organizamos la red con rangos diferentes:
 - por piso
 - por secciones administrativas (personal, administración, etc)
- Para esto necesitamos [dividir el rango de direcciones asignado](#)

División de rangos

- Rango asignado: 223.1.1.0/24
 - Corresponde a las direcciones de 223.1.1.0 a 223.1.1.255

1 1 0 1 1 1 1 1	0 0 0 0 0 0 0 1	0 0 0 0 0 0 0 1	x x x x x x x x
-----------------	-----------------	-----------------	-----------------

- Podemos cambiar el largo del prefijo y **dividirlo en subrangos**
- Si agrandamos el prefijo 1 bit podemos ver ese rango como dos rangos:

1 1 0 1 1 1 1 1	0 0 0 0 0 0 0 1	0 0 0 0 0 0 0 1	0 x x x x x x x
1 1 0 1 1 1 1 1	0 0 0 0 0 0 0 1	0 0 0 0 0 0 0 1	1 x x x x x x x

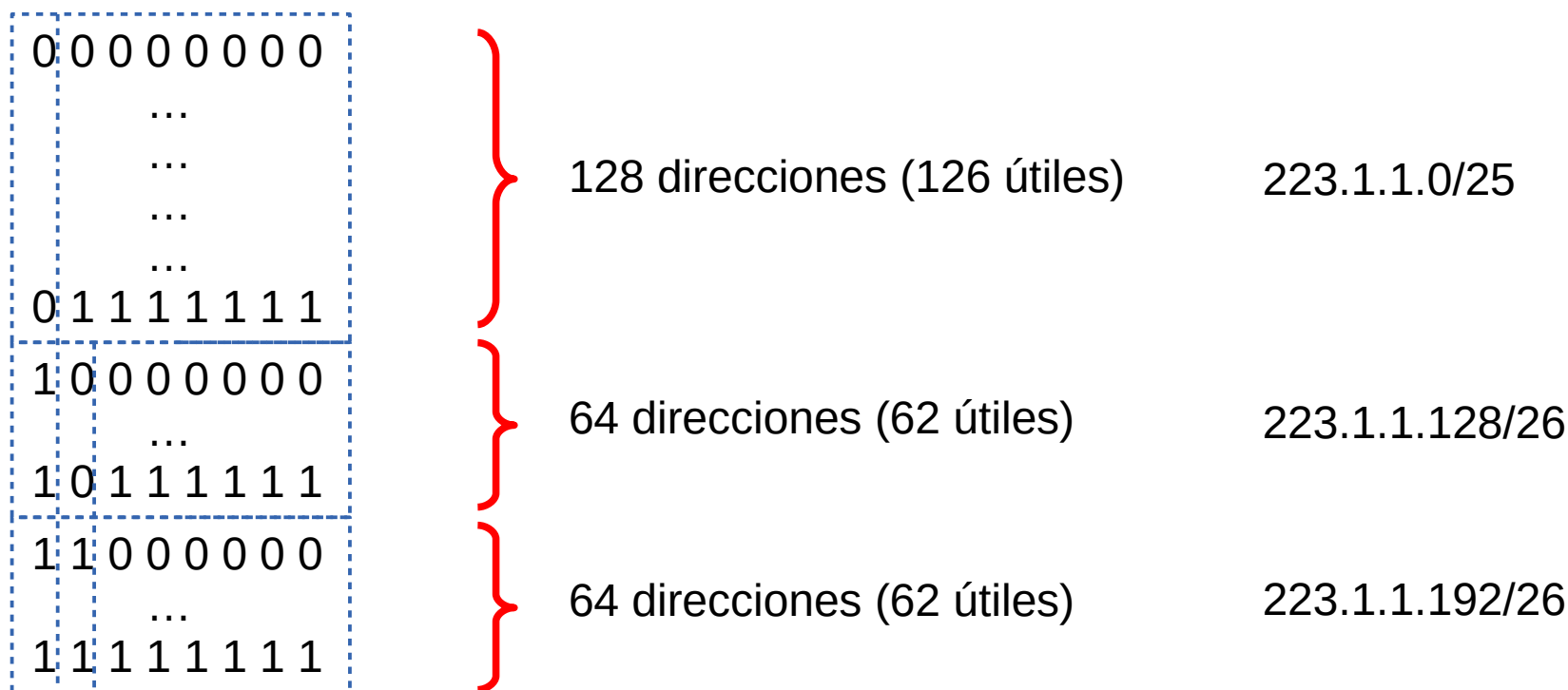
- El rango 223.1.1.0/24 puede partirse en dos agrandando el prefijo
 - 223.1.1.0/25 (128 direcciones, 126 útiles = .1 a .126)
 - 223.1.1.128/25 (128 direcciones, 126 útiles = .129 a .254)

Agregación o sumarización de rangos

- Así como podemos dividir un rango de direcciones en rangos más pequeños podemos **agregar o sumarizar rangos contiguos**
- Si usamos los rangos:
 - 223.1.0.0/24
 - 223.1.1.0/24
 - 223.1.2.0/24
 - 223.1.3.0/24
- Podemos agregarlos o sumarizarlos en dos rangos:
 - 223.1.0.0/23
 - 223.1.2.0/23
- O en un rango
 - 223.1.0.0/22

División de rangos

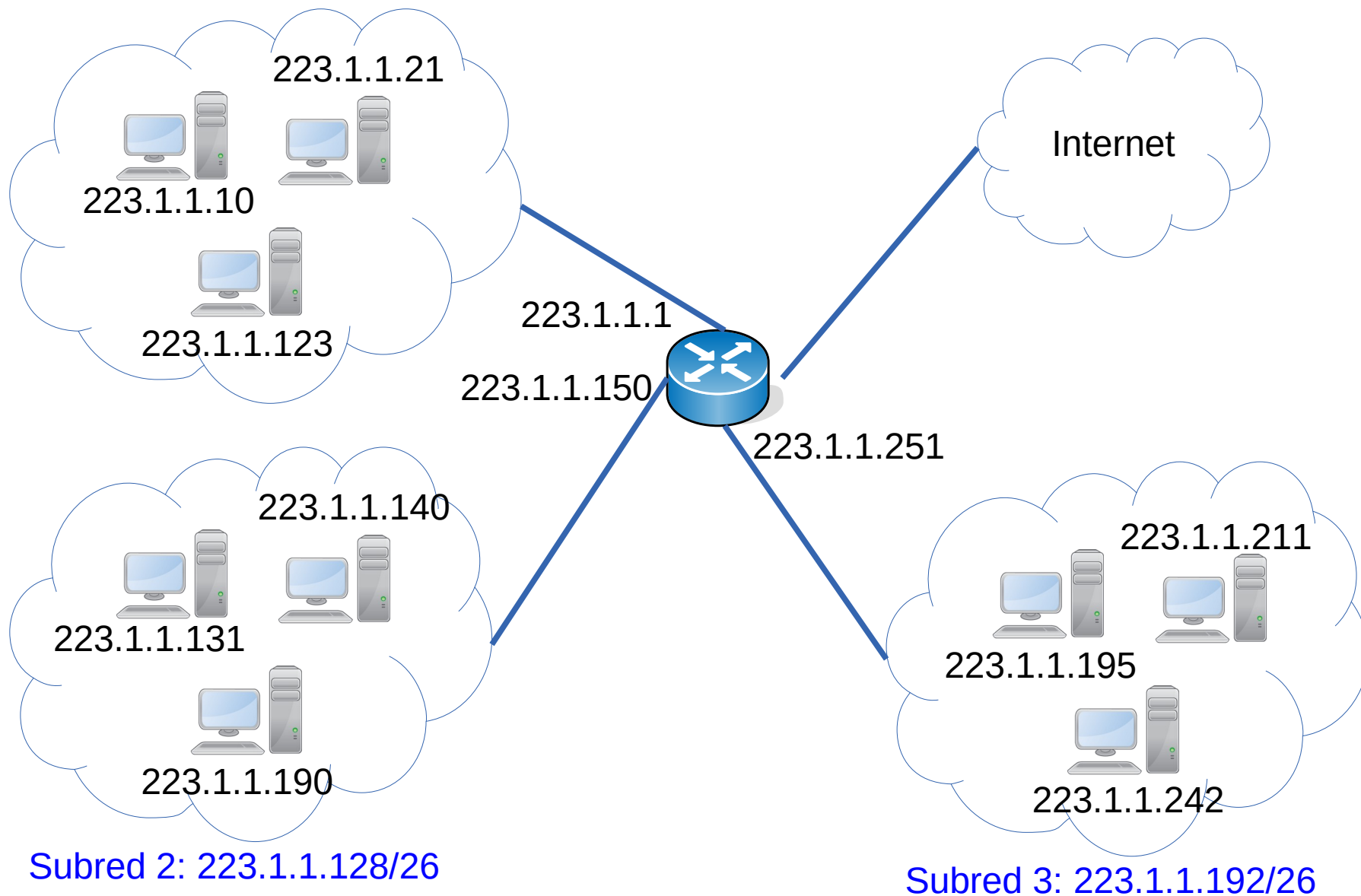
- Ejemplo: Quiero dividir el rango asignado 223.1.1.0/24 en 3 subredes:
 - Subred 1: 70 equipos
 - Subred 2: 40 equipos
 - Subred 3: 40 equipos
- Agrandando el prefijo puedo dividir las 255 direcciones en un bloque de 128 y dos bloques de 64, lo que permitiría cumplir los requerimientos



Asignación de rangos

- Una vez definidas las subredes asigno direcciones a los equipos

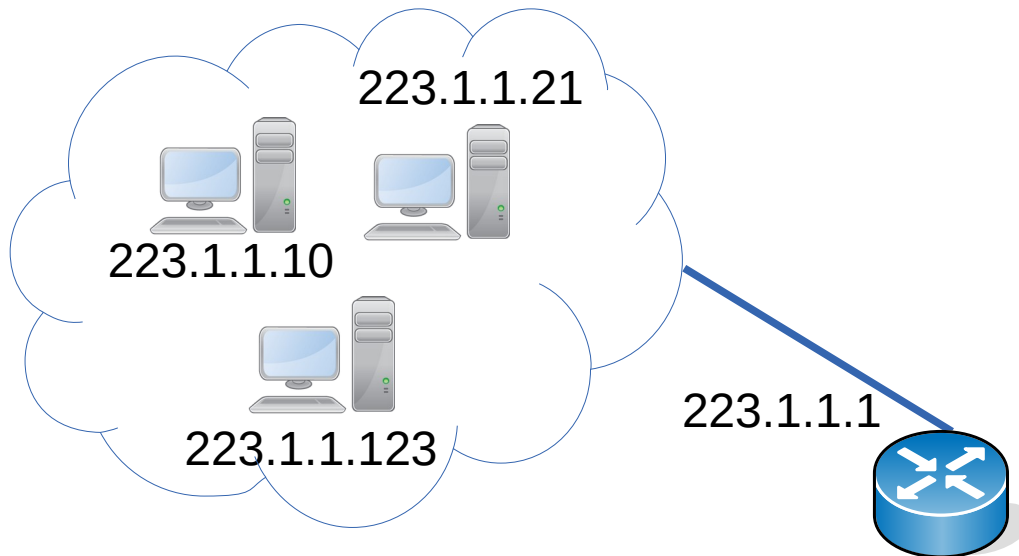
Subred 1: 223.1.1.0/25



Subredes

- Cuando asignamos un rango de direcciones a una subred, se entiende que los equipos de esa subred **se pueden comunicar entre sí sin necesidad de pasar por un enrutador**.
- La conexión puede ser mediante un medio compartido por los equipos de la subred o un enlace punto a punto entre dos equipos
- Estas tecnologías de conexión operan a nivel de **capa de enlace (capa 2)** y las veremos más adelante
- Una subred la representaremos lógicamente como un **medio compartido** entre los equipos

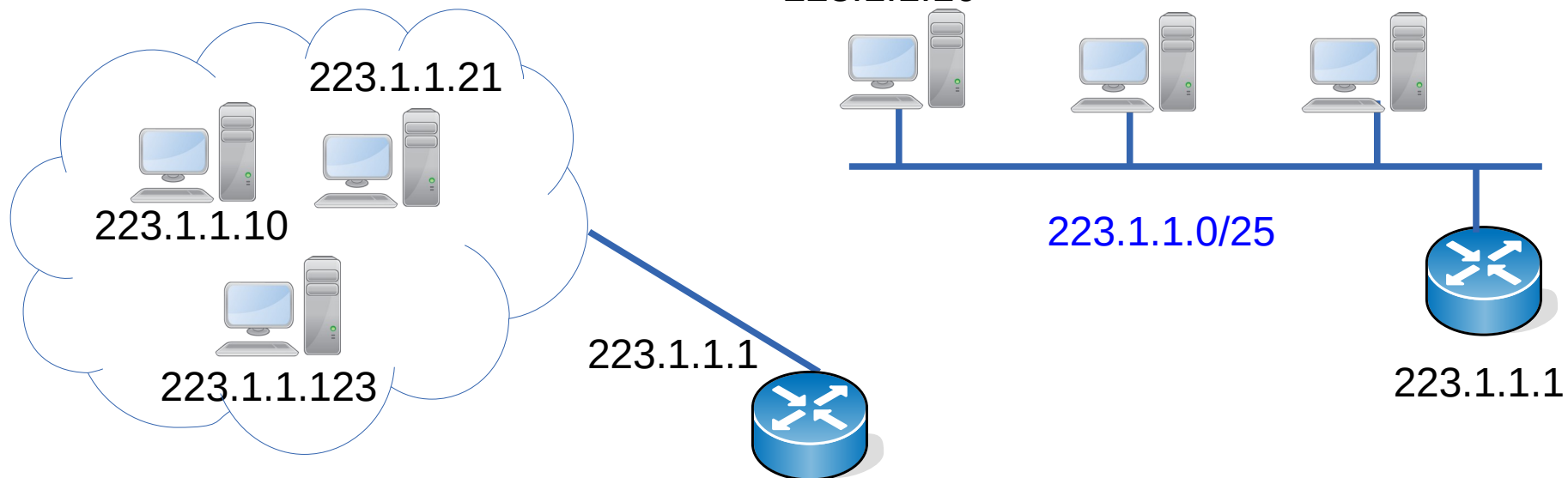
Subred 1: 223.1.1.0/25



Subredes

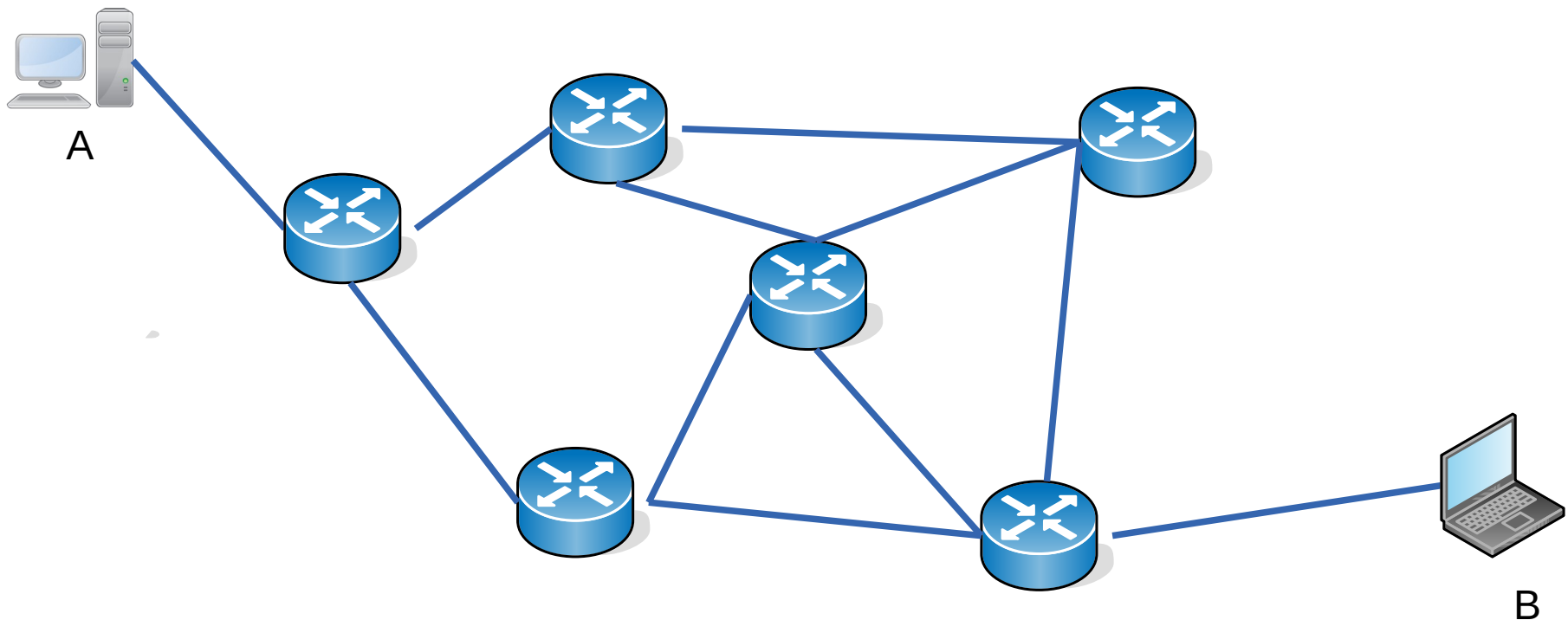
- Cuando asignamos un rango de direcciones a una subred, se entiende que los equipos de esa subred **se pueden comunicar entre sí sin necesidad de pasar por un enrutador**.
- La conexión puede ser mediante un medio compartido por los equipos de la subred o un enlace punto a punto entre dos equipos
- Estas tecnologías de conexión operan a nivel de **capa de enlace (capa 2)** y las veremos más adelante
- Una subred la representaremos lógicamente como un **medio compartido** entre los equipos

Subred 1: 223.1.1.0/25



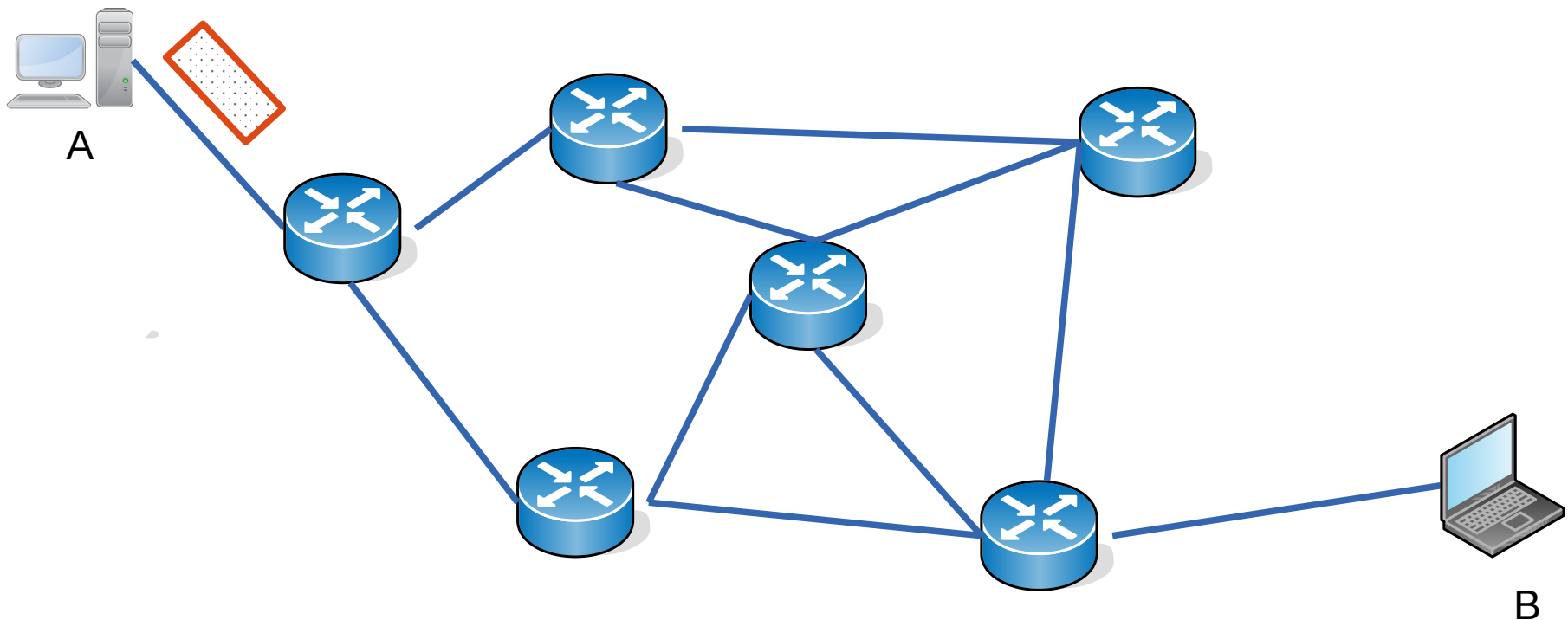
Encaminamiento o forwarding de paquetes

- Genéricamente cada nodo
 - Recibe un paquete por una de sus interfaces de entrada
 - Consulta su [tabla de forwarding](#)
 - Obtiene de la tabla el próximo salto para ese destino
 - Encamina el paquete por esa interfaz o próximo salto



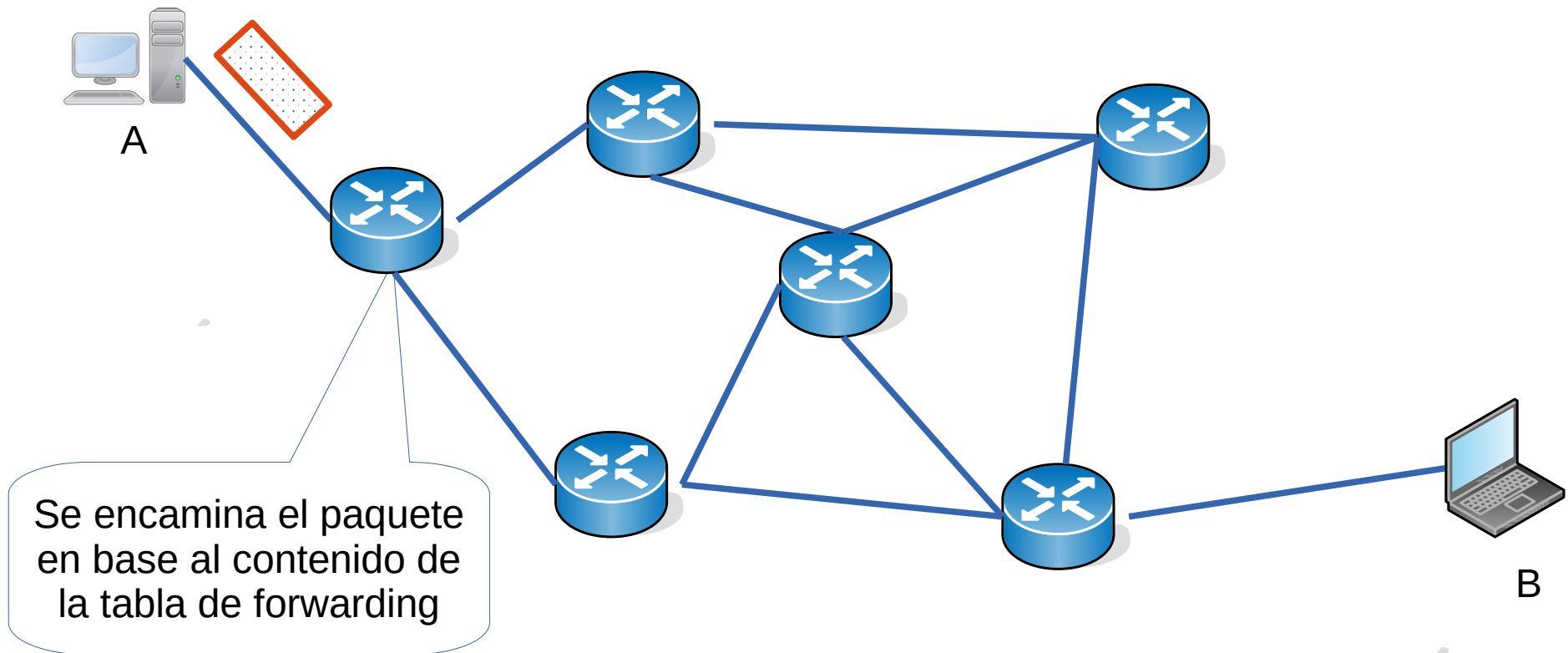
Encaminamiento o forwarding de paquetes

- Genéricamente cada nodo
 - Recibe un paquete por una de sus interfaces de entrada
 - Consulta su [tabla de forwarding](#)
 - Obtiene de la tabla el próximo salto para ese destino
 - Encamina el paquete por esa interfaz o próximo salto



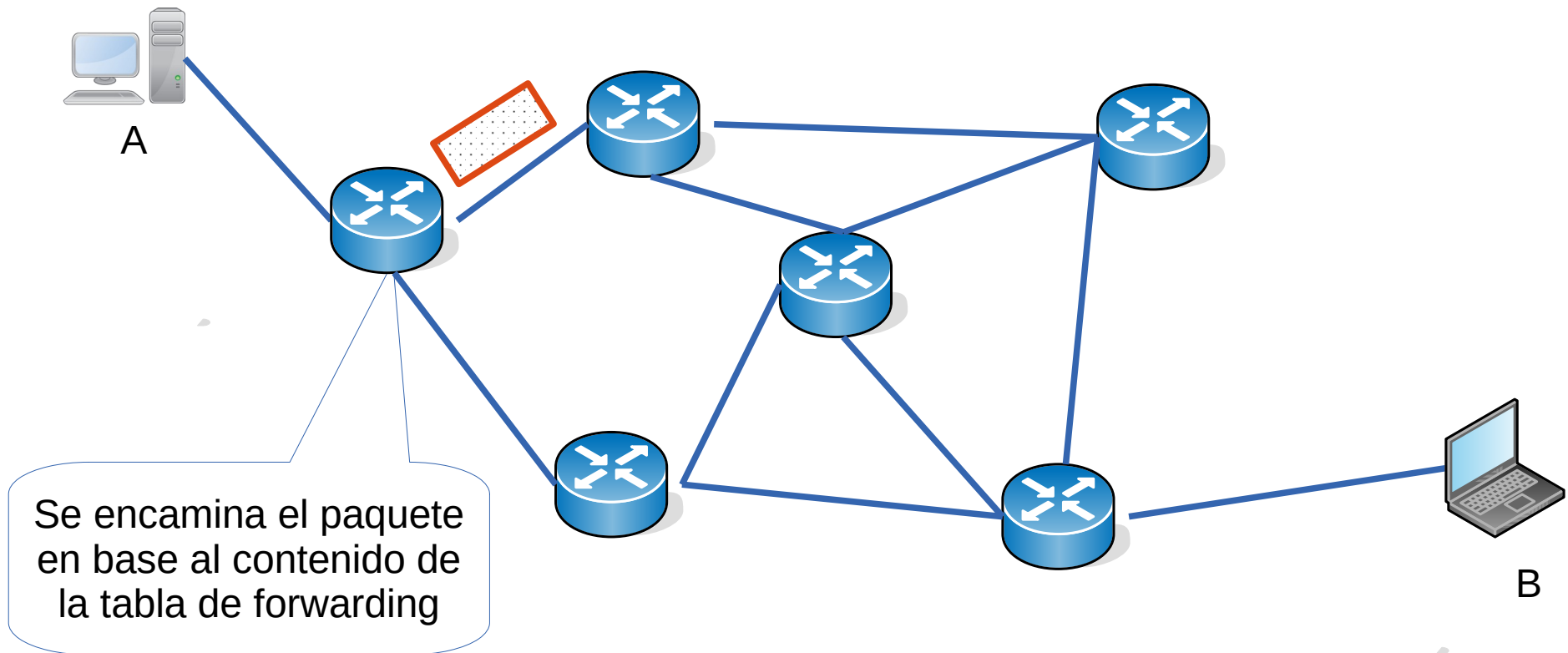
Encaminamiento o forwarding de paquetes

- Genéricamente cada nodo
 - Recibe un paquete por una de sus interfaces de entrada
 - Consulta su [tabla de forwarding](#)
 - Obtiene de la tabla el próximo salto para ese destino
 - Encamina el paquete por esa interfaz o próximo salto



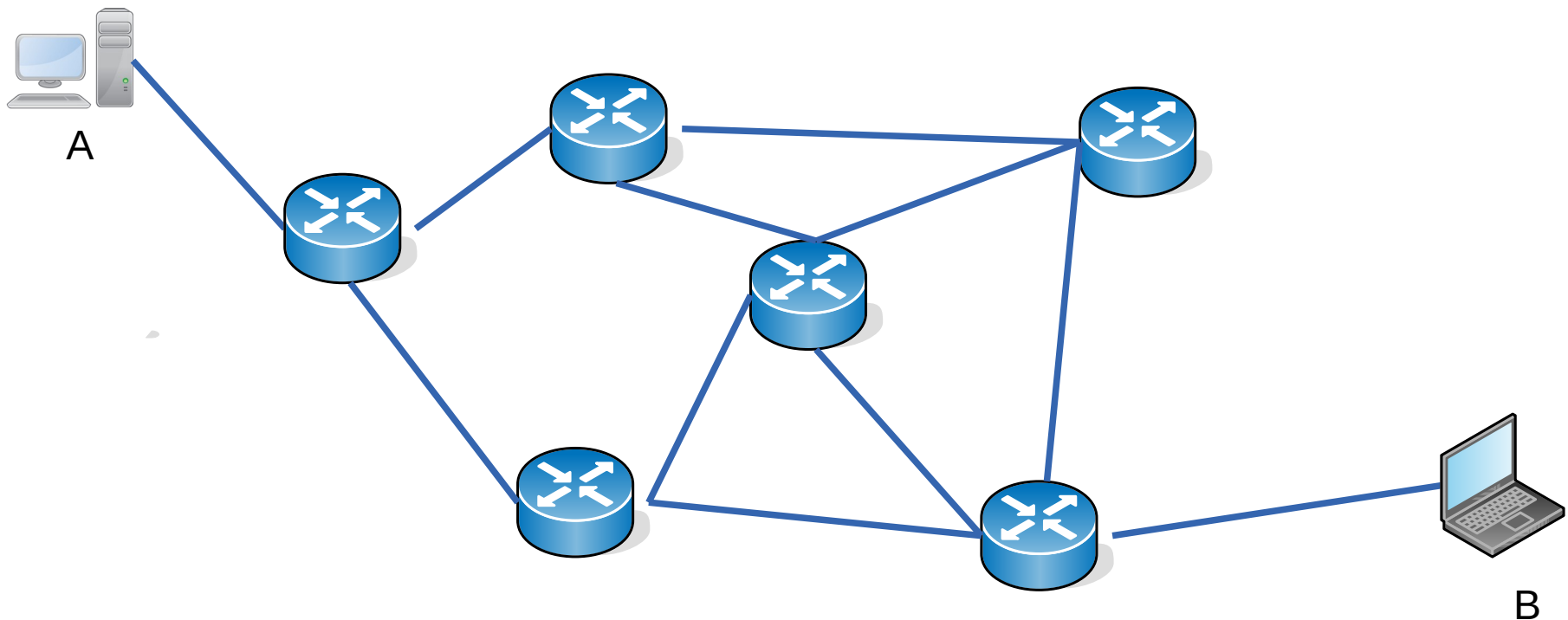
Encaminamiento o forwarding de paquetes

- Genéricamente cada nodo
 - Recibe un paquete por una de sus interfaces de entrada
 - Consulta su [tabla de forwarding](#)
 - Obtiene de la tabla el próximo salto para ese destino
 - Encamina el paquete por esa interfaz o próximo salto



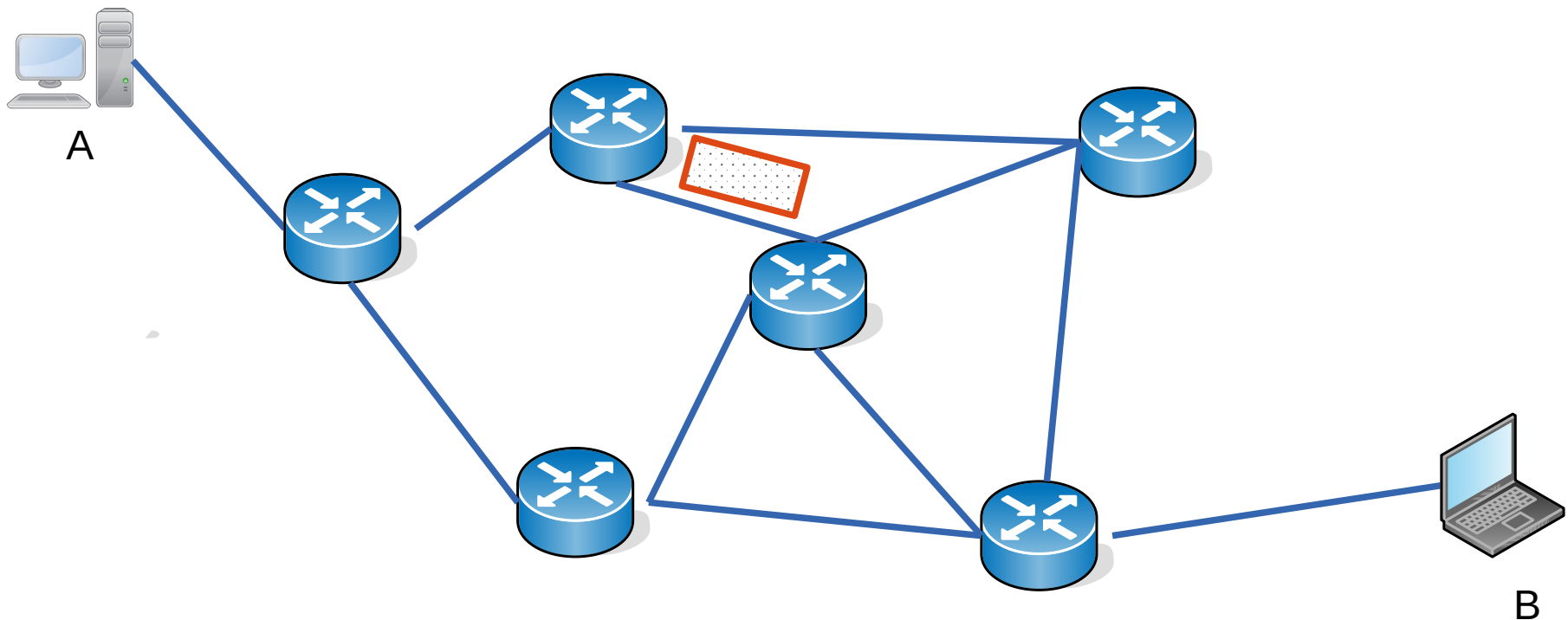
Encaminamiento o forwarding de paquetes

- Genéricamente cada nodo
 - Recibe un paquete por una de sus interfaces de entrada
 - Consulta su [tabla de forwarding](#)
 - Obtiene de la tabla el próximo salto para ese destino
 - Encamina el paquete por esa interfaz o próximo salto



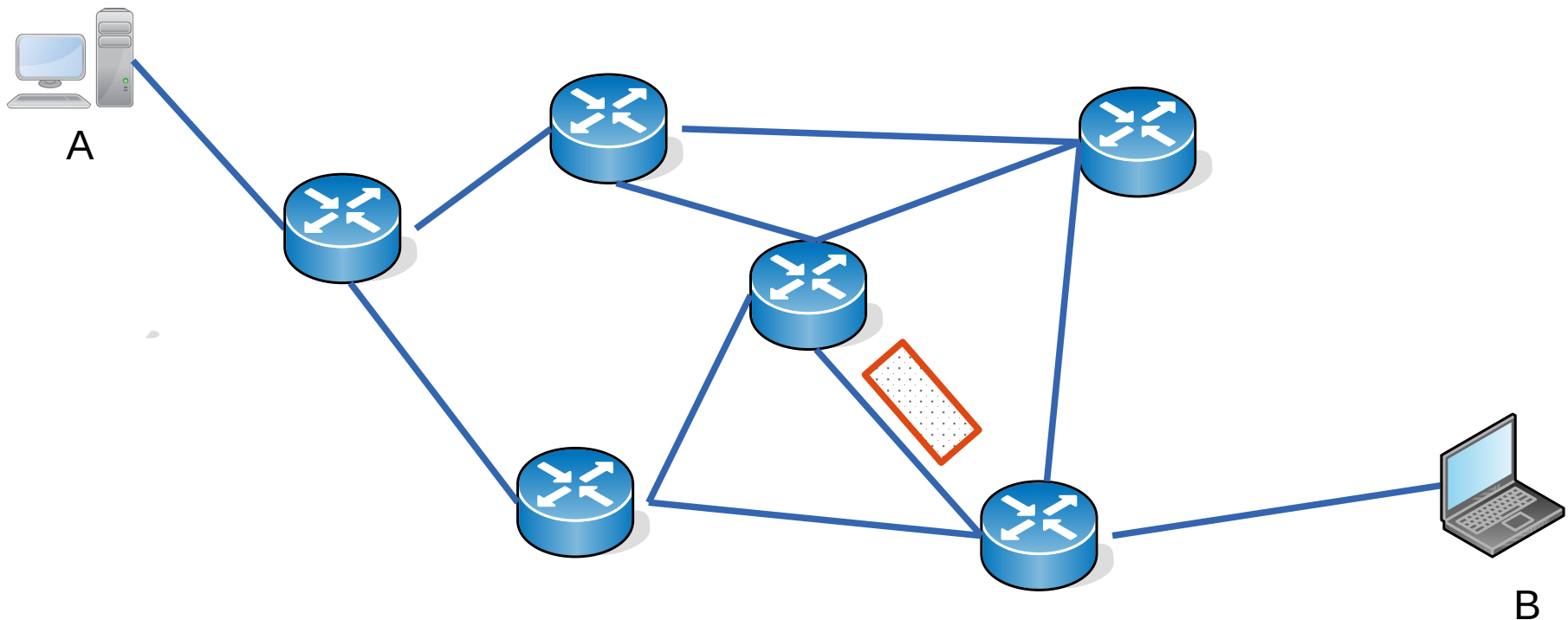
Encaminamiento o forwarding de paquetes

- Genéricamente cada nodo
 - Recibe un paquete por una de sus interfaces de entrada
 - Consulta su [tabla de forwarding](#)
 - Obtiene de la tabla el próximo salto para ese destino
 - Encamina el paquete por esa interfaz o próximo salto



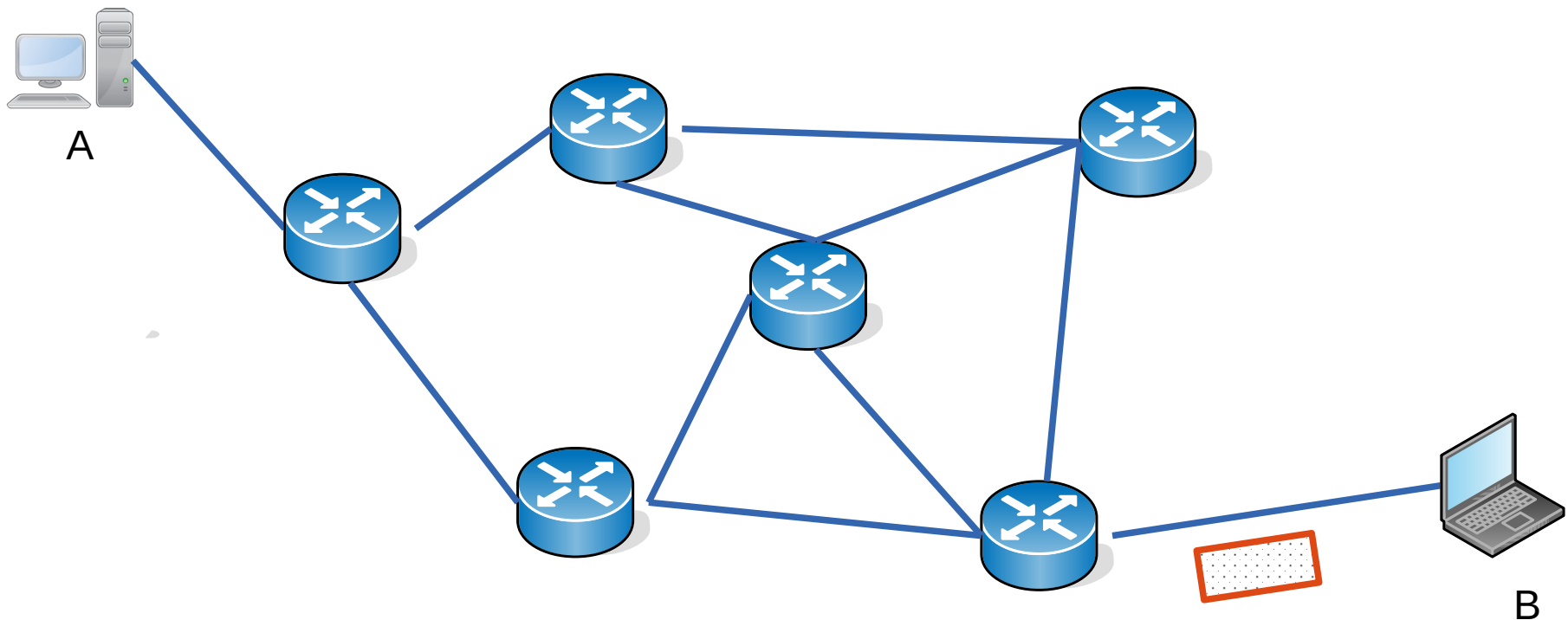
Encaminamiento o forwarding de paquetes

- Genéricamente cada nodo
 - Recibe un paquete por una de sus interfaces de entrada
 - Consulta su [tabla de forwarding](#)
 - Obtiene de la tabla el próximo salto para ese destino
 - Encamina el paquete por esa interfaz o próximo salto



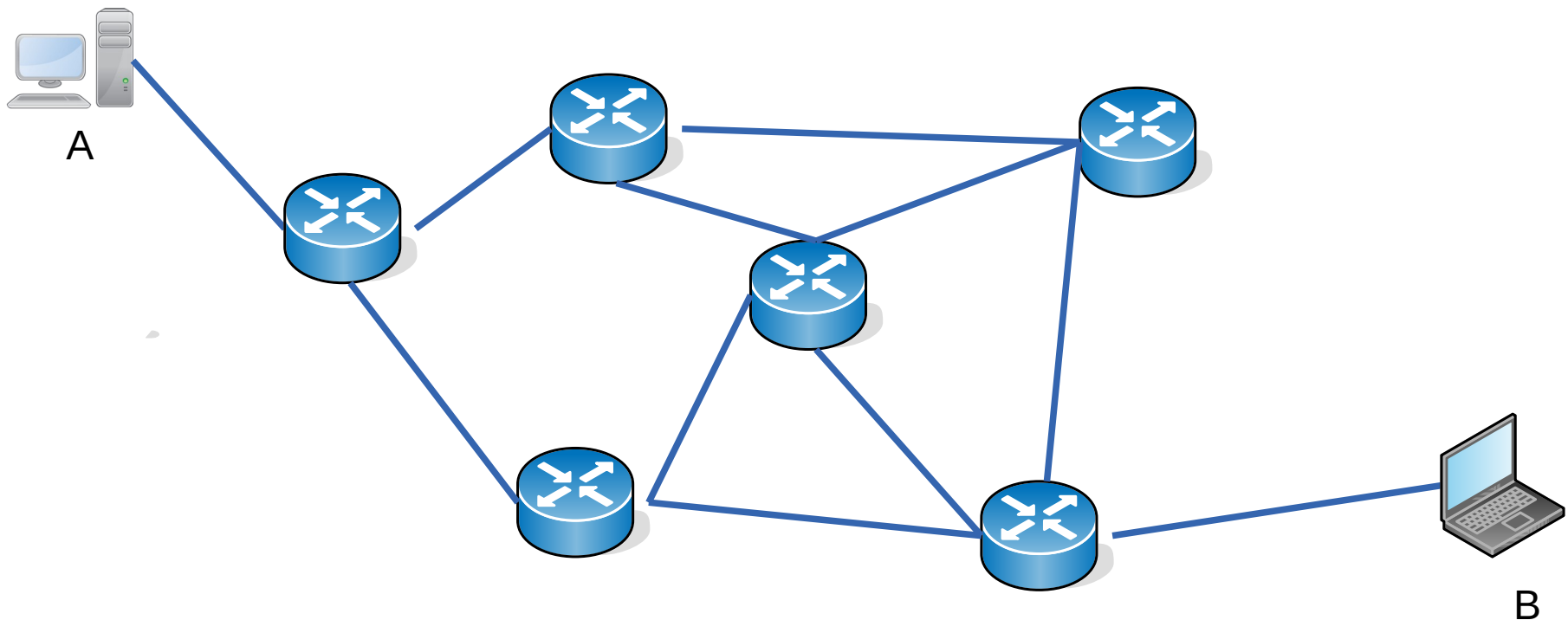
Encaminamiento o forwarding de paquetes

- Genéricamente cada nodo
 - Recibe un paquete por una de sus interfaces de entrada
 - Consulta su [tabla de forwarding](#)
 - Obtiene de la tabla el próximo salto para ese destino
 - Encamina el paquete por esa interfaz o próximo salto



Encaminamiento o forwarding de paquetes

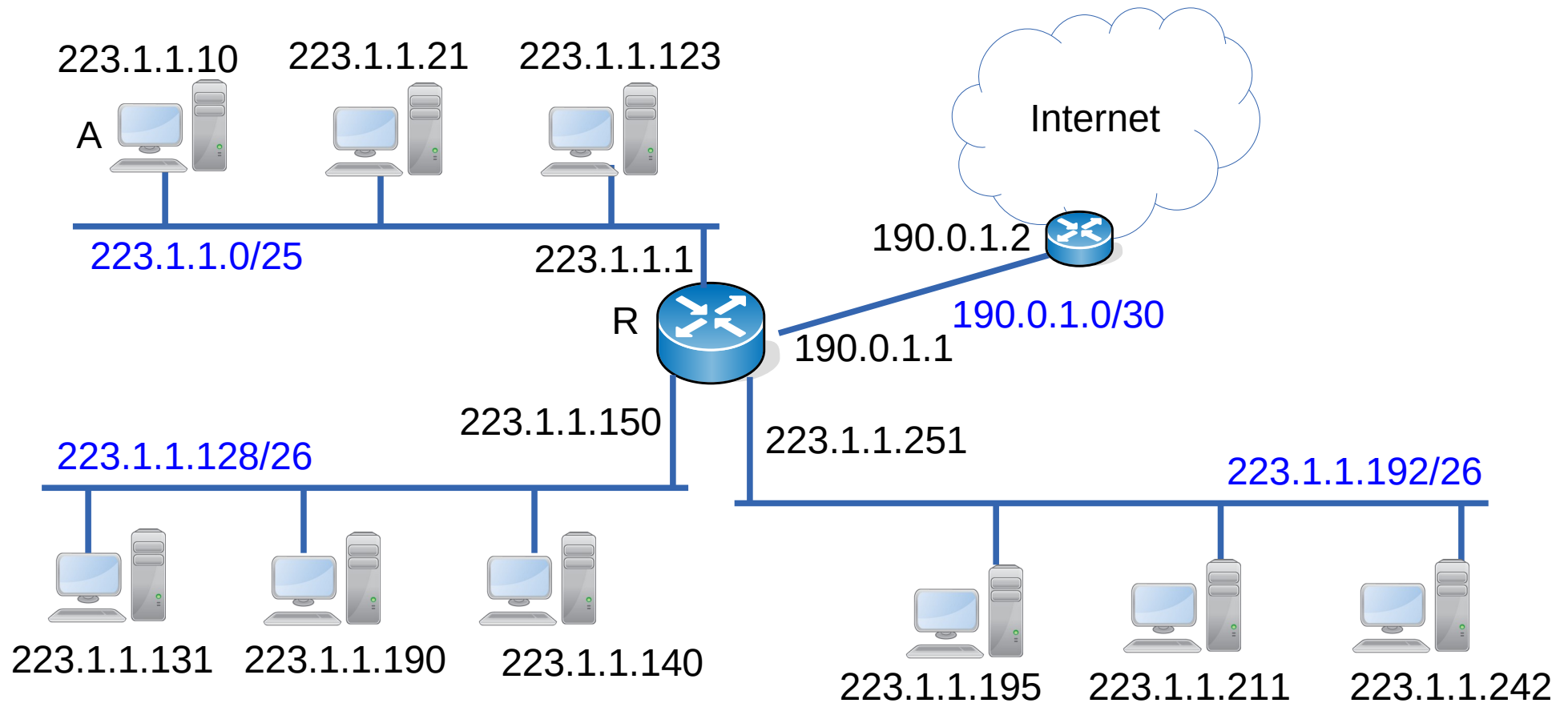
- Genéricamente cada nodo
 - Recibe un paquete por una de sus interfaces de entrada
 - Consulta su [tabla de forwarding](#)
 - Obtiene de la tabla el próximo salto para ese destino
 - Encamina el paquete por esa interfaz o próximo salto



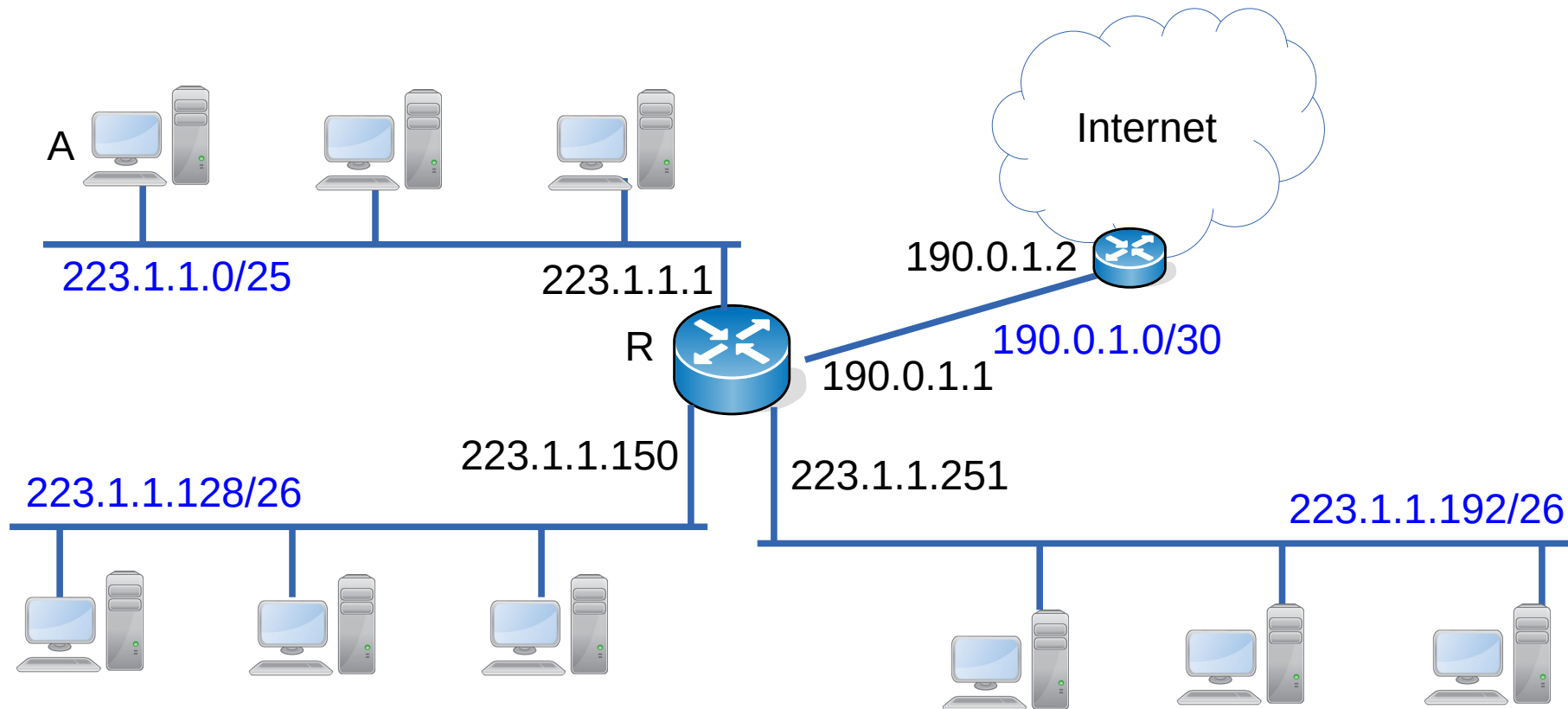
Encaminamiento basado en la dirección destino

- El encaminamiento más sencillo y más clásico es en base a la dirección destino del paquete
- Actualmente en Internet el encaminamiento del paquete se realiza consultando una **tabla de forwarding** que contiene:
 - **Destinos** especificados como rangos identificados con número de red y largo de prefijo: Formato a.b.c.d/x
 - **Próximo salto** (next hop) para cada destino
- Cada enrutador que **recibe** un paquete, **obtiene** la dirección IP de destino del encabezado, lo **busca** en la tabla de forwarding y lo **encamina** al próximo salto
- Si la IP de destino pertenece a una subred directamente conectada al equipo, entonces a ese equipo se accede directamente y se enviará usando los servicios de la capa de enlace

Ejemplo



Ejemplo



Ejemplo

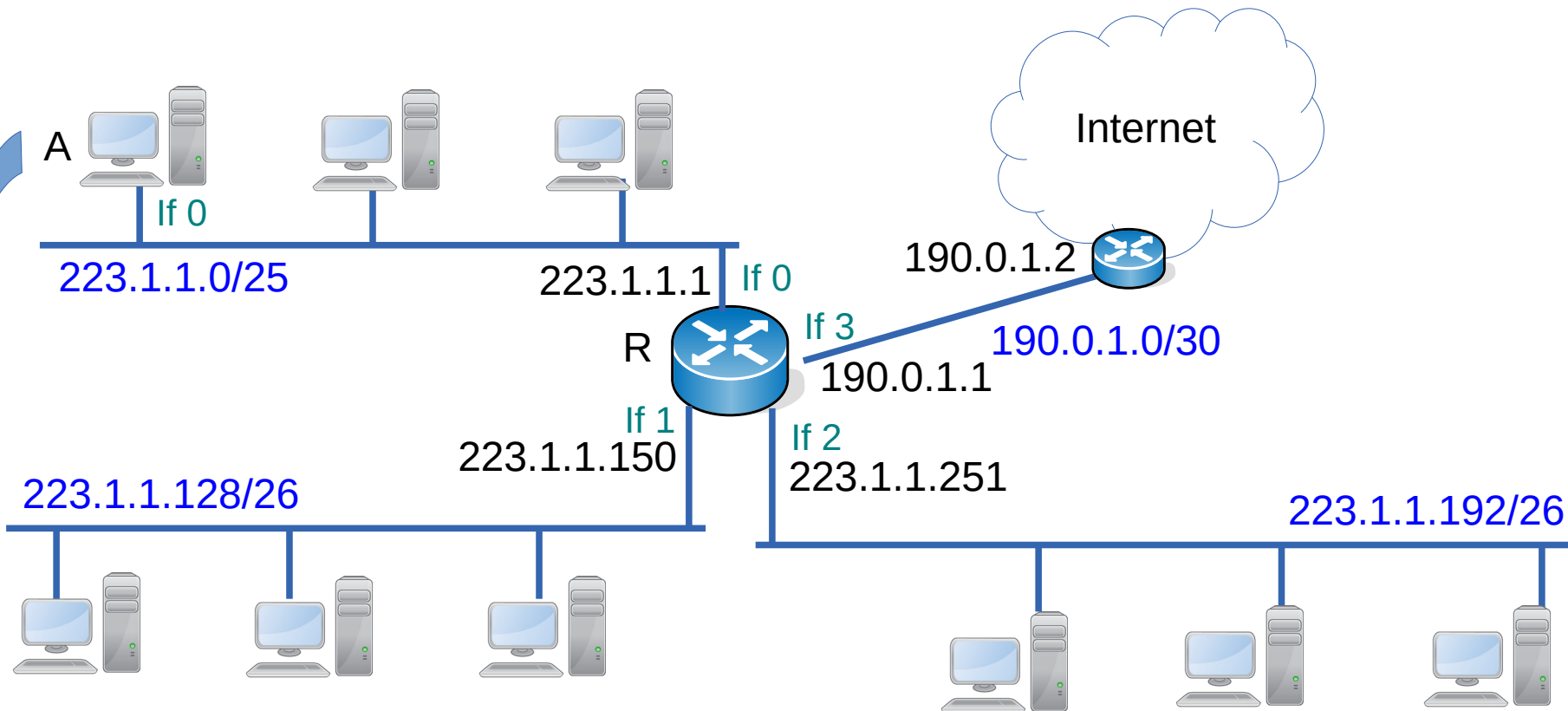


Tabla de forwarding de A	
Destino	Próximo salto
223.1.1.0/25	Local, interfaz 0 (If 0)
223.1.1.128/26	223.1.1.1
223.1.1.192/26	223.1.1.1
internet	223.1.1.1

Ejemplo

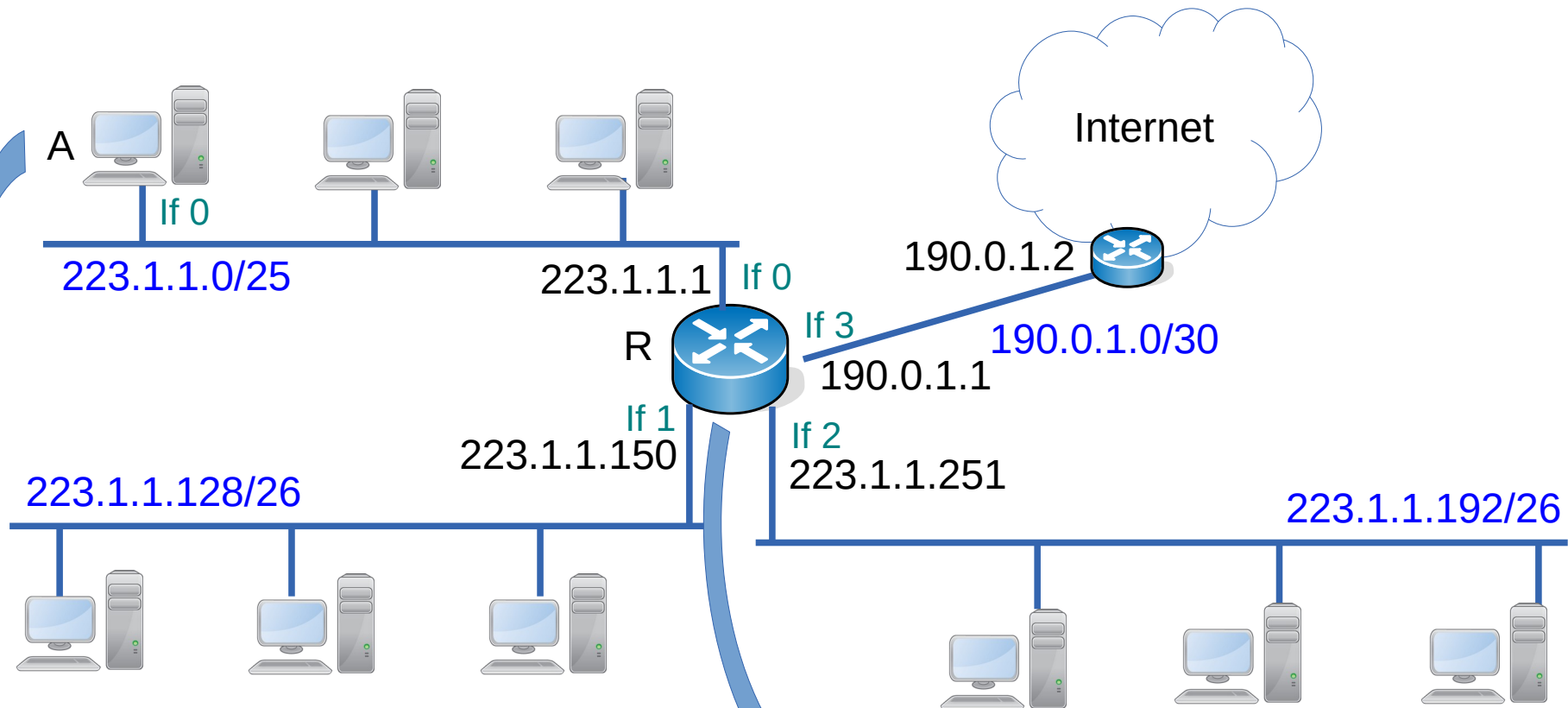


Tabla de forwarding de A	
Destino	Próximo salto
223.1.1.0/25	Local, interfaz 0 (If 0)
223.1.1.128/26	223.1.1.1
223.1.1.192/26	223.1.1.1
internet	223.1.1.1

Tabla de forwarding de R	
Destino	Próximo salto
223.1.1.0/25	Local, interfaz 0 (If 0)
223.1.1.128/26	Local, interfaz 1 (If 1)
223.1.1.192/26	Local, interfaz 2 (If 2)
190.0.1.0/30	Local, interfaz 3 (If 3)
internet	190.0.1.2

Algoritmo de búsqueda

- El algoritmo utilizado en Internet para la búsqueda de una dirección IP en la tabla de forwarding se llama: **Longest-Prefix Match**
- Los rangos que aparecen en la tabla de forwarding podrían solaparse
 - la IP que estoy buscando podría estar incluida en más de un rango
- En ese caso interesará que la búsqueda encuentre primero el rango de prefijo más largo que incluya la dirección que estamos buscando
 - Los prefijos más largos determinan rangos de menos direcciones
 - Interesa encontrar primero las rutas más específicas
- Para eso, las entradas de la tabla de forwarding deben **ordenarse** en base al **largo del prefijo** de los destinos (largo de prefijo descendente, tamaño de la red ascendente)
 - Las redes más pequeñas o más específicas estarán al principio de la tabla
 - Cada vez que se actualice la tabla, será necesario reordenarla

Algoritmo de búsqueda

- Con la tabla ordenada, cuando se necesita encaminar un paquete a un cierto destino, se **recorrerán sus filas** hasta encontrar una coincidencia
 - Si la dirección buscada está en el rango de una fila, se envía el paquete al próximo salto indicado en la fila
 - Si no, se sigue con la siguiente fila
 - Si se recorren todas las filas de la tabla y no hay coincidencia, se descarta el paquete
 - Y se envía al originador un mensaje específico del protocolo Internet Control Message Protocol (ICMP) (lo veremos más adelante)

Longest-Prefix Match

- Para saber si una dirección IP está incluida en un rango se realiza el **AND bit a bit** con la máscara del rango y el resultado se compara con la dirección de red del rango
- Si tengo el rango de direcciones: 223.1.1.128/26
 - La máscara /26 en dotted-decimal notation es **255.255.255.192**
- La dirección **223.1.1.158** pertenece a ese rango?

Longest-Prefix Match

- Para saber si una dirección IP está incluida en un rango se realiza el **AND bit a bit** con la máscara del rango y el resultado se compara con la dirección de red del rango
- Si tengo el rango de direcciones: 223.1.1.128/26
 - La máscara /26 en dotted-decimal notation es **255.255.255.192**
- La dirección **223.1.1.158** pertenece a ese rango?

1 1 0 1 1 1 1 1	0 0 0 0 0 0 0 1	0 0 0 0 0 0 0 1	1 0 0 1 1 1 1 0
-----------------	-----------------	-----------------	-----------------

223.1.1.158

Longest-Prefix Match

- Para saber si una dirección IP está incluida en un rango se realiza el **AND bit a bit** con la máscara del rango y el resultado se compara con la dirección de red del rango
- Si tengo el rango de direcciones: 223.1.1.128/26
 - La máscara /26 en dotted-decimal notation es **255.255.255.192**
- La dirección **223.1.1.158** pertenece a ese rango?

1 1 0 1 1 1 1 1	0 0 0 0 0 0 0 1	0 0 0 0 0 0 0 1	1 0 0 1 1 1 1 0	223.1.1.158
1 1 1 1 1 1 1 1	1 1 1 1 1 1 1 1	1 1 1 1 1 1 1 1	1 1 0 0 0 0 0 0	255.255.255.192

Longest-Prefix Match

- Para saber si una dirección IP está incluida en un rango se realiza el **AND bit a bit** con la máscara del rango y el resultado se compara con la dirección de red del rango
- Si tengo el rango de direcciones: 223.1.1.128/26
 - La máscara /26 en dotted-decimal notation es 255.255.255.192
- La dirección 223.1.1.158 pertenece a ese rango?

AND	1 1 0 1 1 1 1 1	0 0 0 0 0 0 0 1	0 0 0 0 0 0 0 1	1 0 0 1 1 1 1 0	223.1.1.158
	1 1 1 1 1 1 1 1	1 1 1 1 1 1 1 1	1 1 1 1 1 1 1 1	1 1 0 0 0 0 0 0	255.255.255.192

Longest-Prefix Match

- Para saber si una dirección IP está incluida en un rango se realiza el **AND bit a bit** con la máscara del rango y el resultado se compara con la dirección de red del rango
- Si tengo el rango de direcciones: 223.1.1.128/26
 - La máscara /26 en dotted-decimal notation es 255.255.255.192
- La dirección 223.1.1.158 pertenece a ese rango?

AND	1 1 0 1 1 1 1 1	0 0 0 0 0 0 0 1	0 0 0 0 0 0 0 1	1 0 0 1 1 1 1 0	223.1.1.158
	1 1 1 1 1 1 1 1	1 1 1 1 1 1 1 1	1 1 1 1 1 1 1 1	1 1 0 0 0 0 0 0	255.255.255.192
	1 1 0 1 1 1 1 1	0 0 0 0 0 0 0 1	0 0 0 0 0 0 0 1	1 0 0 0 0 0 0 0	223.1.1.128

Longest-Prefix Match

- Para saber si una dirección IP está incluida en un rango se realiza el **AND bit a bit** con la máscara del rango y el resultado se compara con la dirección de red del rango
- Si tengo el rango de direcciones: **223.1.1.128/26**
 - La máscara /26 en dotted-decimal notation es **255.255.255.192**
- La dirección **223.1.1.158** pertenece a ese rango?

AND	1 1 0 1 1 1 1 1	0 0 0 0 0 0 0 1	0 0 0 0 0 0 0 1	1 0 0 1 1 1 1 0	223.1.1.158
	1 1 1 1 1 1 1 1	1 1 1 1 1 1 1 1	1 1 1 1 1 1 1 1	1 1 0 0 0 0 0 0	255.255.255.192
	1 1 0 1 1 1 1 1	0 0 0 0 0 0 0 1	0 0 0 0 0 0 0 1	1 0 0 0 0 0 0 0	223.1.1.128

La base del rango coincide!

La IP buscada está comprendida en ese rango

Longest-Prefix Match

- La tabla de forwarding contiene filas o entradas con:
 - Rangos de direcciones especificados con subred S_i y máscara M_i
 - Próximo salto o next hop N_i para ese destino

Destino		Próximo salto
S_i	M_i	N_i

- Como **paso previo** la tabla debe estar **ordenada** en forma descendente por el largo de M_i
- Si se requiere encaminar un paquete cuya Dirección IP destino es D
 - Se recorre la tabla hasta **encontrar** el primer rango que incluya a D
mientras (hay entradas)
si ($S_i == D$ AND M_i)
entregar el paquete a capa 2 para que lo haga llegar a N_i
 - Si no se encuentran coincidencias descartar paquete y enviar mensaje ICMP al originador del paquete

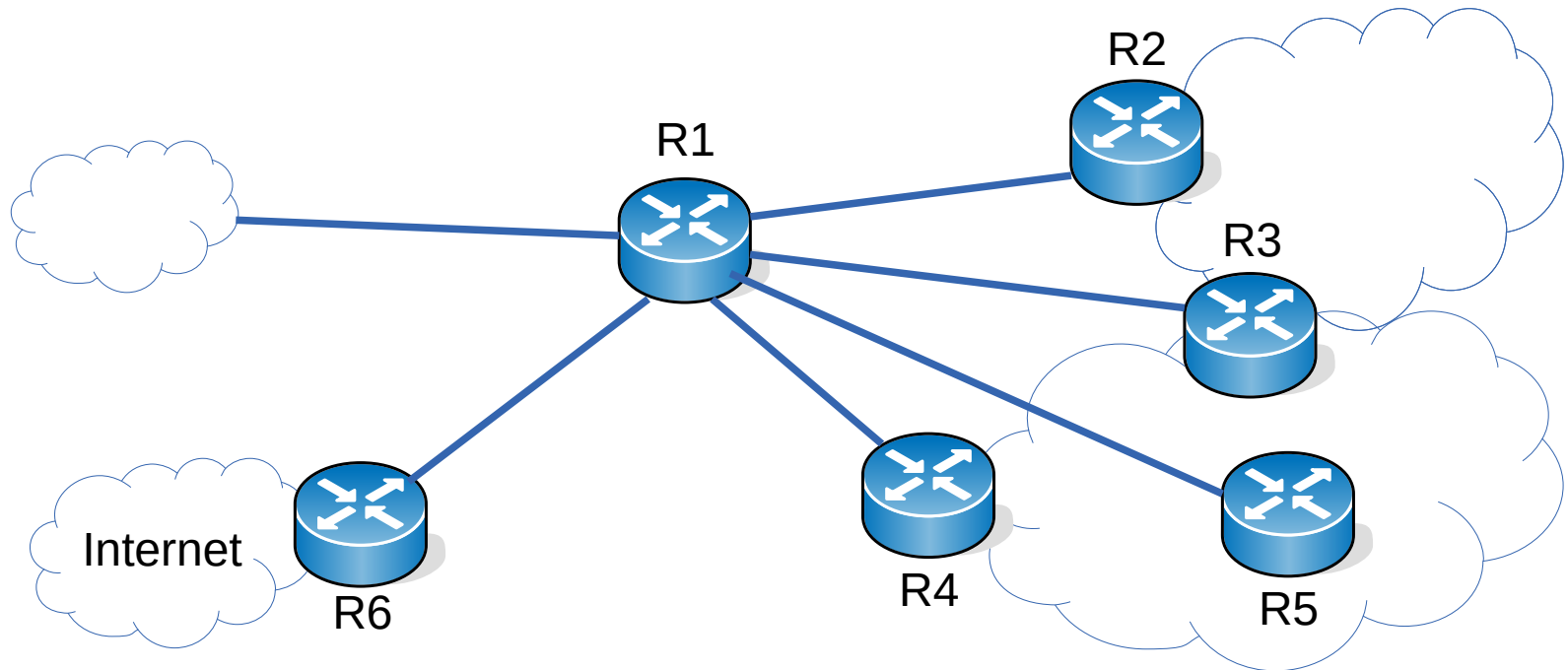
Longest-Prefix Match

- Puede existir una **ruta por defecto**
 - ruta que tomarán todos los paquetes que no hayan encontrado una coincidencia con ninguna entrada de la tabla
- Esa ruta naturalmente debería ir al final de la tabla
- Para lograr eso, se agrega una entrada:

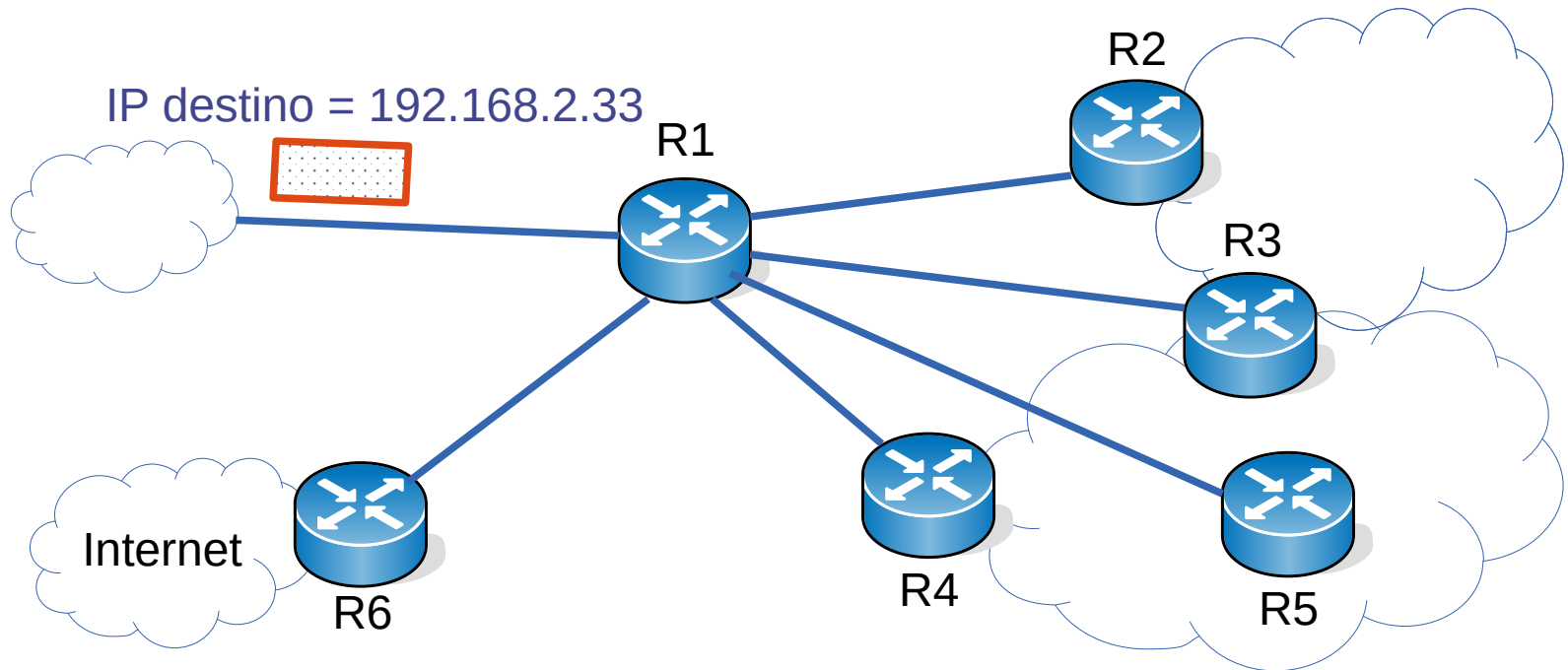
Destino	Próximo salto
0.0.0.0/0	IP por defecto

- Como tiene máscara de largo 0, al ordenar la tabla quedará al final
- En base al algoritmo, para cualquier dirección IP se obtendrá una coincidencia
- En caso de existir ruta por defecto, nunca habrá descartes de paquetes por no encontrar coincidencias en la tabla
- En la mayoría de los casos la ruta por defecto es necesaria, pero existe un conjunto de enrutadores de Internet que no tienen rutas por defecto. Son los que pertenecen a la “zona libre de rutas por defecto” (default-free zone, DFZ).
- Estos equipos conocen todos los rangos de redes en uso y evitan que paquetes destinados a IPs no asignadas queden en “loop”

Ejemplo



Ejemplo



Ejemplo

Ojo! Esta
tabla no está
completa!
Al menos le
faltan las
redes
directamente
conectadas

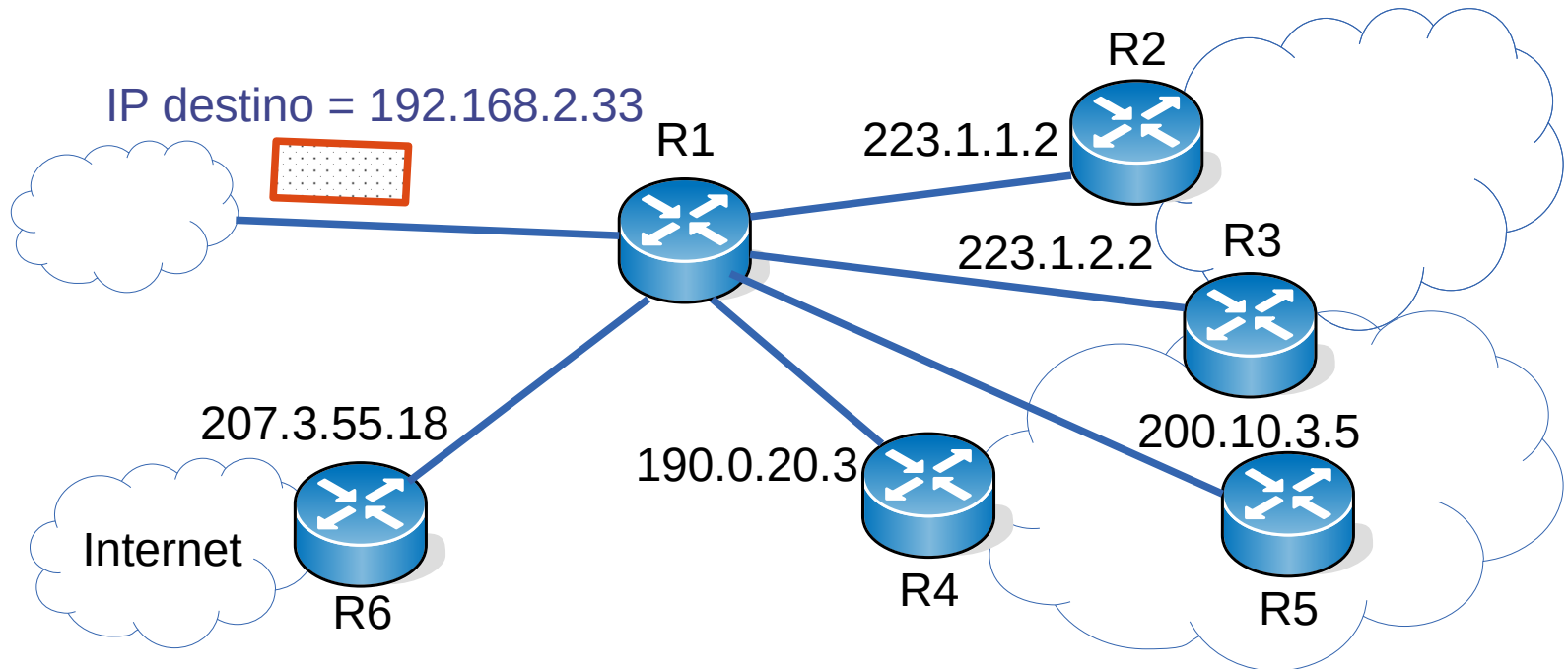
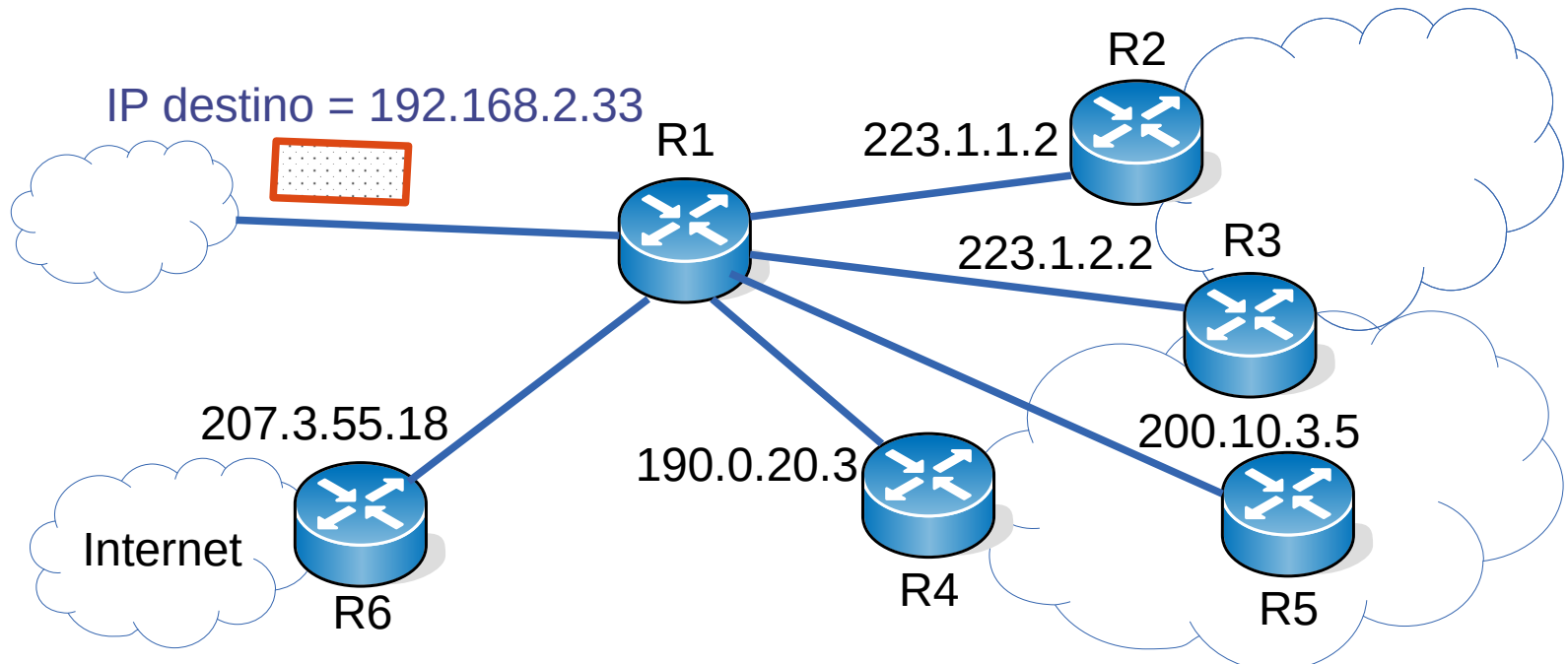


Tabla de R1	
Destino	Próximo salto
192.168.1.128/25	223.1.1.2
192.168.0.0/23	223.1.2.2
192.168.0.0/22	190.0.20.3
192.0.0.0/8	200.10.3.5
0.0.0.0/0	207.3.55.18

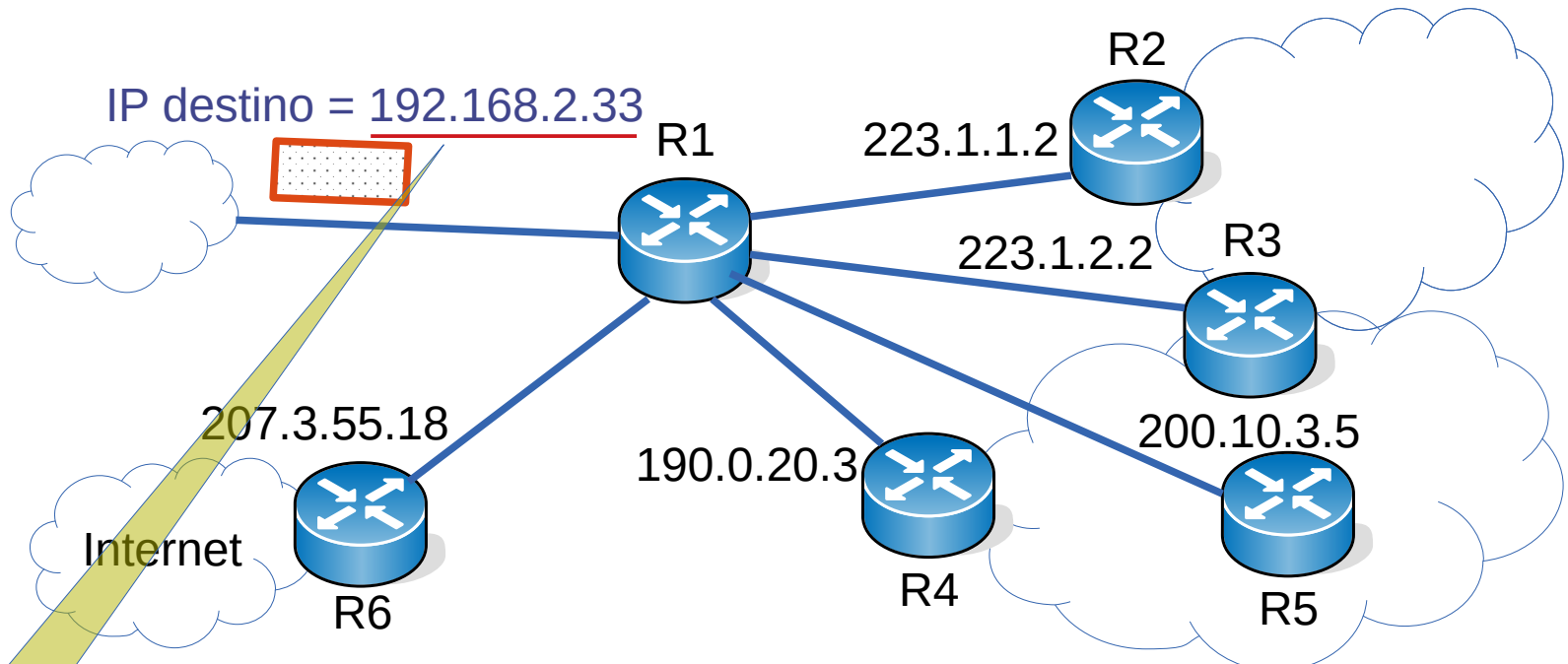
Ejemplo



192.168.2.33 AND 255.255.255.128 =

Tabla de R1	
Destino	Próximo salto
192.168.1.128/25	223.1.1.2
192.168.0.0/23	223.1.2.2
192.168.0.0/22	190.0.20.3
192.0.0.0/8	200.10.3.5
0.0.0.0/0	207.3.55.18

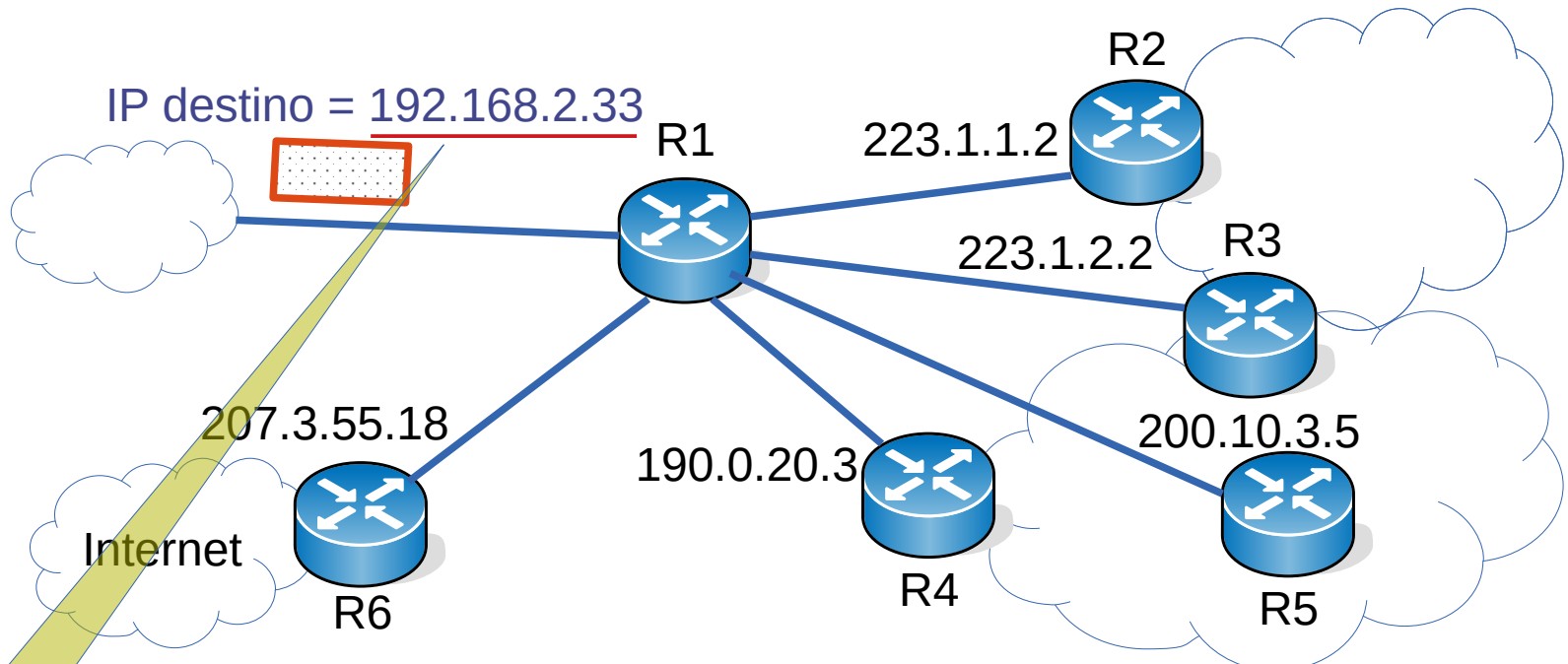
Ejemplo



192.168.2.33 AND 255.255.255.128 =

Tabla de R1	
Destino	Próximo salto
192.168.1.128/25	223.1.1.2
192.168.0.0/23	223.1.2.2
192.168.0.0/22	190.0.20.3
192.0.0.0/8	200.10.3.5
0.0.0.0/0	207.3.55.18

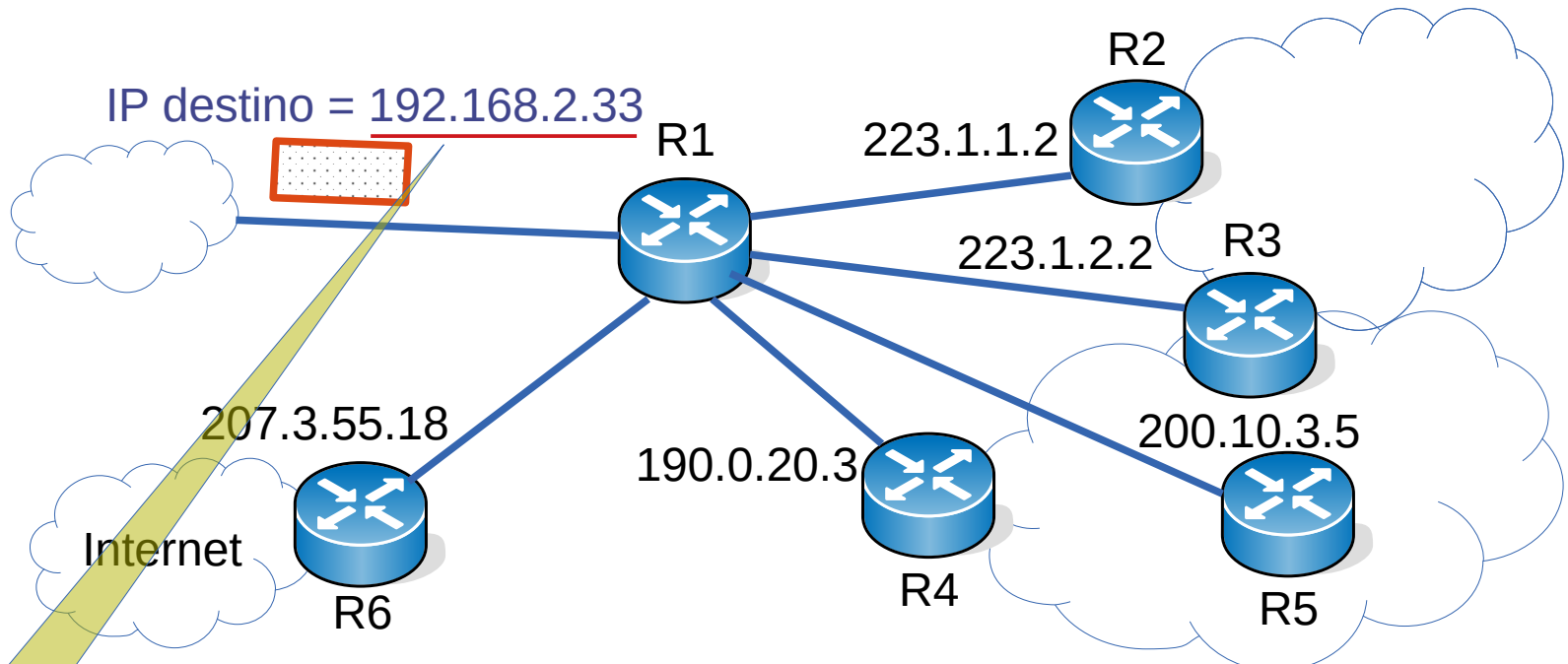
Ejemplo



192.168.2.33 AND 255.255.255.128 =

Tabla de R1	
Destino	Próximo salto
192.168.1.128/ <u>25</u>	223.1.1.2
192.168.0.0/23	223.1.2.2
192.168.0.0/22	190.0.20.3
192.0.0.0/8	200.10.3.5
0.0.0.0/0	207.3.55.18

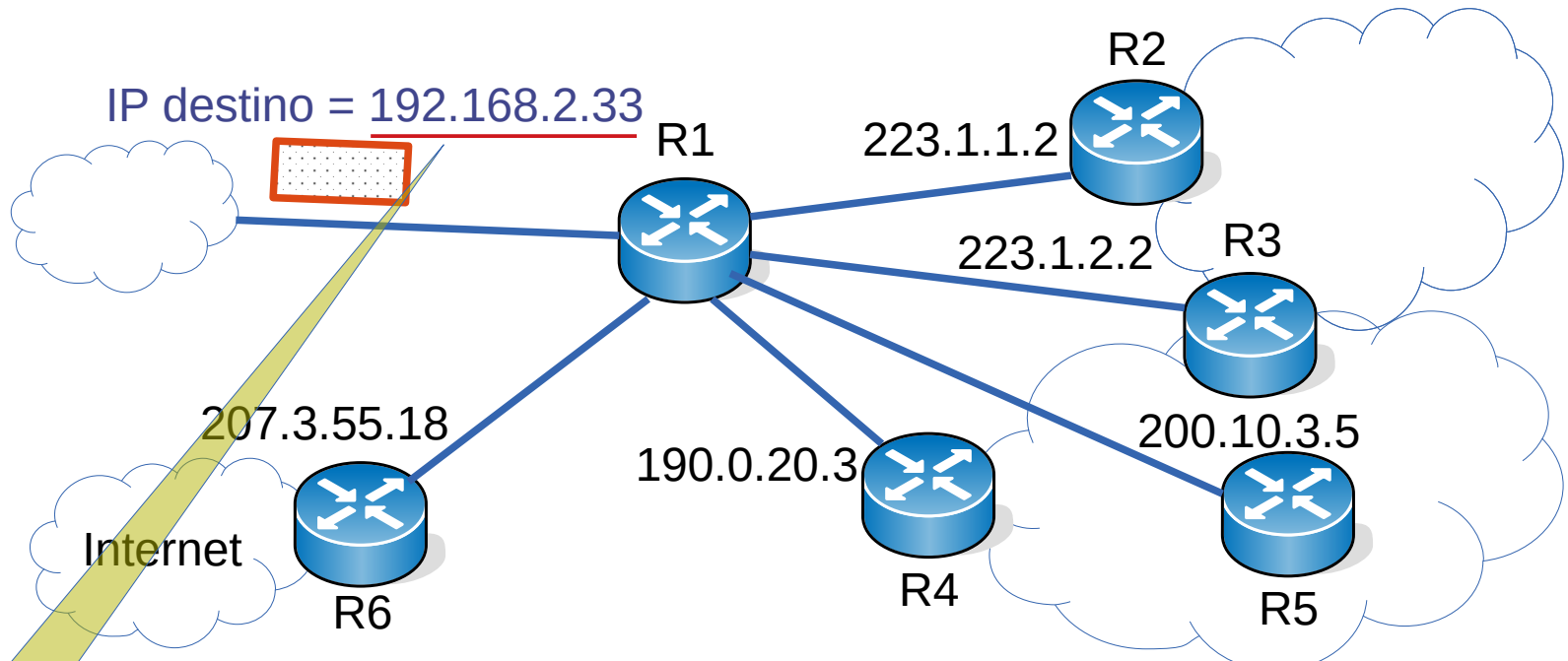
Ejemplo



192.168.2.33 AND 255.255.255.128 = 192.168.2.0

Tabla de R1	
Destino	Próximo salto
192.168.1.128/ <u>25</u>	223.1.1.2
192.168.0.0/23	223.1.2.2
192.168.0.0/22	190.0.20.3
192.0.0.0/8	200.10.3.5
0.0.0.0/0	207.3.55.18

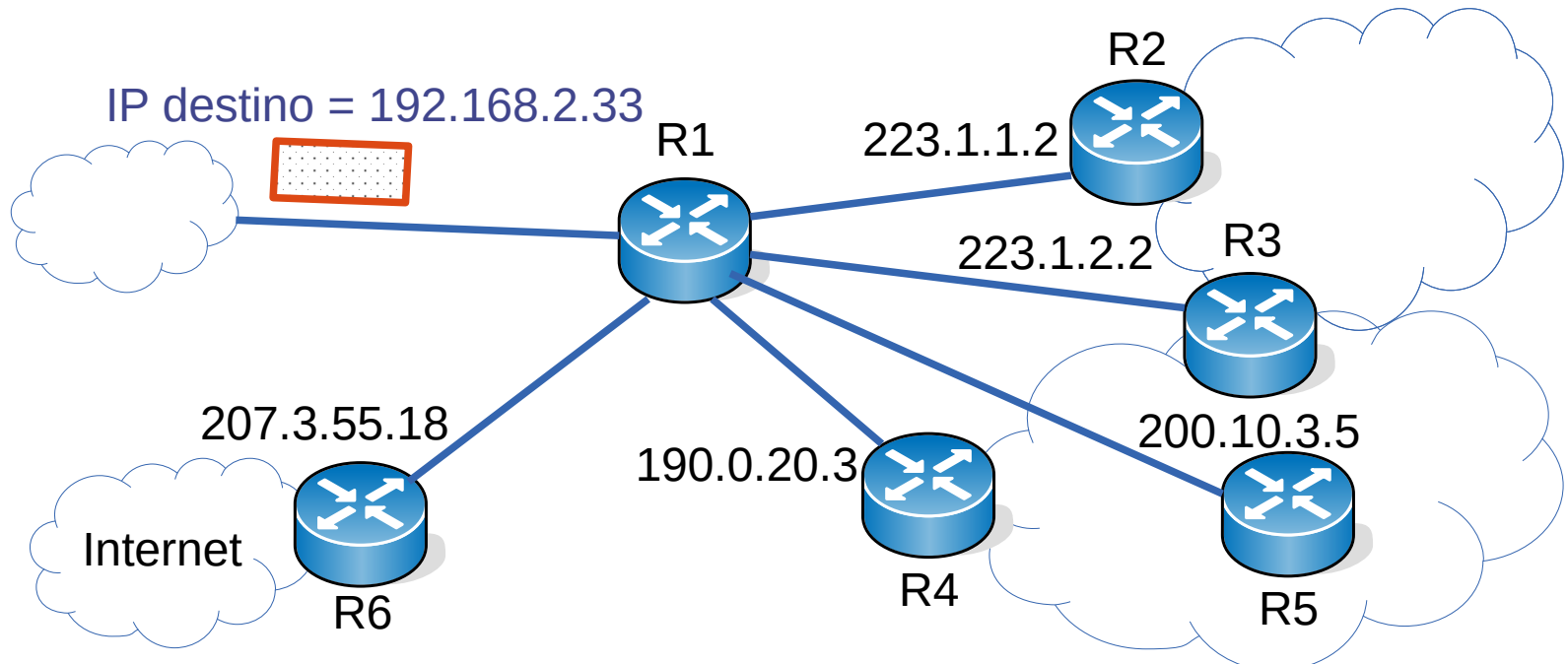
Ejemplo



192.168.2.33 AND 255.255.255.128 = 192.168.2.0 Distinto de 192.168.1.128

Tabla de R1	
Destino	Próximo salto
<u>192.168.1.128/25</u>	223.1.1.2
192.168.0.0/23	223.1.2.2
192.168.0.0/22	190.0.20.3
192.0.0.0/8	200.10.3.5
0.0.0.0/0	207.3.55.18

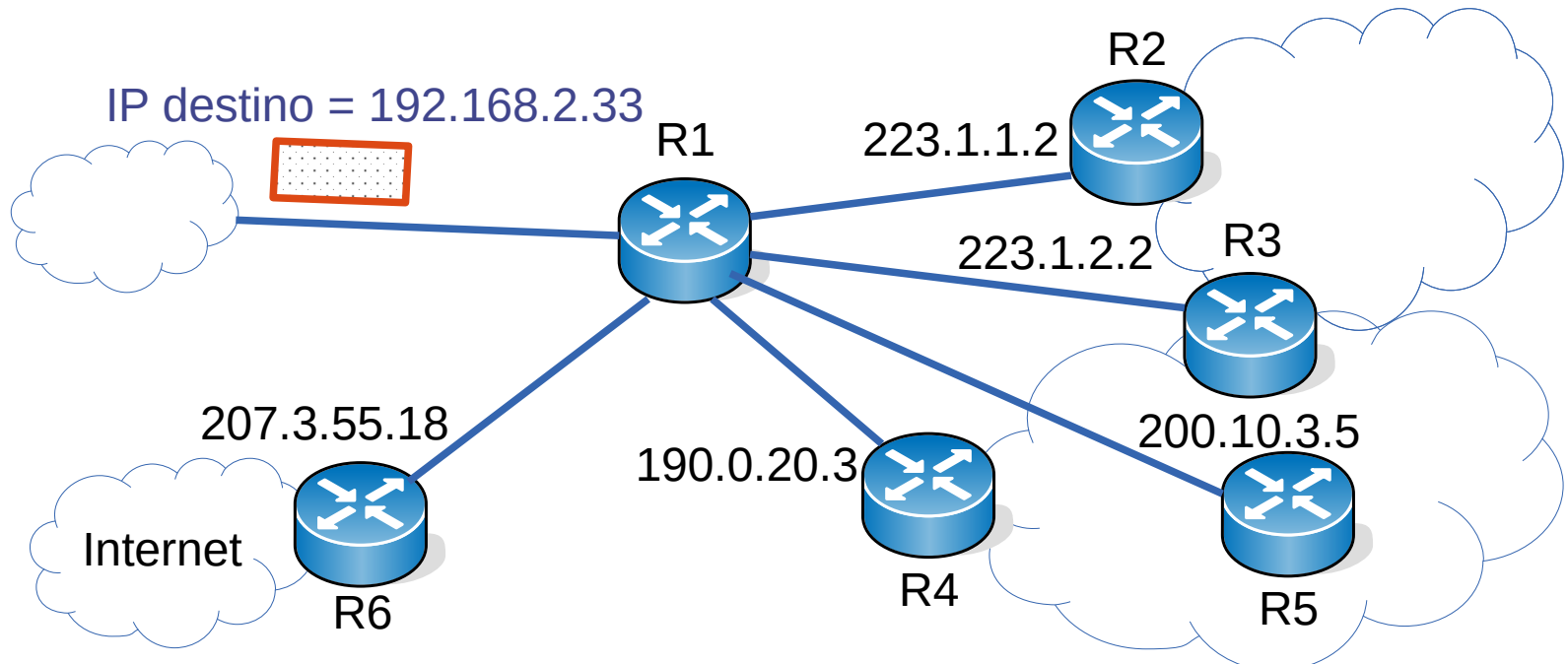
Ejemplo



192.168.2.33 AND 255.255.254.0 =

Tabla de R1	
Destino	Próximo salto
192.168.1.128/25	223.1.1.2
192.168.0.0/23	223.1.2.2
192.168.0.0/22	190.0.20.3
192.0.0.0/8	200.10.3.5
0.0.0.0/0	207.3.55.18

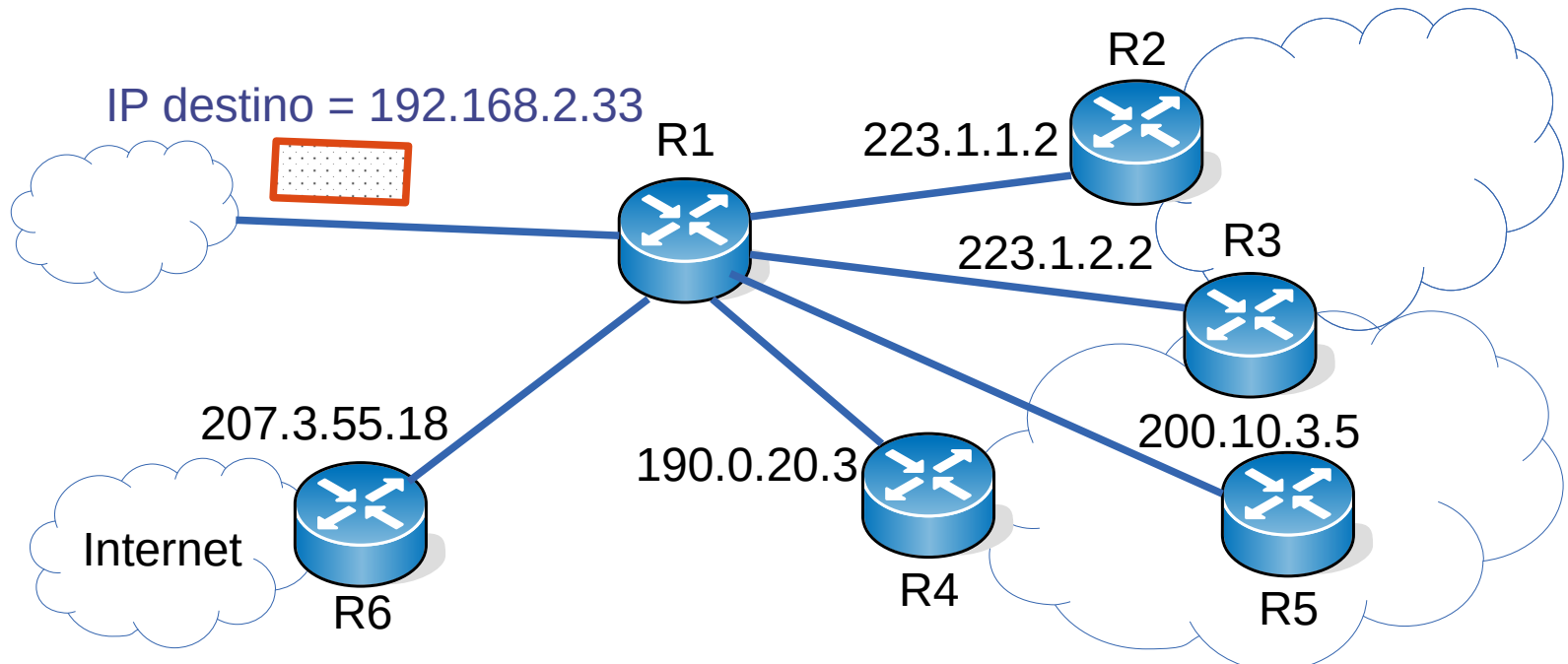
Ejemplo



192.168.2.33 AND 255.255.254.0 = 192.168.2.0

Tabla de R1	
Destino	Próximo salto
192.168.1.128/25	223.1.1.2
192.168.0.0/23	223.1.2.2
192.168.0.0/22	190.0.20.3
192.0.0.0/8	200.10.3.5
0.0.0.0/0	207.3.55.18

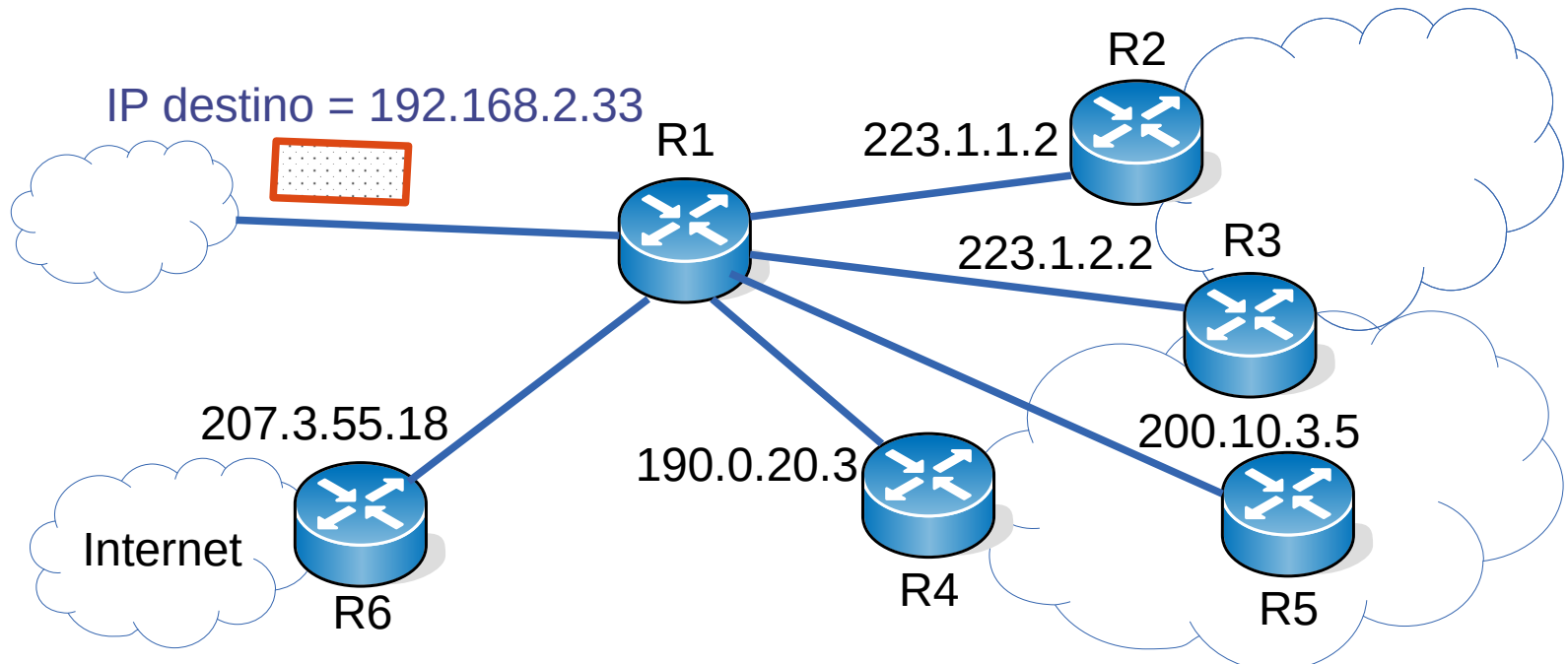
Ejemplo



192.168.2.33 AND 255.255.254.0 = 192.168.2.0 Distinto de 192.168.0.0

Tabla de R1	
Destino	Próximo salto
192.168.1.128/25	223.1.1.2
192.168.0.0/23	223.1.2.2
192.168.0.0/22	190.0.20.3
192.0.0.0/8	200.10.3.5
0.0.0.0/0	207.3.55.18

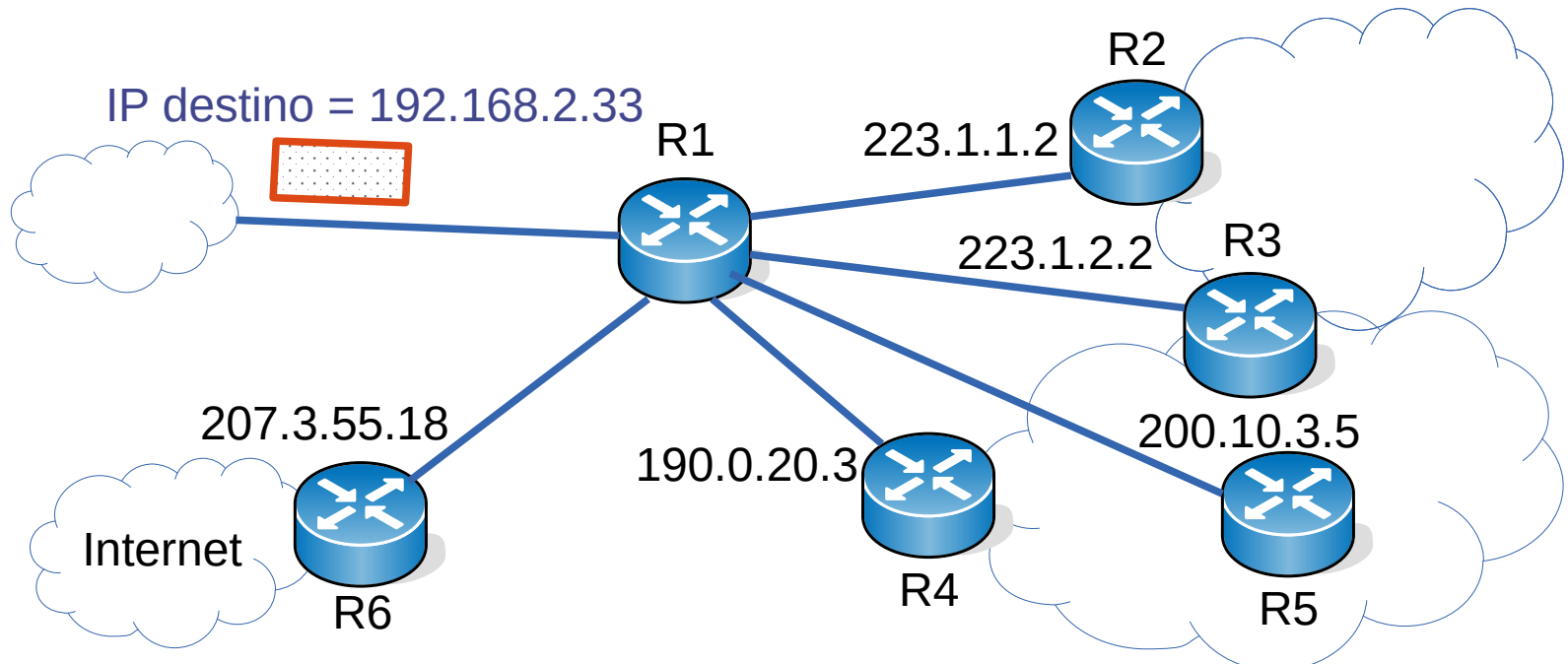
Ejemplo



192.168.2.33 AND 255.255.252.0 =

Tabla de R1	
Destino	Próximo salto
192.168.1.128/25	223.1.1.2
192.168.0.0/23	223.1.2.2
192.168.0.0/22	190.0.20.3
192.0.0.0/8	200.10.3.5
0.0.0.0/0	207.3.55.18

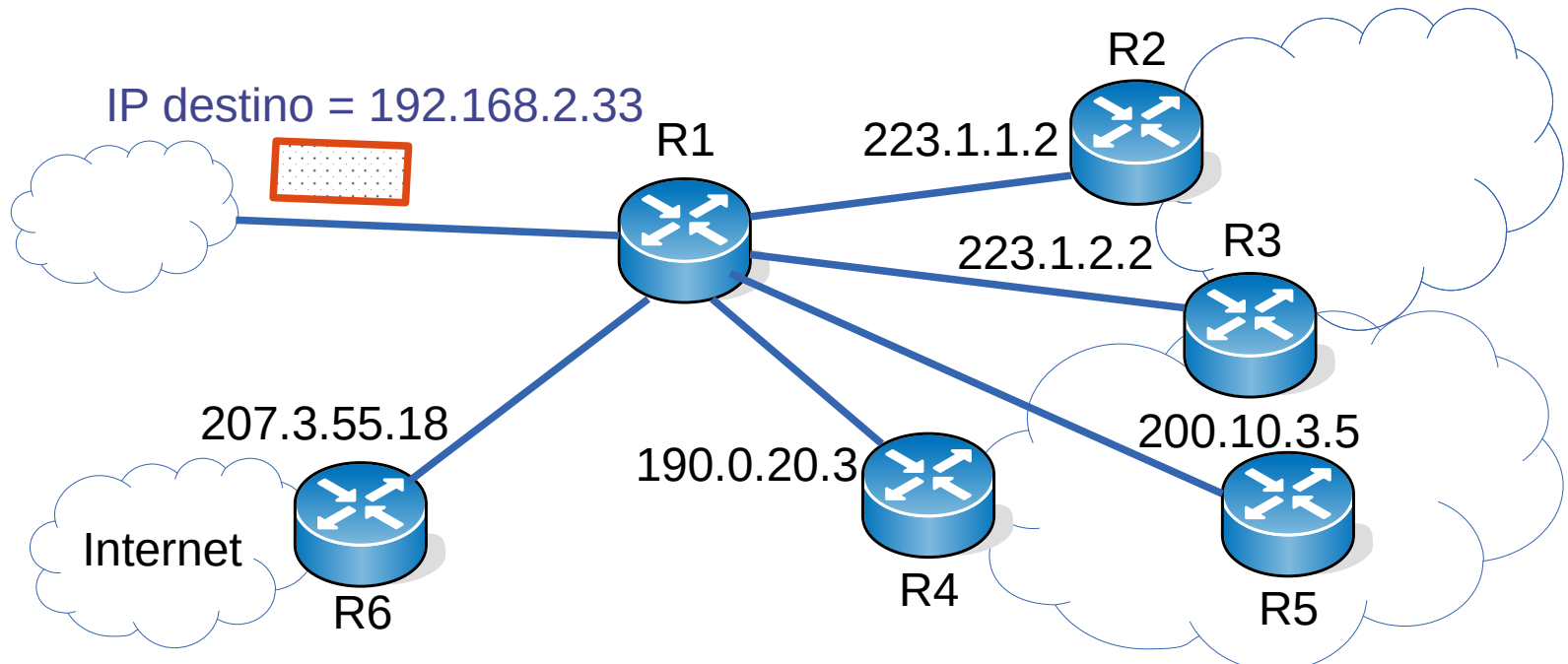
Ejemplo



192.168.2.33 AND 255.255.252.0 = 192.168.0.0

Tabla de R1	
Destino	Próximo salto
192.168.1.128/25	223.1.1.2
192.168.0.0/23	223.1.2.2
192.168.0.0/22	190.0.20.3
192.0.0.0/8	200.10.3.5
0.0.0.0/0	207.3.55.18

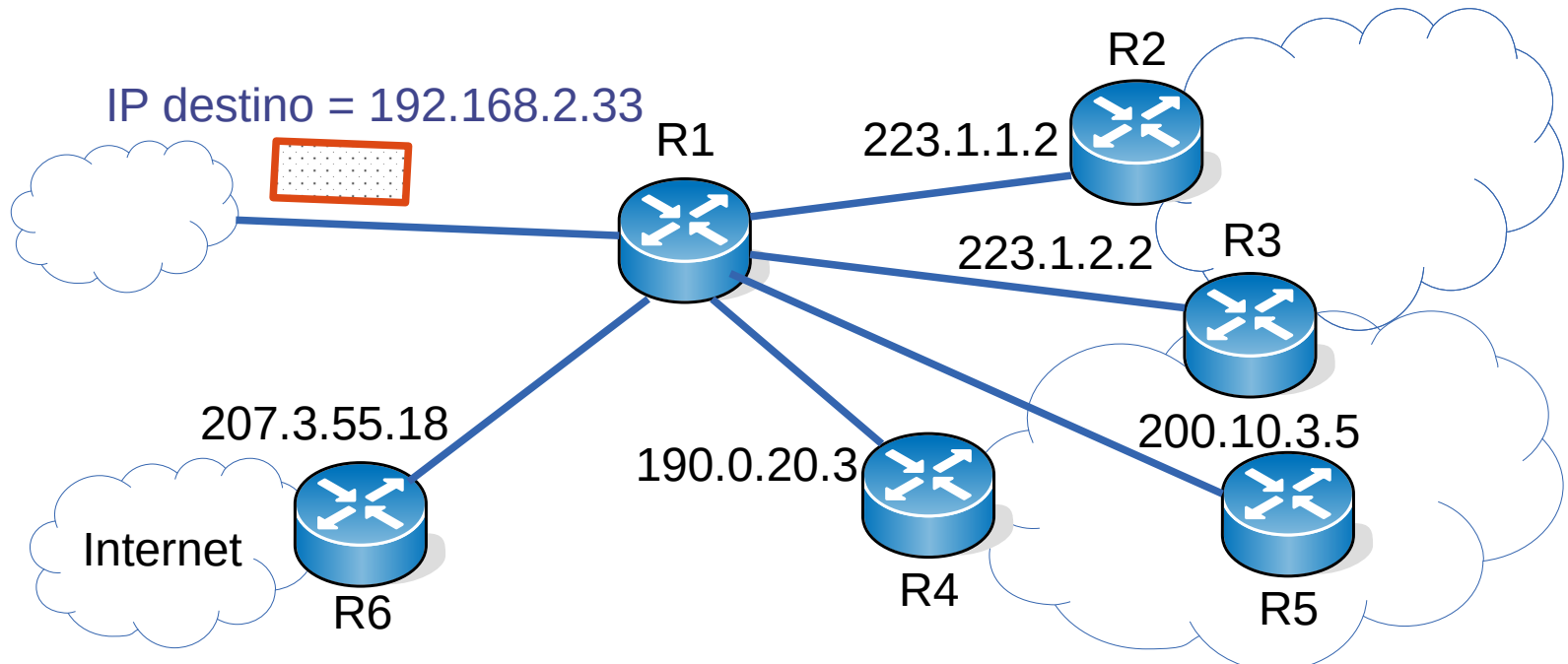
Ejemplo



192.168.2.33 AND 255.255.252.0 = 192.168.0.0 Coincide con 192.168.0.0

Tabla de R1	
Destino	Próximo salto
192.168.1.128/25	223.1.1.2
192.168.0.0/23	223.1.2.2
192.168.0.0/22	190.0.20.3
192.0.0.0/8	200.10.3.5
0.0.0.0/0	207.3.55.18

Ejemplo

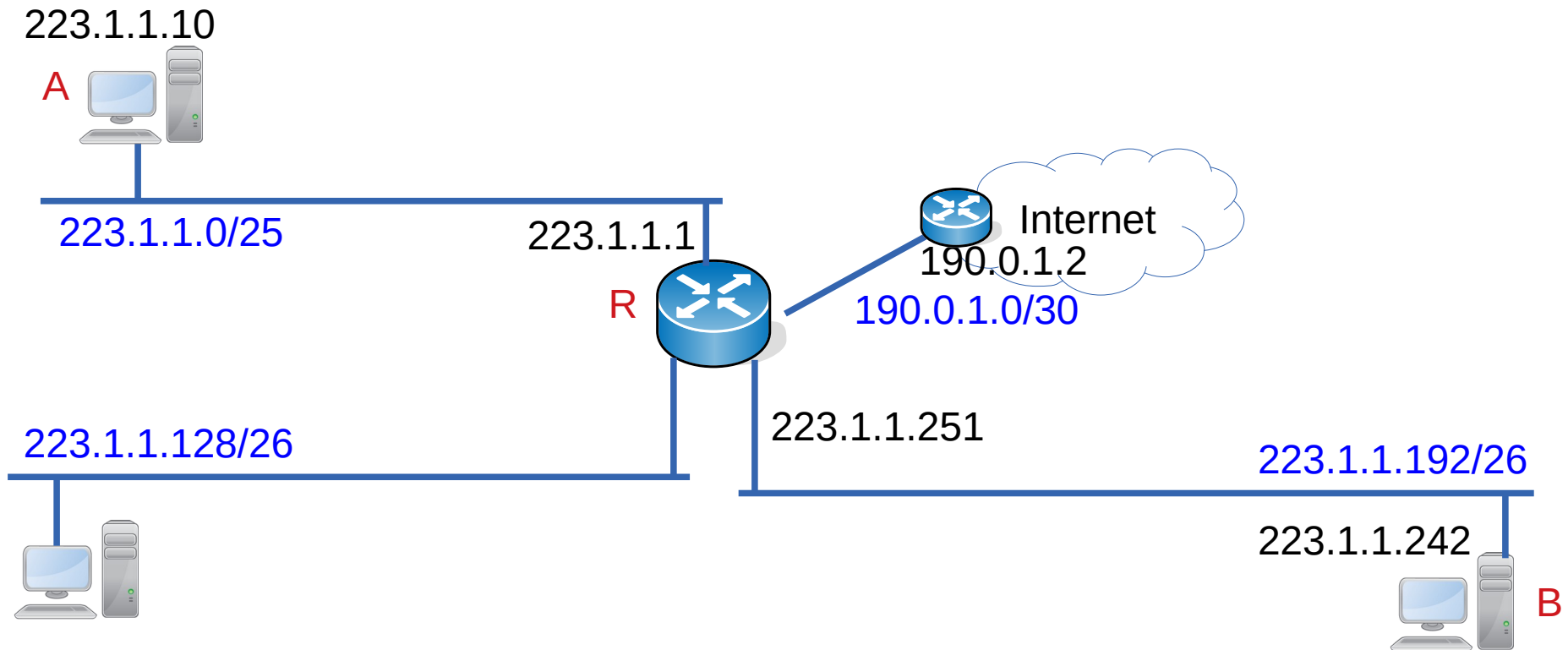


192.168.2.33 AND 255.255.252.0 = 192.168.0.0 Coincide con 192.168.0.0

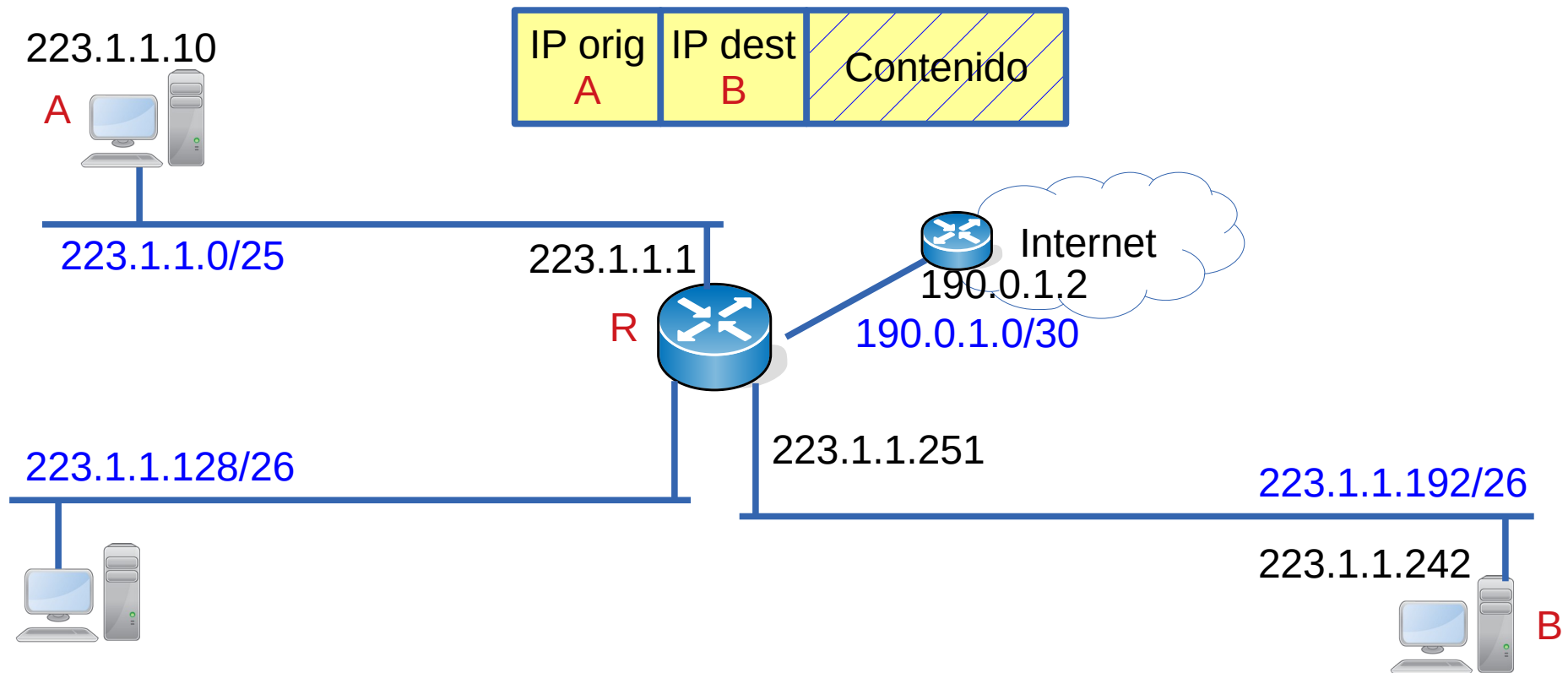
Tabla de R1	
Destino	Próximo salto
192.168.1.128/25	223.1.1.2
192.168.0.0/23	223.1.2.2
192.168.0.0/22	<u>190.0.20.3</u>
192.0.0.0/8	200.10.3.5
0.0.0.0/0	207.3.55.18

Envío el
Paquete a
190.0.20.3
(R4)

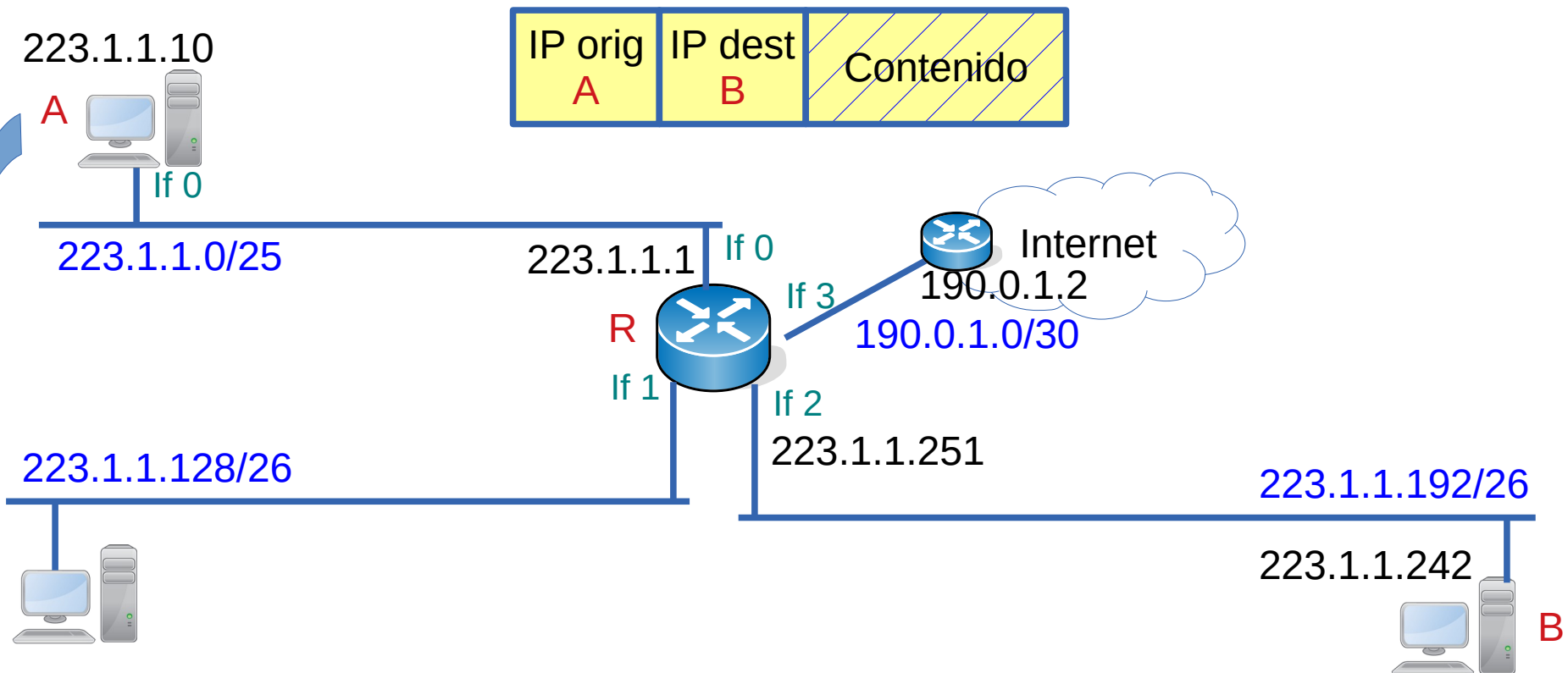
¿Qué pasa cuando A quiere enviar un paquete a B?



¿Qué pasa cuando A quiere enviar un paquete a B?



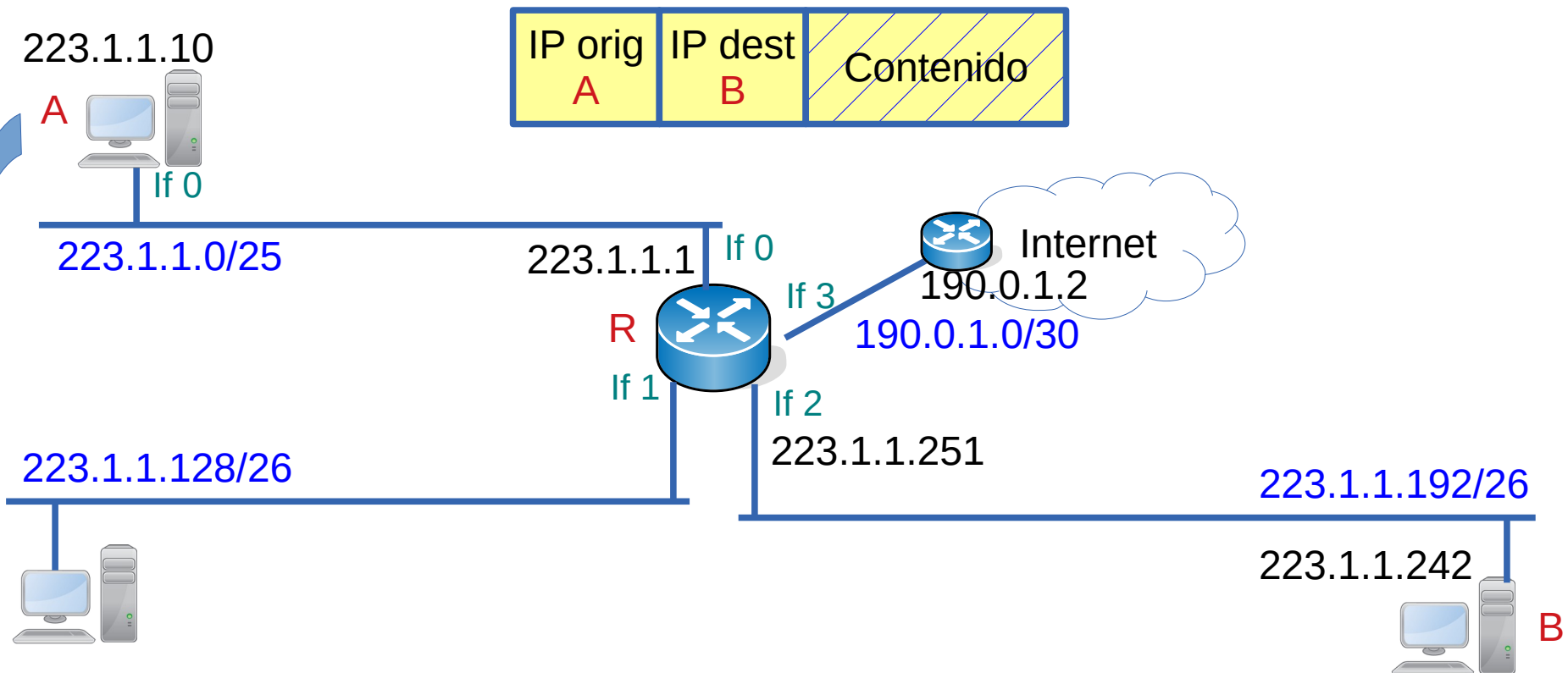
¿Qué pasa cuando A quiere enviar un paquete a B?



A consulta su tabla de forwarding para saber cómo llegar a B!

Destino	Próximo salto
223.1.1.128/26	223.1.1.1
223.1.1.192/26	223.1.1.1
223.1.1.0/25	Local, interfaz 0 (If 0)
0.0.0.0/0	223.1.1.1

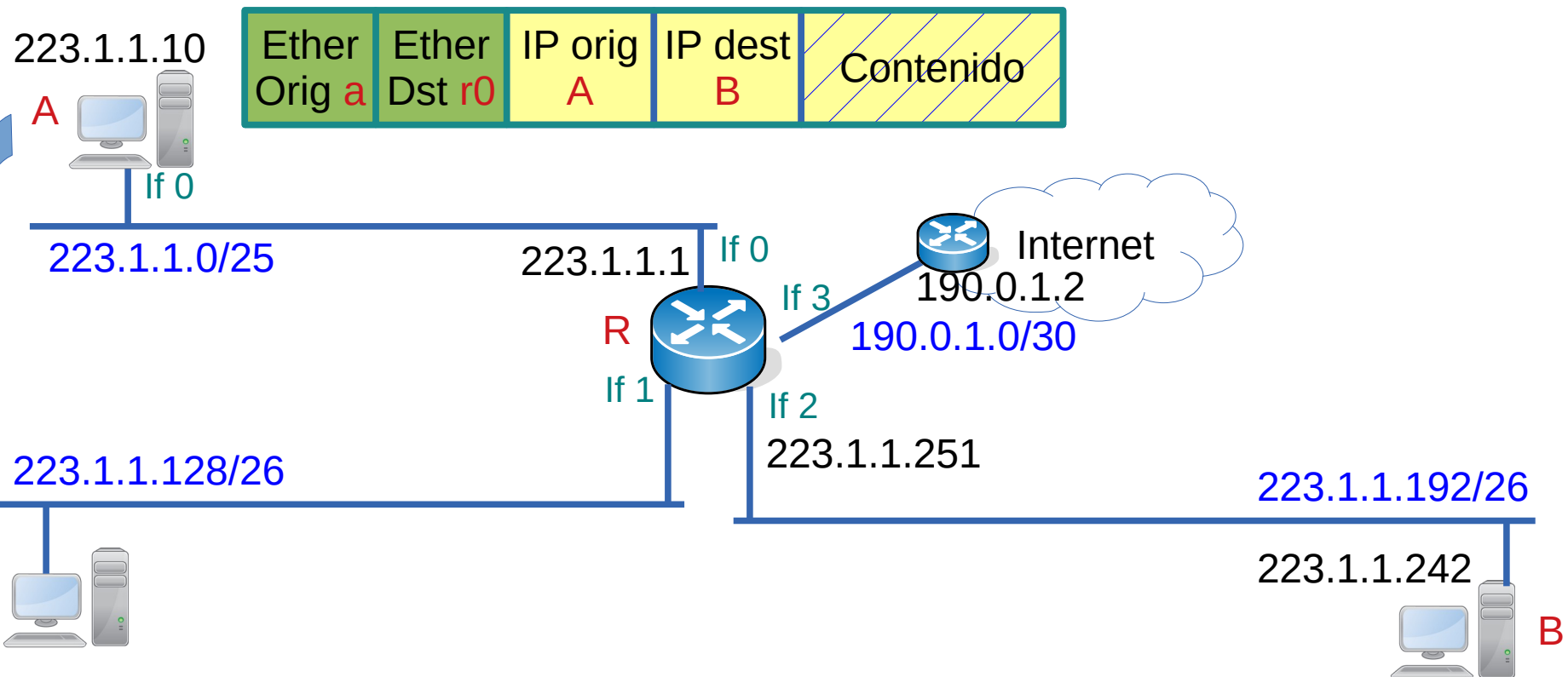
¿Qué pasa cuando A quiere enviar un paquete a B?



A consulta su tabla de forwarding para saber cómo llegar a B!

Destino	Próximo salto
223.1.1.128/26	223.1.1.1
223.1.1.192/26	223.1.1.1
223.1.1.0/25	Local, interfaz 0 (If 0)
0.0.0.0/0	223.1.1.1

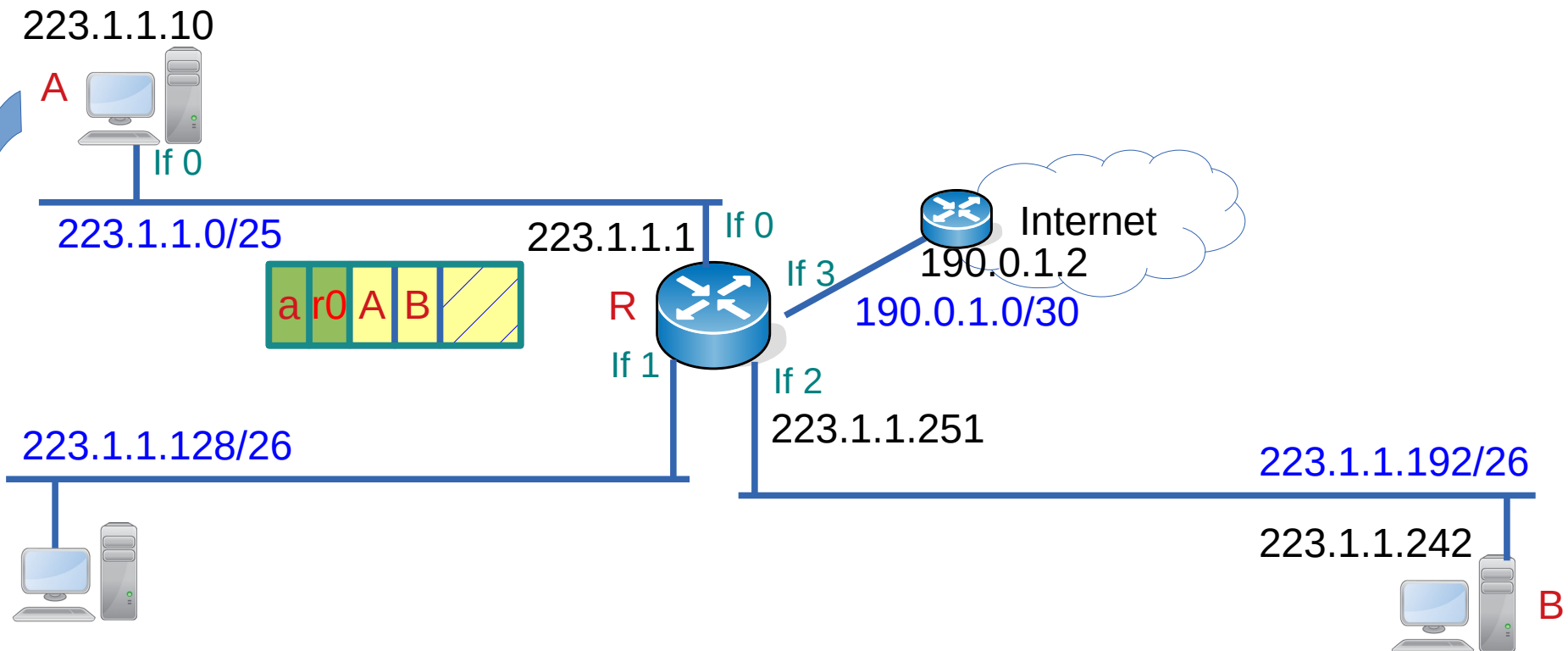
¿Qué pasa cuando A quiere enviar un paquete a B?



A consulta su tabla de forwarding para saber cómo llegar a B!

Destino	Próximo salto
223.1.1.128/26	223.1.1.1
223.1.1.192/26	223.1.1.1
223.1.1.0/25	Local, interfaz 0 (If 0)
0.0.0.0/0	223.1.1.1

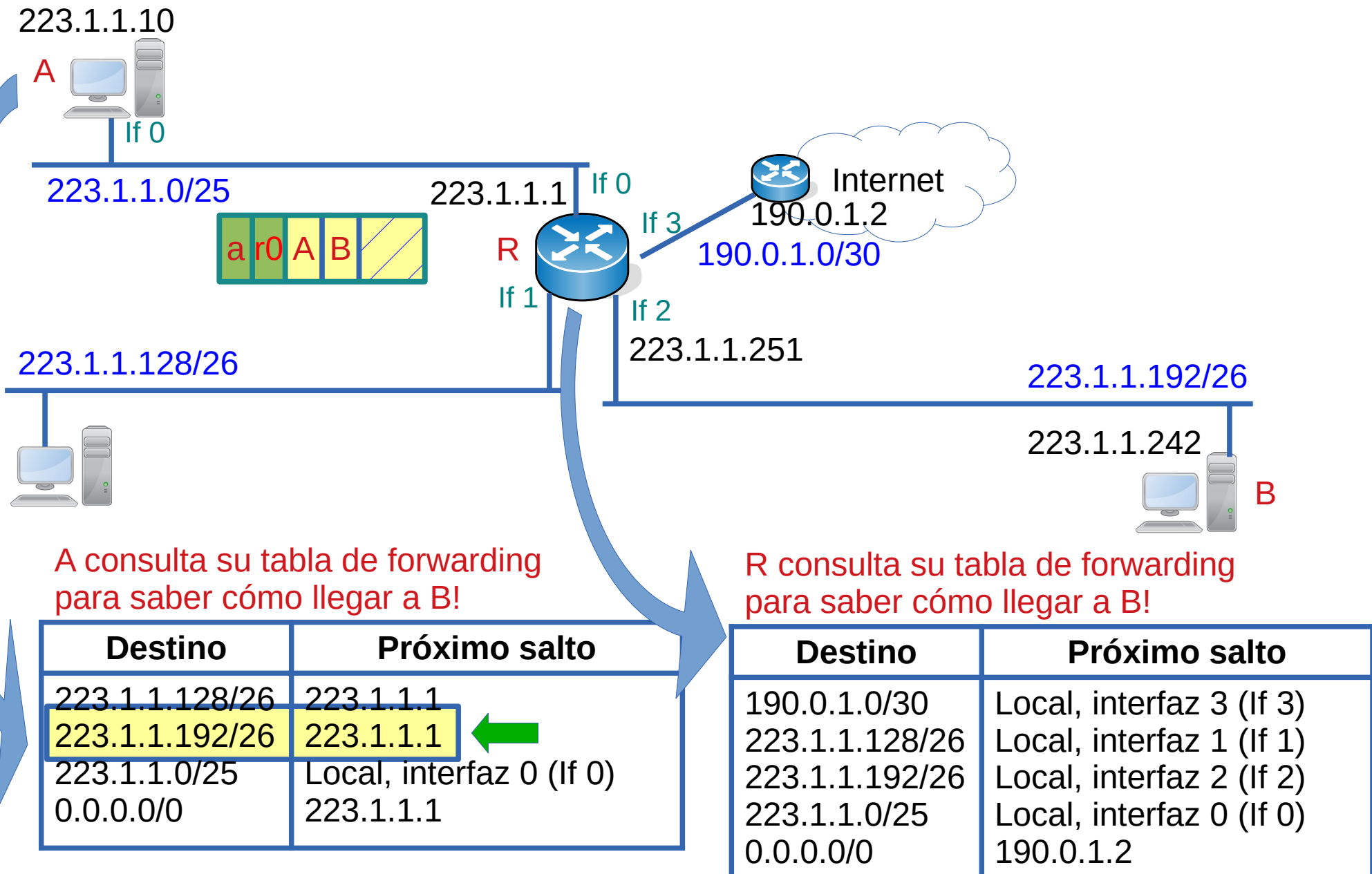
¿Qué pasa cuando A quiere enviar un paquete a B?



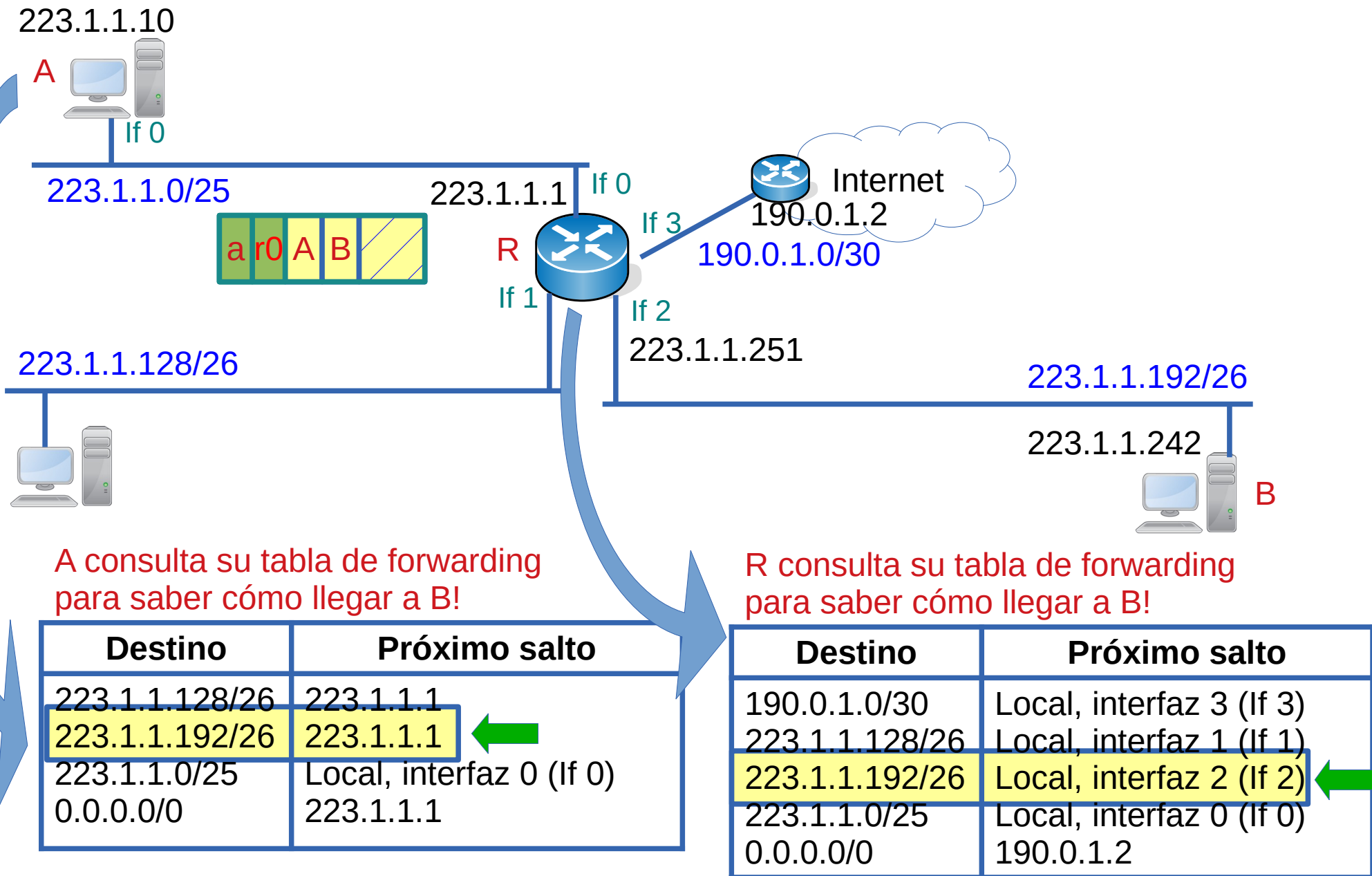
A consulta su tabla de forwarding para saber cómo llegar a B!

Destino	Próximo salto
223.1.1.128/26	223.1.1.1
223.1.1.192/26	223.1.1.1
223.1.1.0/25	Local, interfaz 0 (If 0)
0.0.0.0/0	223.1.1.1

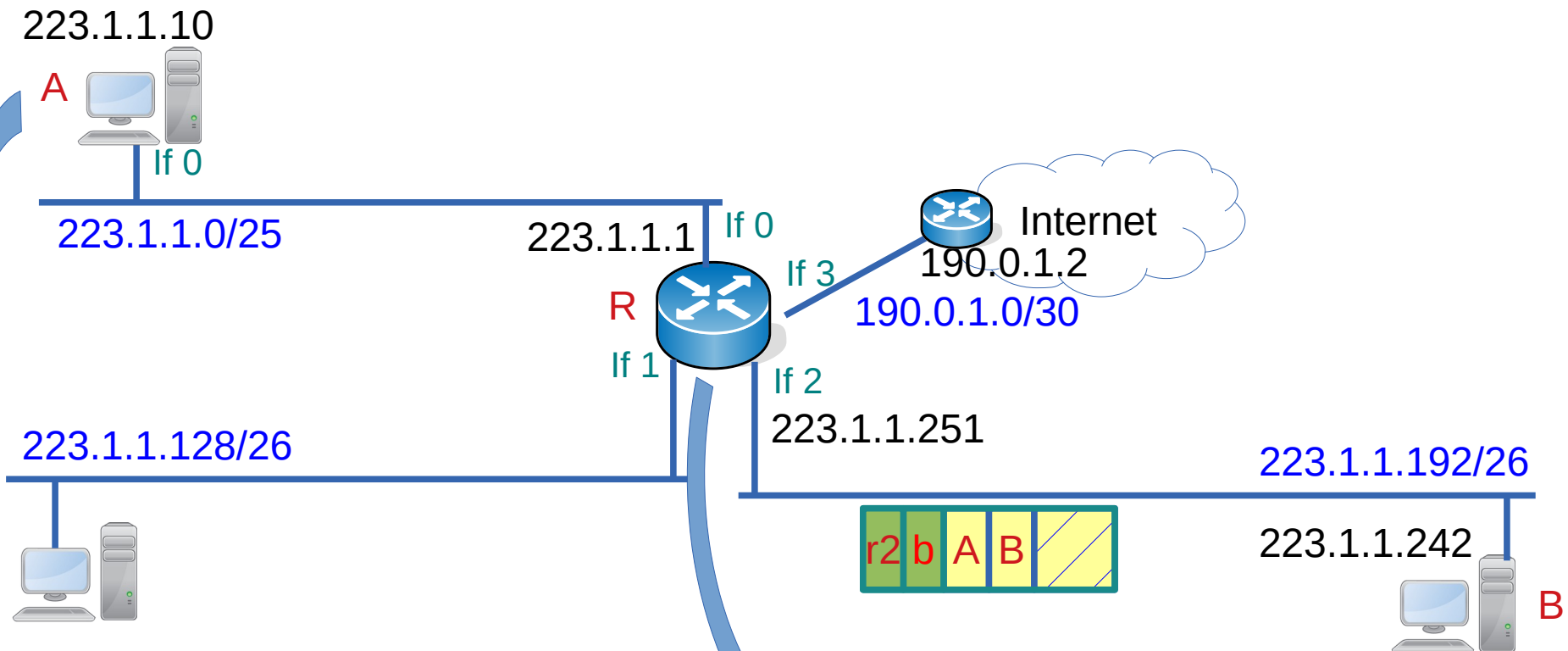
¿Qué pasa cuando A quiere enviar un paquete a B?



¿Qué pasa cuando A quiere enviar un paquete a B?



¿Qué pasa cuando A quiere enviar un paquete a B?



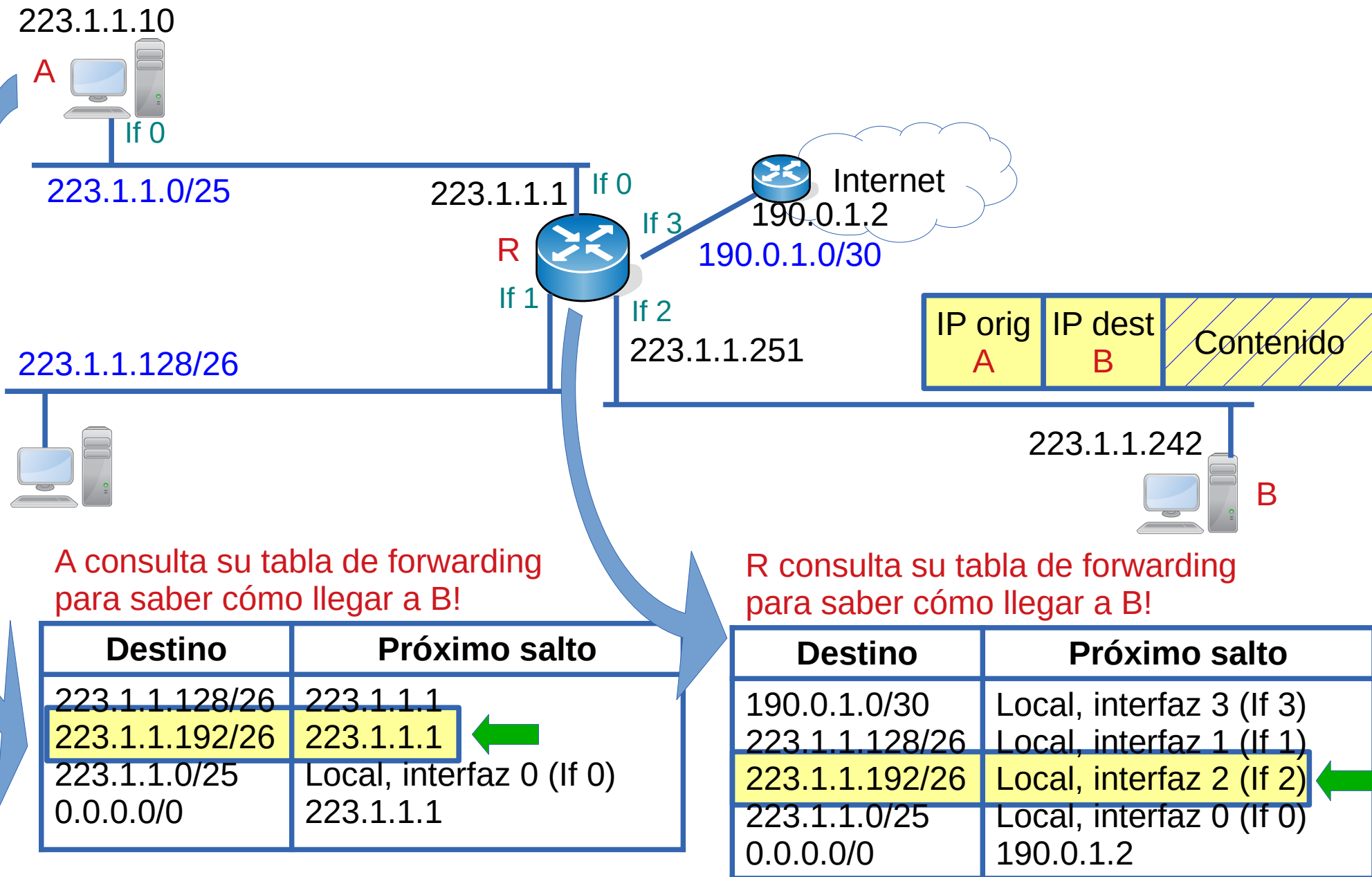
A consulta su tabla de forwarding para saber cómo llegar a B!

Destino	Próximo salto
223.1.1.128/26	223.1.1.1
223.1.1.192/26	223.1.1.1
223.1.1.0/25	Local, interfaz 0 (If 0)
0.0.0.0/0	223.1.1.1

R consulta su tabla de forwarding para saber cómo llegar a B!

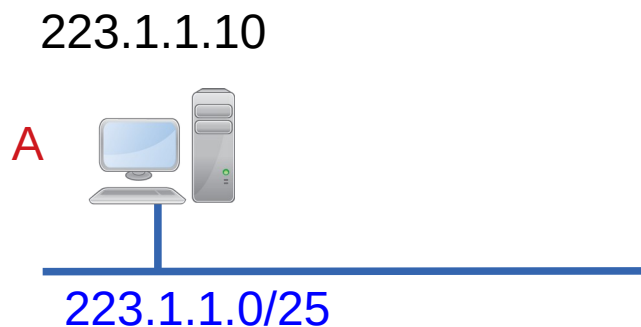
Destino	Próximo salto
190.0.1.0/30	Local, interfaz 3 (If 3)
223.1.1.128/26	Local, interfaz 1 (If 1)
223.1.1.192/26	Local, interfaz 2 (If 2)
223.1.1.0/25	Local, interfaz 0 (If 0)
0.0.0.0/0	190.0.1.2

¿Qué pasa cuando A quiere enviar un paquete a B?



Configuración de IP y máscara en una interfaz

- Cuando se configura una interfaz de un enrutador o equipo, se debe indicar la dirección IP de la interfaz y la máscara. De esta forma el equipo sabe cuál es el rango de direcciones que pueden accederse directamente usando los servicios de capa 2, sin pasar por un enrutador intermedio
- Esto se refleja en que se agrega automáticamente una entrada en la tabla de forwarding para llegar a esa red local
- Ej. Al configurar la interfaz de A con IP 223.1.1.10 y máscara /25, se agrega automáticamente la entrada a la red local.

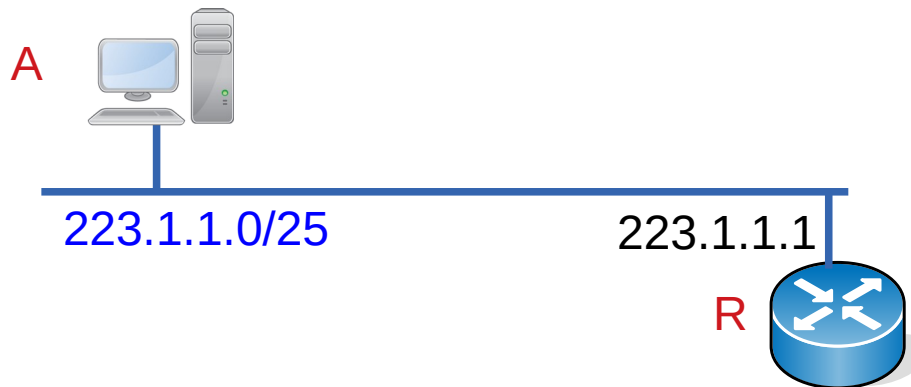


Destino	Próximo salto
223.1.1.0/25	Local, interfaz 0 (If 0)

¿Cómo A llega a equipos fuera de su red local?

- Necesito agregar entradas en la tabla de forwarding
- El próximo salto de esas entradas debe ser un destino **alcanzable** localmente. A llega a la IP 223.1.1.1 porque está conectada a su red local
- Ej. Agregamos entradas en A para llegar a las subredes 2, 3 e Internet

223.1.1.10



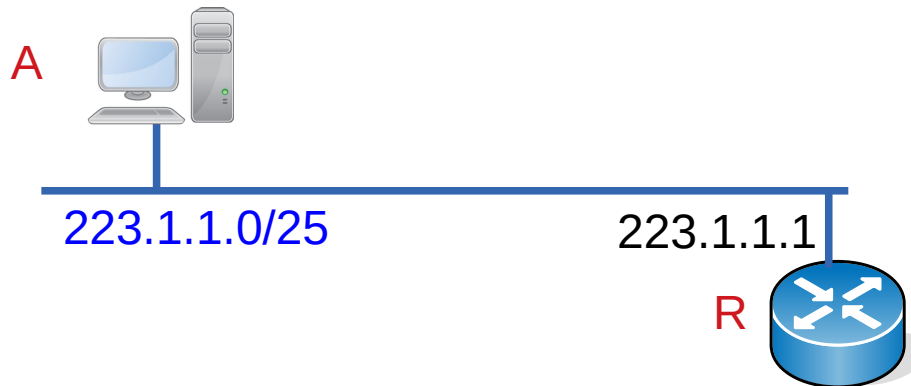
Destino	Próximo salto
223.1.1.128/26	223.1.1.1
223.1.1.192/26	223.1.1.1
223.1.1.0/25	Local, interfaz 0 (If 0)
0.0.0.0/0	223.1.1.1

- Se pueden sumarizar entradas que tengan el mismo “próximo salto” de modo de minimizar la tabla.

¿Cómo A llega a equipos fuera de su red local?

- Necesito agregar entradas en la tabla de forwarding
- El próximo salto de esas entradas debe ser un destino **alcanzable** localmente. A llega a la IP 223.1.1.1 porque está conectada a su red local
- Ej. Agregamos entradas en A para llegar a las subredes 2, 3 e Internet

223.1.1.10



Destino	Próximo salto
223.1.1.128/26	223.1.1.1
223.1.1.192/26	223.1.1.1
223.1.1.0/25	Local, interfaz 0 (If 0)
0.0.0.0/0	223.1.1.1

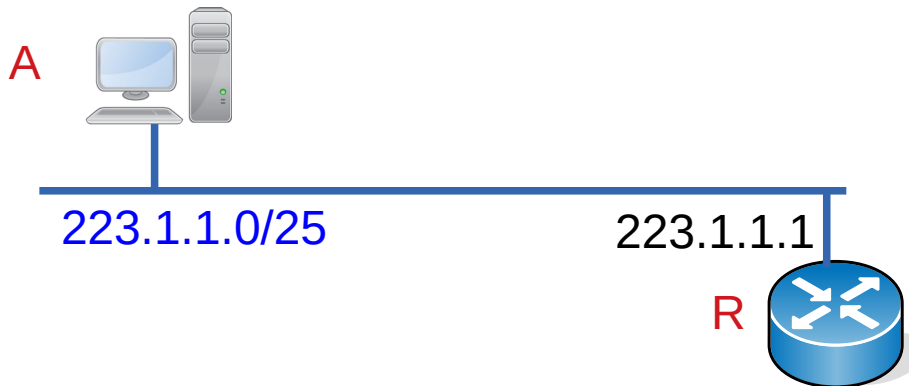
- Se pueden sumarizar entradas que tengan el mismo “próximo salto” de modo de minimizar la tabla.

Destino	Próximo salto
223.1.1.128/25	223.1.1.1
223.1.1.0/25	Local, interfaz 0 (If 0)
0.0.0.0/0	223.1.1.1

¿Cómo A llega a equipos fuera de su red local?

- Necesito agregar entradas en la tabla de forwarding
- El próximo salto de esas entradas debe ser un destino **alcanzable** localmente. A llega a la IP 223.1.1.1 porque está conectada a su red local
- Ej. Agregamos entradas en A para llegar a las subredes 2, 3 e Internet

223.1.1.10



Destino	Próximo salto
223.1.1.128/26	223.1.1.1
223.1.1.192/26	223.1.1.1
223.1.1.0/25	Local, interfaz 0 (If 0)
0.0.0.0/0	223.1.1.1

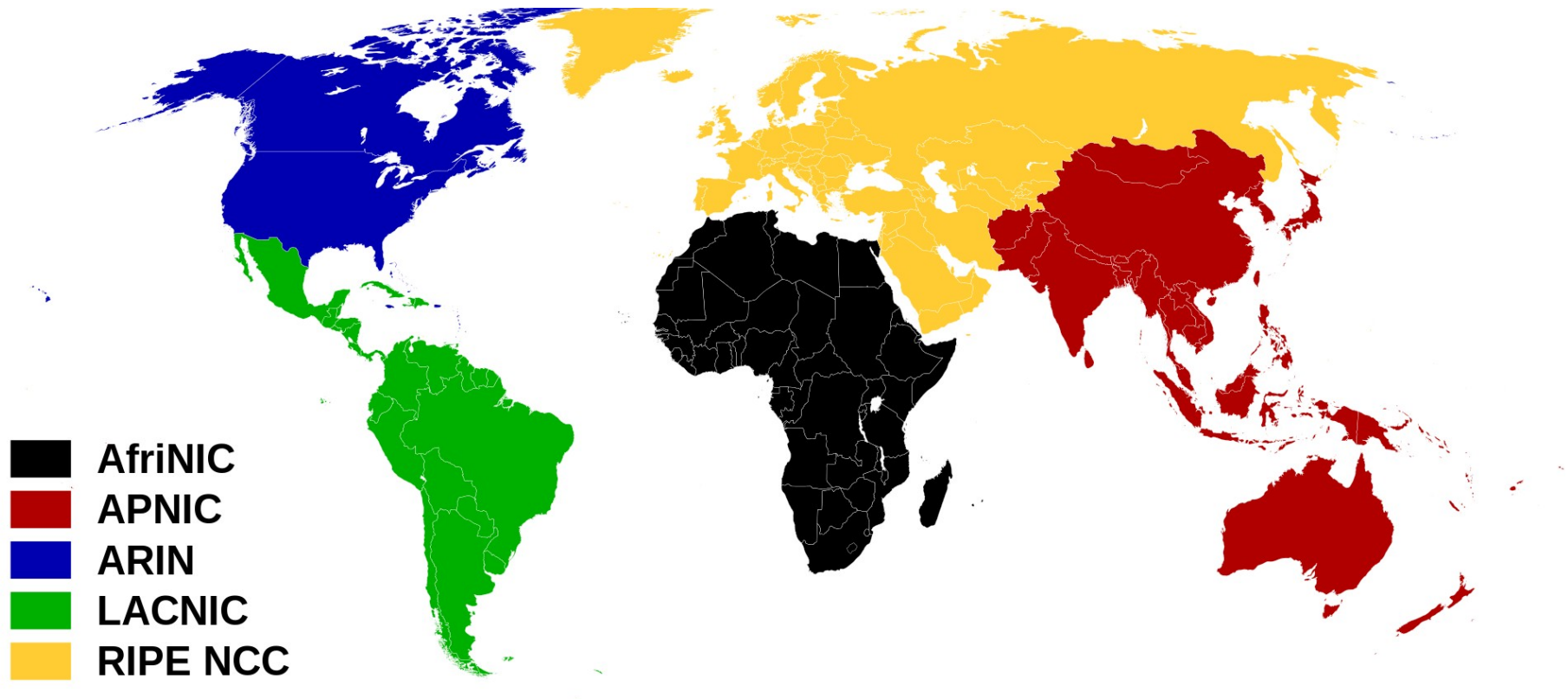
- Se pueden sumarizar entradas que tengan el mismo “próximo salto” de modo de minimizar la tabla.

Destino	Próximo salto
223.1.1.128/25	223.1.1.1
223.1.1.0/25	Local, interfaz 0 (If 0)
0.0.0.0/0	223.1.1.1

Destino	Próximo salto
223.1.1.0/25	Local, interfaz 0 (If 0)
0.0.0.0/0	223.1.1.1

Asignación de rangos de direcciones

- Las direcciones en Internet son administradas por el **ICANN** (Internet Corporation for Assigned Names and Numbers)
 - Números IP, números de puertos, nombres de dominio del DNS
- El ICANN asigna rangos de direcciones IP a los registros regionales
 - **RIRs** (Regional Internet Registries)



Asignación de rangos de direcciones

- Los **RIRs** asignan rangos de direcciones a los Proveedores de Servicio (Internet Service Provider, **ISP**) o también directamente a organizaciones de gran porte
- Los Proveedores de Internet (**ISP**) asignan rangos de direcciones a sus **clientes** (empresas, universidades, usuarios finales)
- Para **usuarios finales**, las IP y otros datos se pueden asignar:
 - En la negociación con PPPoE (Point to Point Protocol over Ethernet)
 - Con DHCP (Dynamic Host Configuration Protocol)
 - En el caso de los servicios móviles (celulares), con protocolos específicos de la gestión de acceso
- **Organizaciones** que requieren un rango de direcciones
 - Se asigna administrativamente un rango que el administrador de la red asigna a su vez a sus equipos internos

Asignación de direcciones a dispositivos de la red

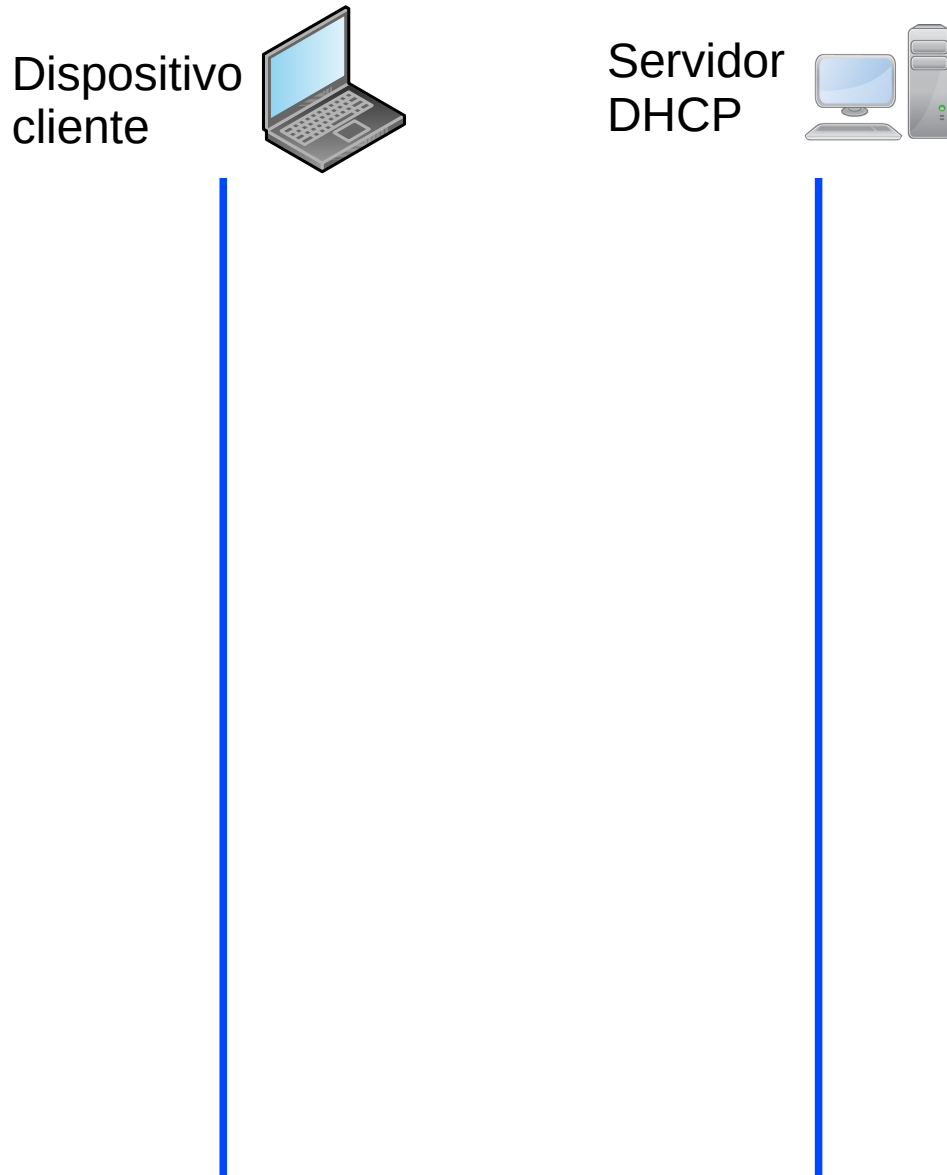
- Las direcciones IP a cada dispositivo de una red local pueden asignarse de forma **manual** o de forma **dinámica**
- La asignación **manual** consiste en que un administrador asigne una dirección a cada dispositivo y la configure en el equipo
 - Será necesario llevar un registro de las direcciones en uso dentro del rango disponible y gestionar las altas, bajas y modificaciones
- La asignación **dinámica** consiste en que los dispositivos al conectarse a la red obtienen los datos de configuración de un servidor
 - Debe existir un servidor que gestione las direcciones disponibles
 - Se requiere que los dispositivos se configuren para obtener una dirección dinámicamente
 - Existe un protocolo llamado DHCP (Dynamic Host Configuration Protocol) estandarizado en la RFC 2131 que resuelve esta configuración dinámica en una red local (LAN)
- Actualmente, prácticamente todos los dispositivos están por defecto configurados para obtener la configuración por DHCP

Ventajas de la asignación dinámica

- La configuración se realiza de forma **centralizada** en un servidor
 - No necesito recorrer todos los dispositivos y configurarlos, la configuración se realiza enteramente en el servidor DHCP
 - El servidor DHCP permite configurar:
 - **Dirección IP** asignada al dispositivo
 - La **máscara** de la red local (define el rango de máquinas directamente conectadas)
 - Las direcciones IP de los **servidores de DNS**
 - La dirección IP del próximo salto asociado a la **ruta por defecto** (default gateway) (agrega una entrada en la tabla de forwarding)
 - Permite también algunas otras configuraciones
- Permite usar más **eficientemente** el rango de direcciones asignado cuando tengo equipos portátiles que se conectan y desconectan (por ej. laptops)
 - Dispongo de un rango de direcciones y las asigno a los equipos conectados
 - No necesito asignar una dirección a equipos no conectados
 - Cuando un equipo se desconecta de la red, libera la dirección asignada y queda disponible para asignársela a otro equipo más tarde

DHCP (Dynamic Host Configuration Protocol)

- Es un protocolo tipo cliente-servidor



DHCP (Dynamic Host Configuration Protocol)

- Es un protocolo tipo cliente-servidor



Servidor DHCP



Src: 0.0.0.0 Dst: 255.255.255.255

UDP Src port: 68 Dst port: 67

Descubrimiento de servidor DHCP
Mensaje "DHCP discover"

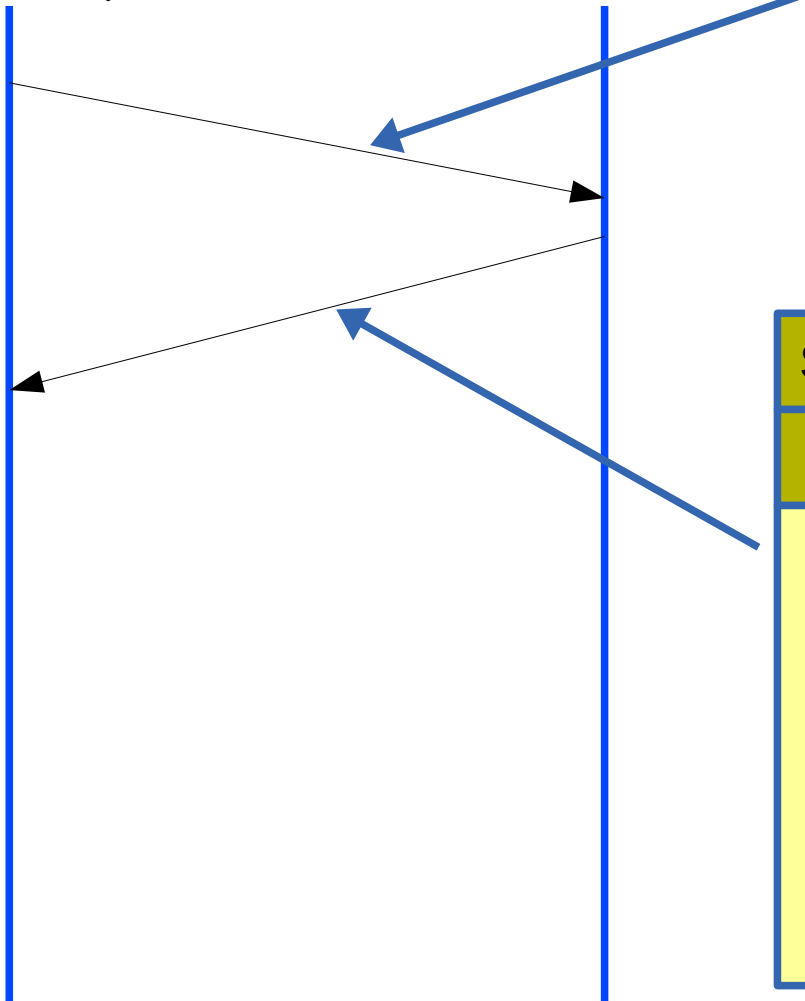
xid: X (identificador de transacción)
yaddress: 0.0.0.0

DHCP (Dynamic Host Configuration Protocol)

- Es un protocolo tipo cliente-servidor



Servidor DHCP



Src: 0.0.0.0 Dst: 255.255.255.255

UDP Src port: 68 Dst port: 67

Descubrimiento de servidor DHCP
Mensaje "DHCP discover"

xid: X (identificador de transacción)
yaddress: 0.0.0.0

Src: servidor DHCP Dst: 255.255.255.255

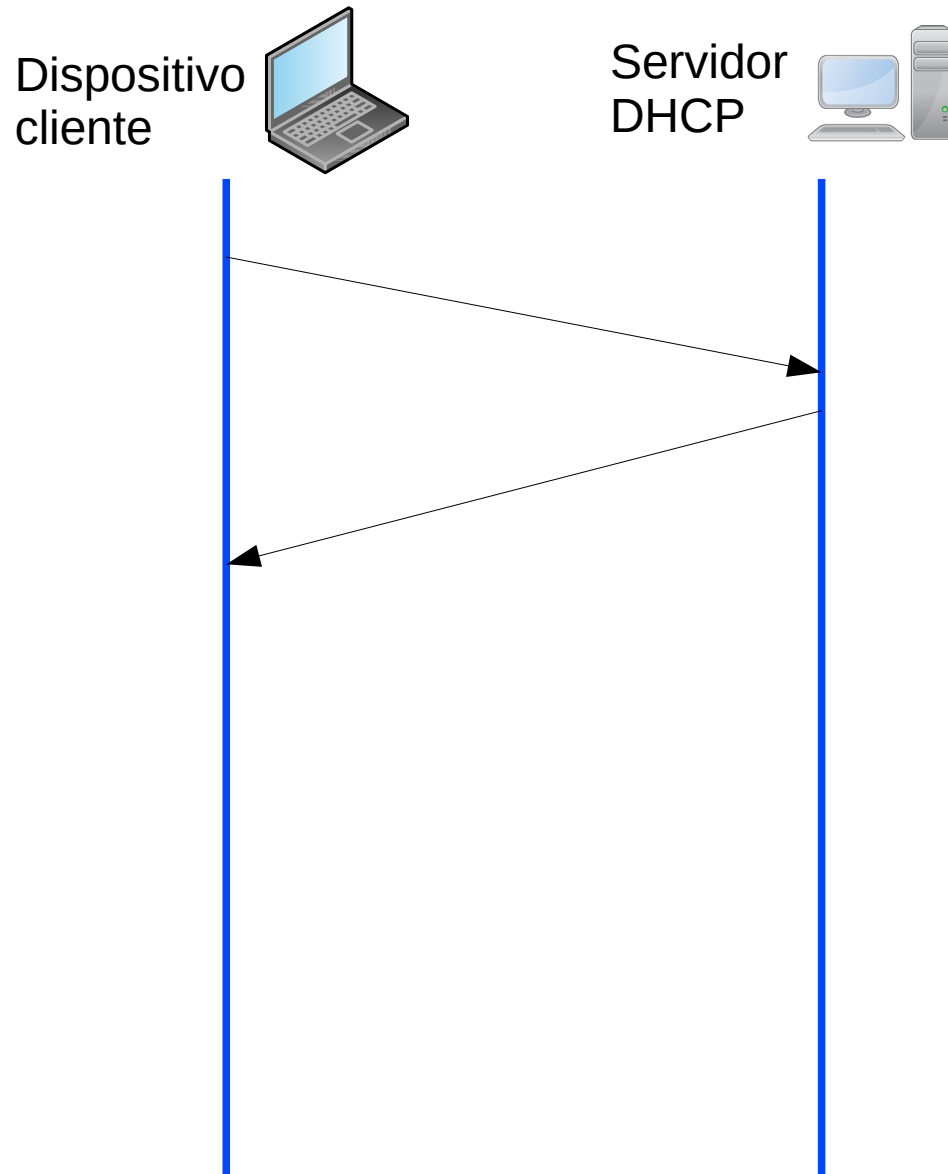
UDP Src port: 67 Dst port: 68

Oferta de servidor(es) DHCP
Mensaje "DHCP offer"

yaddress: IP asignada
+Otros parámetros
xid: X
DHCP server: IP del servidor DHCP
Lease time: 3600 seg

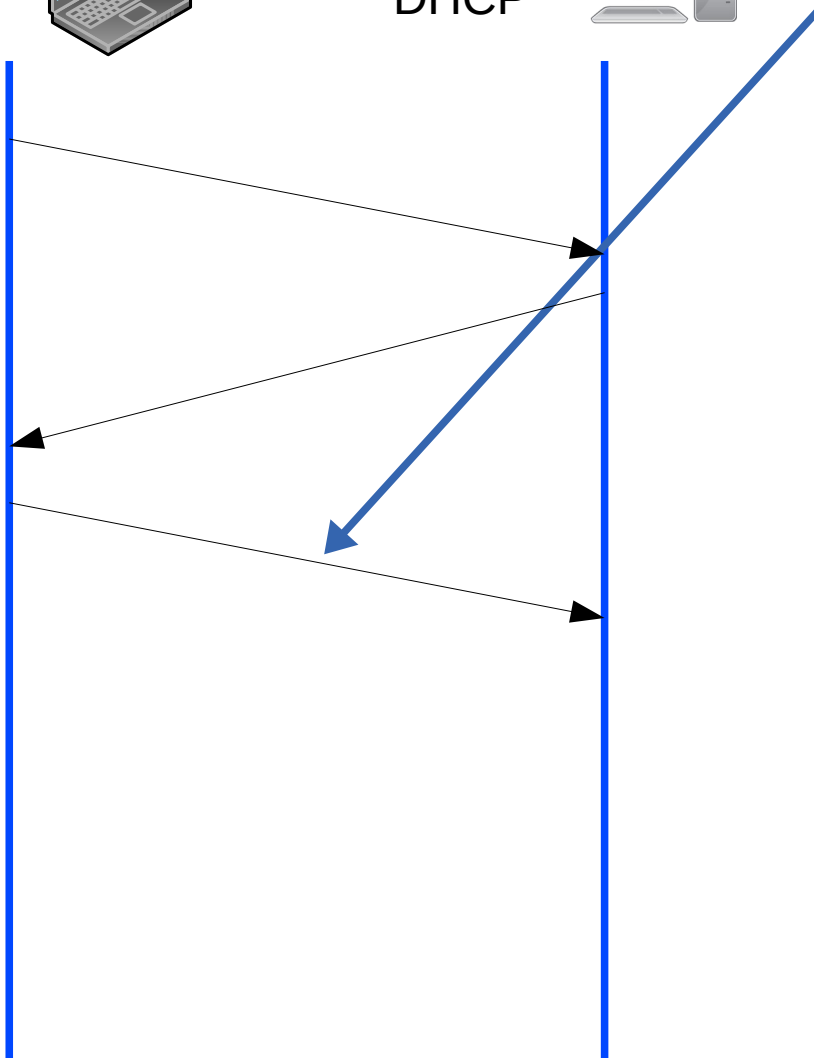
DHCP (Dynamic Host Configuration Protocol)

- Es un protocolo tipo cliente-servidor



DHCP (Dynamic Host Configuration Protocol)

- Es un protocolo tipo cliente-servidor



Src: 0.0.0.0 Dst: 255.255.255.255

UDP Src port: 68 Dst port: 67

Pedido de DHCP

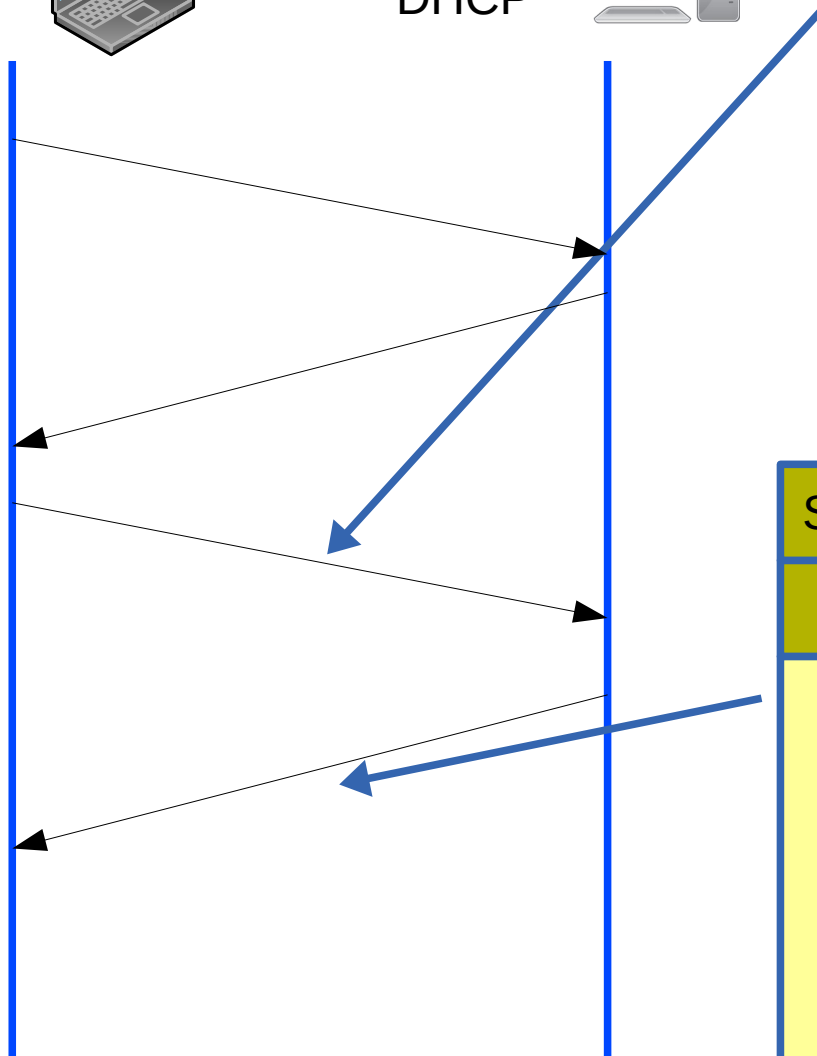
Mensaje "DHCP request"

yaddress: IP asignada
+Otros parámetros
xid: X

DHCP server: IP del servidor DHCP
Lease time: 3600 seg

DHCP (Dynamic Host Configuration Protocol)

- Es un protocolo tipo cliente-servidor



Src: 0.0.0.0 Dst: 255.255.255.255

UDP Src port: 68 Dst port: 67

Pedido de DHCP

Mensaje "DHCP request"

yaddress: IP asignada
+Otros parámetros
xid: X

DHCP server: IP del servidor DHCP
Lease time: 3600 seg

Src: servidor DHCP Dst: 255.255.255.255

UDP Src port: 67 Dst port: 68

Reconocimiento DHCP

Mensaje "DHCP ACK"

yaddress: IP asignada
+Otros parámetros
xid: X

DHCP server: IP del servidor DHCP
Lease time: 3600 seg

DHCP (Dynamic Host Configuration Protocol)

- Existen más mensajes, en particular el “**renew**” para renovar la IP otorgada si la quiero usar más tiempo que lo que indica el “**lease time**”
- El mensaje “**discover**” de un cliente, podría recibir respuestas de varios servidores DHCP por lo que el cliente decide con cuál sigue la negociación
 - Esto es un potencial problema en la práctica ya que cualquier usuario con capacidad de administración de un equipo (por ejemplo su laptop!) puede levantar por error o malicia un servidor DHCP
- Como está pensado para usar en una red local, al menos el mensaje de descubrimiento se envía usando direcciones broadcast de capa de enlace (para que lleguen a todos los equipos de la red)
 - Las respuestas podrían o no ser en modo broadcast de capa 2
- Cuando el servidor DHCP no está en la misma red, se usa un “DHCP relay”
- Hay varias RFC que describen opciones posibles
 - RFC 2132, RFC 3442, RFC 3942, RFC 4361, RFC 4833, RFC 5494
- Es posible **asignar a un equipo siempre la misma IP** (según la dirección MAC) lo que puede ser útil para servidores (aunque normalmente se les asigna una IP manualmente)

DHCP Discover

0.0.0.0:68 > 255.255.255.255:67: xid 0x4d369682

Client-Ethernet-Address f4:8e:38:f2:f3:58

Vendor Extensions

DHCP-Message Option 53, length 1: Discover

Client-ID Option 61, length 7: ether f4:8e:38:f2:f3:58

Requested-IP Option 50, length 4: 164.73.38.222

Hostname Option 12, length 15: "DESKTOP-FLS593Q"

Parameter-Request Option 55, length 14:

Subnet-Mask, Default-Gateway, Domain-Name-Server, Domain-Name

Router-Discovery, Static-Route, Vendor-Option, Netbios-Name-Server

Netbios-Node, Netbios-Scope, Option 119, Classless-Static-Route

Classless-Static-Route-Microsoft, Option 252

DHCP Offer

164.73.38.1:67 > 255.255.255.255:68: xid 0x4d369682

Your-IP 164.73.38.222

Server-IP 164.73.32.216

Gateway-IP 164.73.38.1

Client-Ethernet-Address f4:8e:38:f2:f3:58

Vendor Extensions

DHCP-Message Option 53, length 1: Offer

Server-ID Option 54, length 4: 164.73.32.131

Lease-Time Option 51, length 4: 14400

Subnet-Mask Option 1, length 4: 255.255.255.0

Default-Gateway Option 3, length 4: 164.73.38.1

Domain-Name-Server Option 6, length 8: 164.73.32.2,164.73.32.4

Domain-Name Option 15, length 11: "fing.edu.uy"

DHCP Request

0.0.0.0:68 > 255.255.255.255:67: xid 0x4d369682

Client-Ethernet-Address f4:8e:38:f2:f3:58

Vendor Extensions

DHCP-Message Option 53, length 1: Request

Client-ID Option 61, length 7: ether f4:8e:38:f2:f3:58

Requested-IP Option 50, length 4: 164.73.38.222

Server-ID Option 54, length 4: 164.73.32.131

Hostname Option 12, length 15: "DESKTOP-FLS593Q"

Parameter-Request Option 55, length 14:

Subnet-Mask, Default-Gateway, Domain-Name-Server, Domain-Name

Router-Discovery, Static-Route, Vendor-Option, Netbios-Name-Server

Netbios-Node, Netbios-Scope, Option 119, Classless-Static-Route

Classless-Static-Route-Microsoft, Option 252

DHCP ACK

164.73.38.1:67 > 255.255.255.255:68: xid 0x4d369682

Your-IP 164.73.38.222

Server-IP 164.73.32.216

Gateway-IP 164.73.38.1

Client-Ethernet-Address f4:8e:38:f2:f3:58

Vendor Extensions

DHCP-Message Option 53, length 1: ACK

Server-ID Option 54, length 4: 164.73.32.131

Lease-Time Option 51, length 4: 14400

Subnet-Mask Option 1, length 4: 255.255.255.0

Default-Gateway Option 3, length 4: 164.73.38.1

Domain-Name-Server Option 6, length 8: 164.73.32.2,164.73.32.4

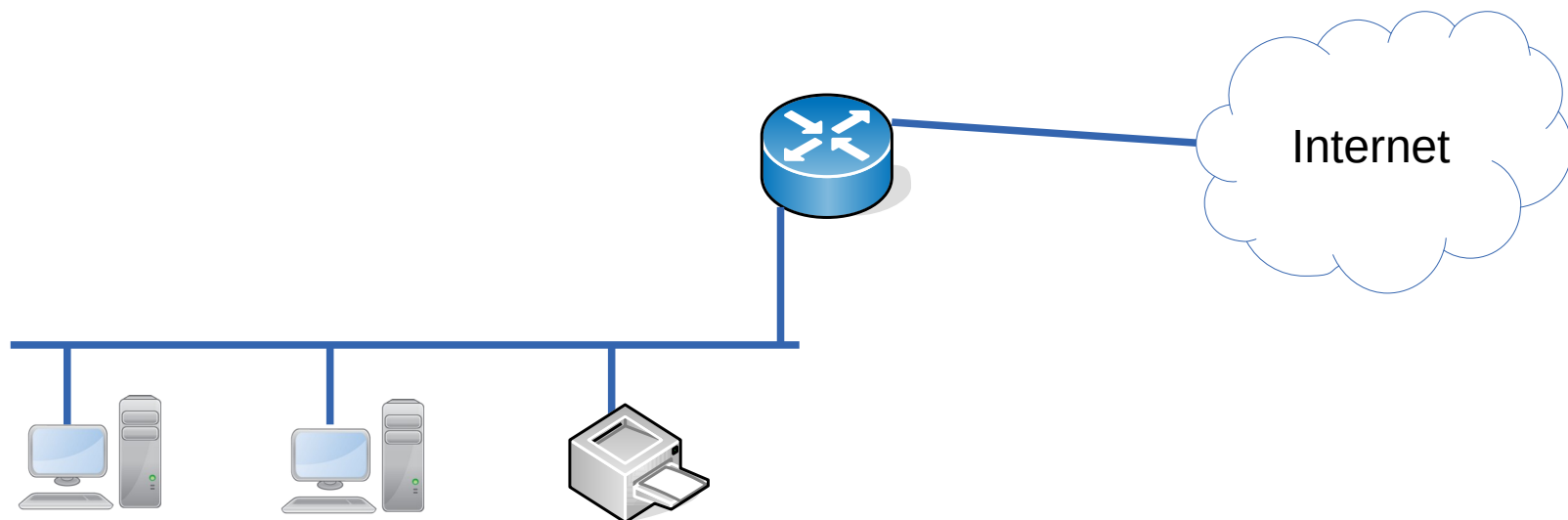
Domain-Name Option 15, length 11: "fing.edu.uy"

Escasez de direcciones IPv4

- Con la proliferación de dispositivos que requieren direcciones IP aparecen problemas:
 - El **rango de direcciones disponibles en IPv4 está limitado**
 - Inicialmente (con las clases A, B y C de direcciones) las direcciones **no se asignaron de forma óptima**
 - Aún haciendo un uso más óptimo, los rangos se fragmentarían implicando un crecimiento en las tablas de forwarding de los enrutadores
 - Se complica la gestión de direcciones: Hay que asignarle un rango a cada pequeña empresa u hogar
- **Solución de fondo:** Implementar un nuevo protocolo con mayor espacio de direcciones
 - **IPv6** : 128 bits para las direcciones
- **Solución transitoria:**
 - Network Address Translation: **NAT**
 - RFC 2663, RFC 3022

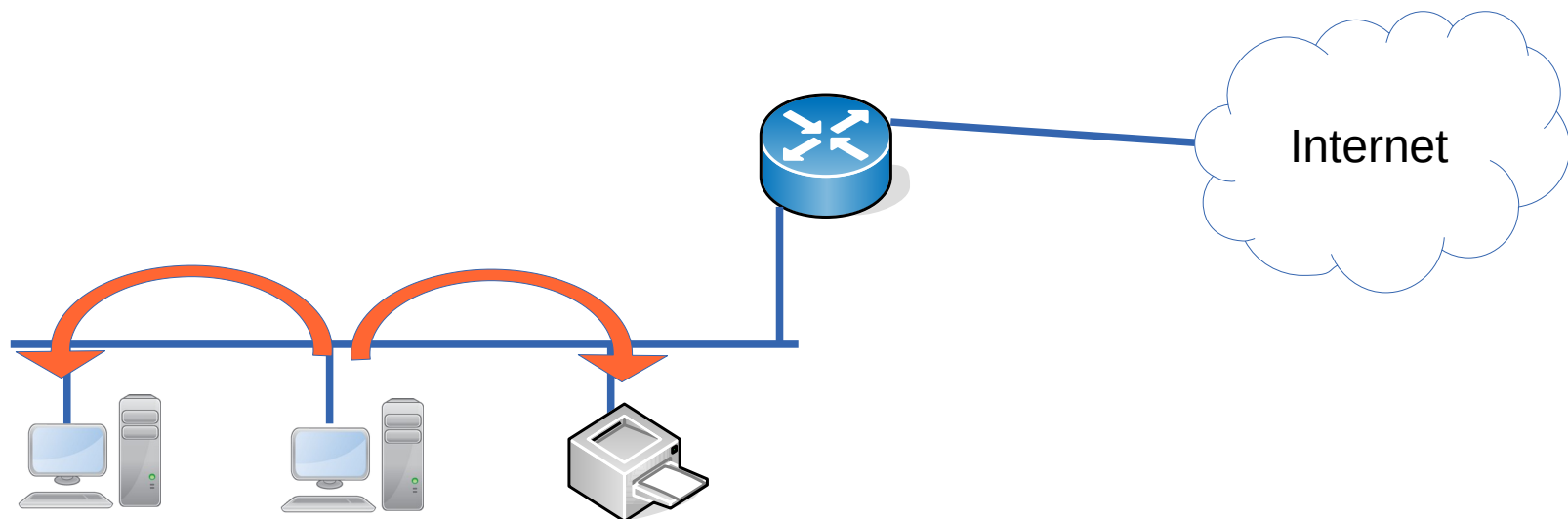
NAT: Network Address Translation

- ¿Necesitamos direcciones con validez global para comunicarnos entre los equipos dentro de mi red?
- Podríamos usar otro protocolo diferente de IP, pero eso sería poco práctico
- La idea es usar IP con rangos de “direcciones privadas” (RFC 1918), en contraposición con “direcciones públicas” o direcciones globales
- Esos rangos se pueden usar libremente dentro de mi red, pero no tienen validez en la red global
- ¿Cómo hago cuando necesito conectarme con un equipo de Internet?



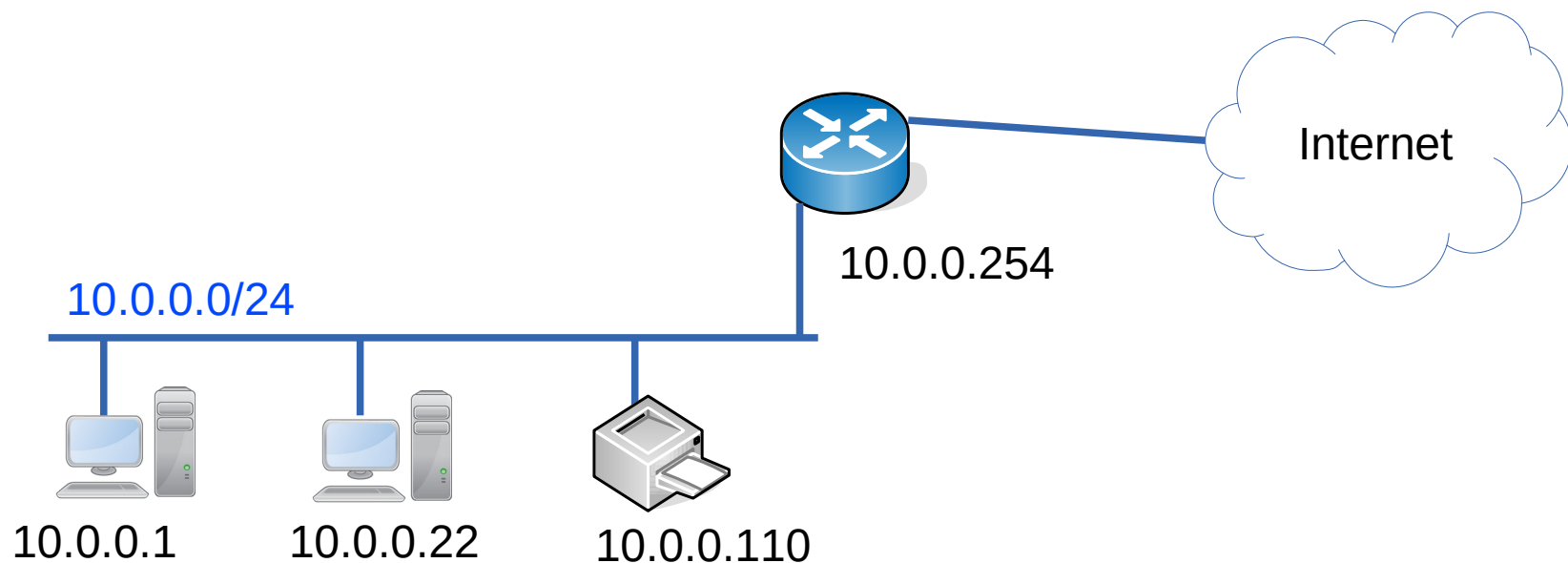
NAT: Network Address Translation

- ¿Necesitamos direcciones con validez global para comunicarnos entre los equipos dentro de mi red?
- Podríamos usar otro protocolo diferente de IP, pero eso sería poco práctico
- La idea es usar IP con rangos de “direcciones privadas” (RFC 1918), en contraposición con “direcciones públicas” o direcciones globales
- Esos rangos se pueden usar libremente dentro de mi red, pero no tienen validez en la red global
- ¿Cómo hago cuando necesito conectarme con un equipo de Internet?



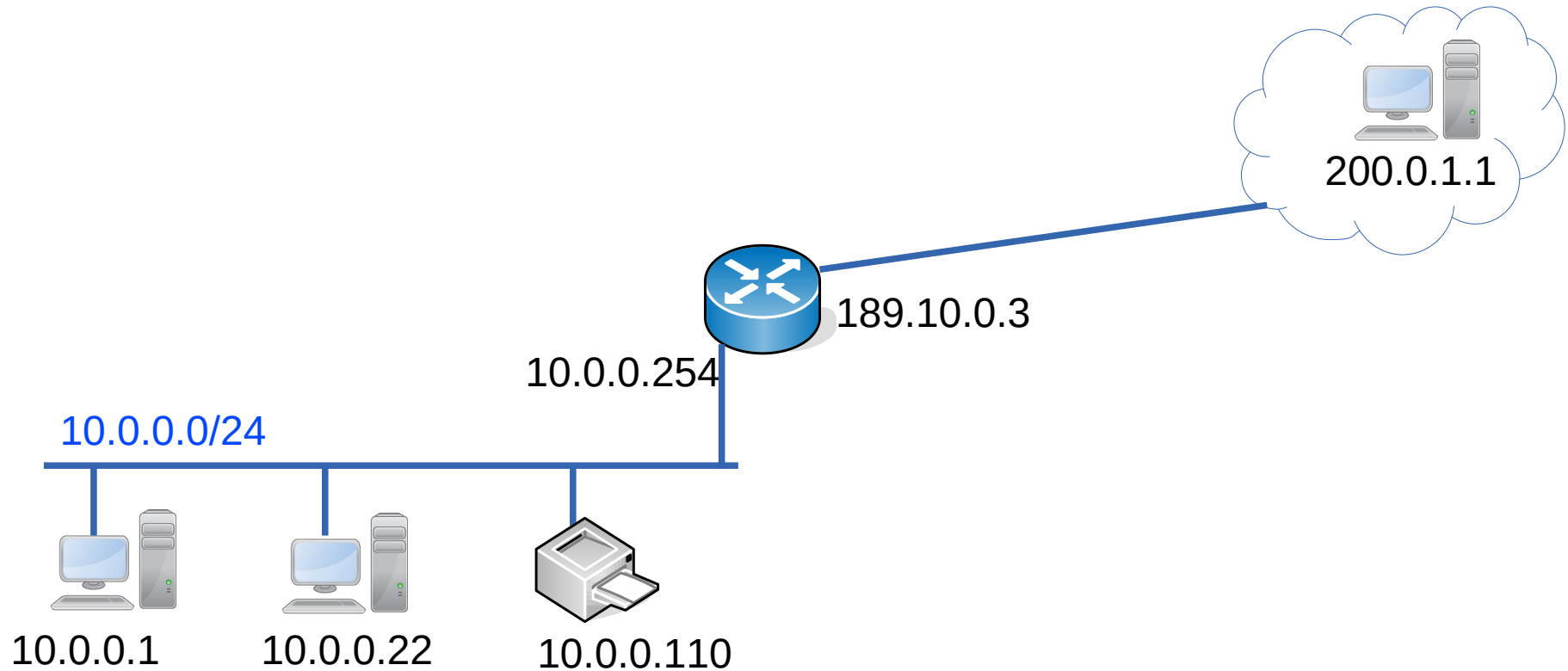
NAT: Network Address Translation

- ¿Necesitamos direcciones con validez global para comunicarnos entre los equipos dentro de mi red?
- Podríamos usar otro protocolo diferente de IP, pero eso sería poco práctico
- La idea es usar IP con rangos de “direcciones privadas” (RFC 1918), en contraposición con “direcciones públicas” o direcciones globales
- Esos rangos se pueden usar libremente dentro de mi red, pero no tienen validez en la red global
- ¿Cómo hago cuando necesito conectarme con un equipo de Internet?



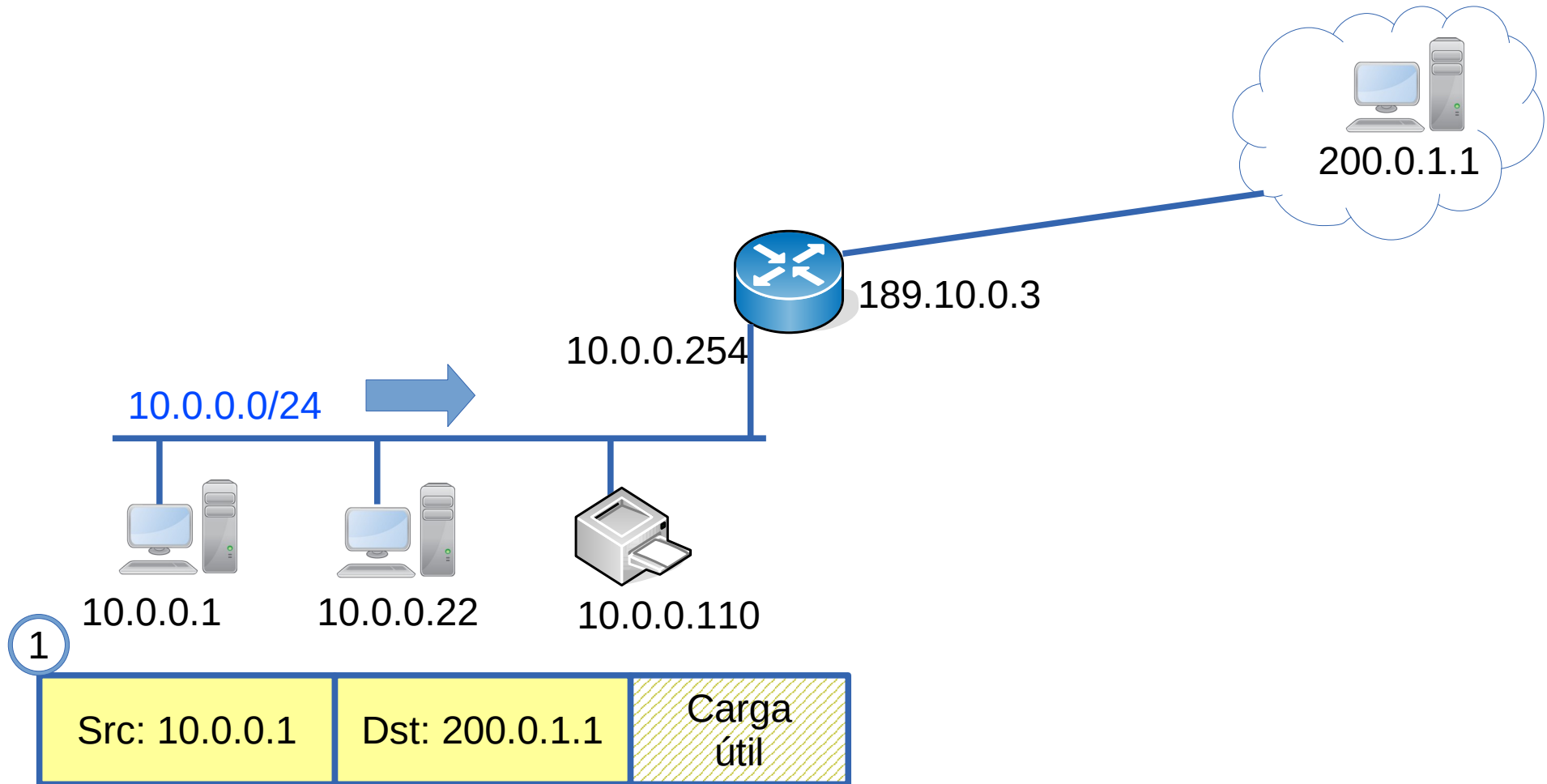
NAT: Network Address Translation

- La idea es cambiar o **traducir** las direcciones privadas en direcciones públicas, sólo cuando los paquetes tienen que salir a la red global



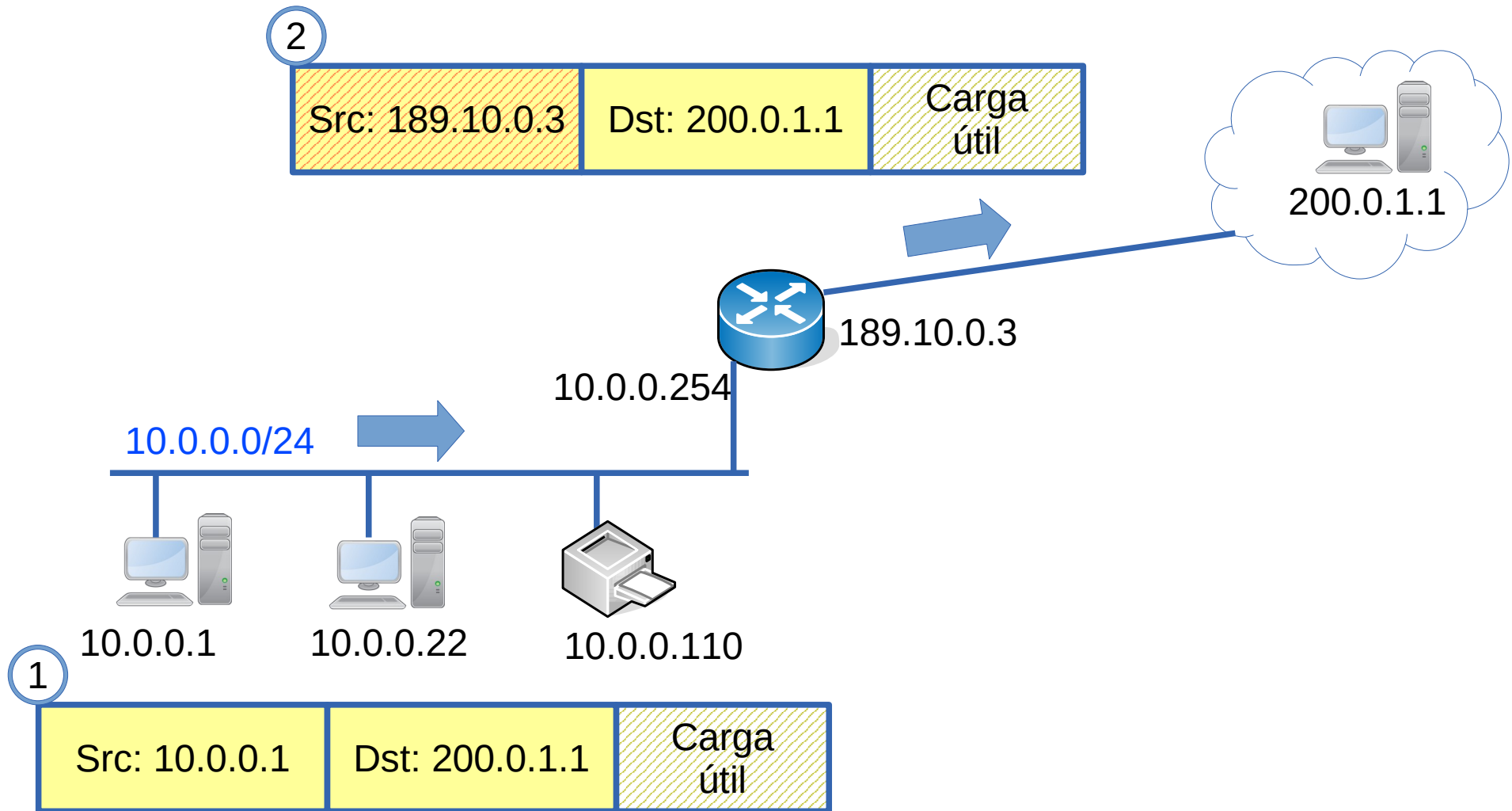
NAT: Network Address Translation

- La idea es cambiar o **traducir** las direcciones privadas en direcciones públicas, sólo cuando los paquetes tienen que salir a la red global



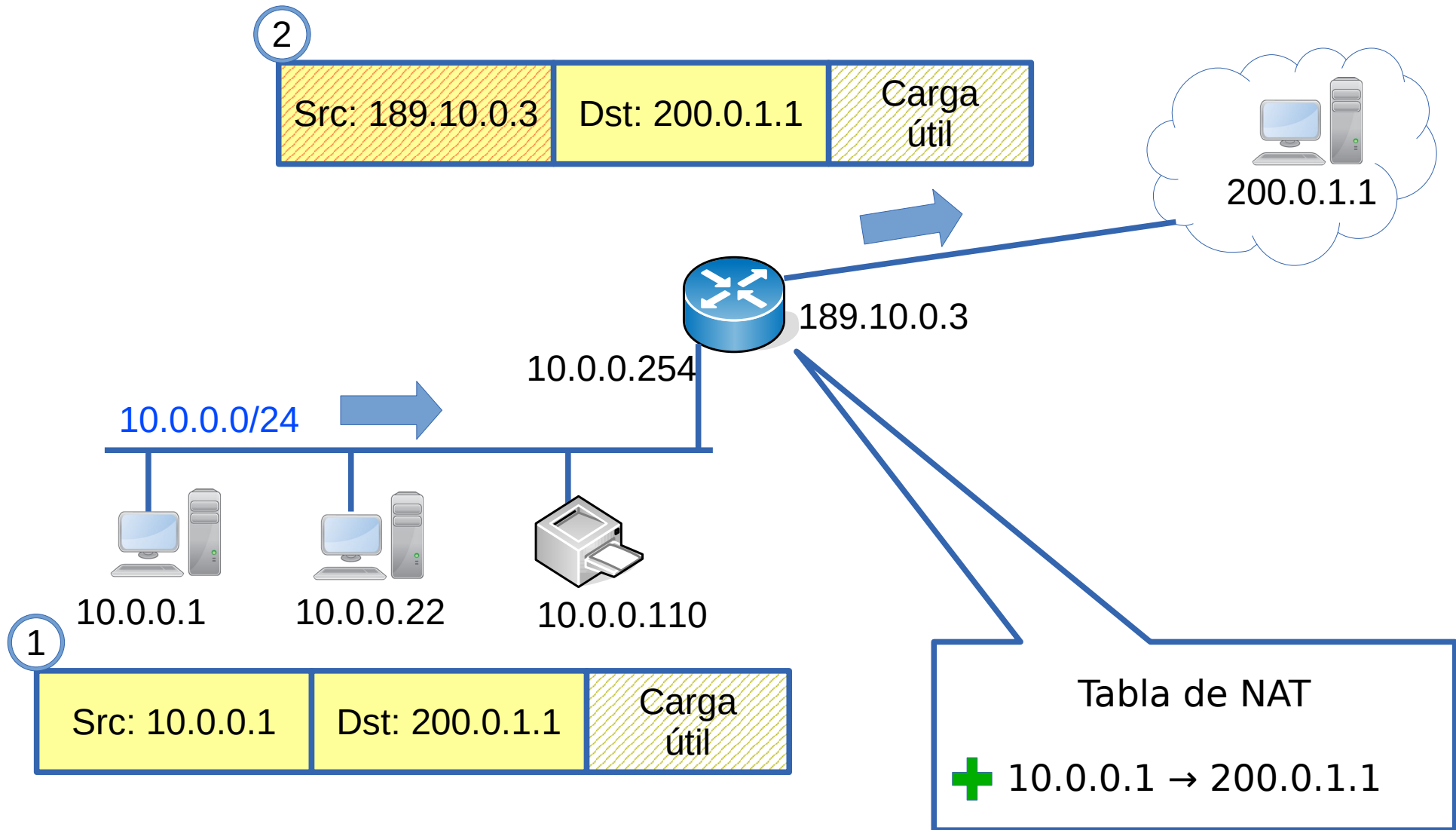
NAT: Network Address Translation

- La idea es cambiar o **traducir** las direcciones privadas en direcciones públicas, sólo cuando los paquetes tienen que salir a la red global



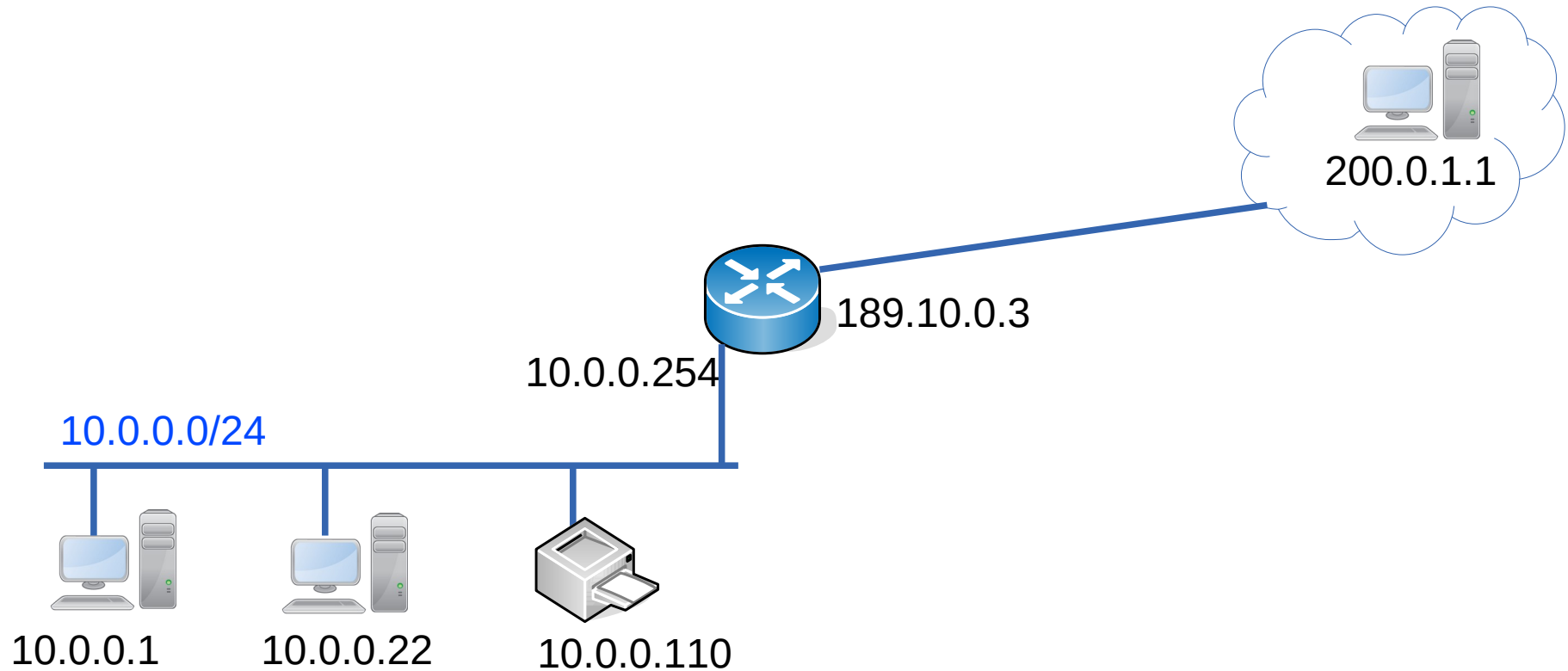
NAT: Network Address Translation

- La idea es cambiar o **traducir** las direcciones privadas en direcciones públicas, sólo cuando los paquetes tienen que salir a la red global



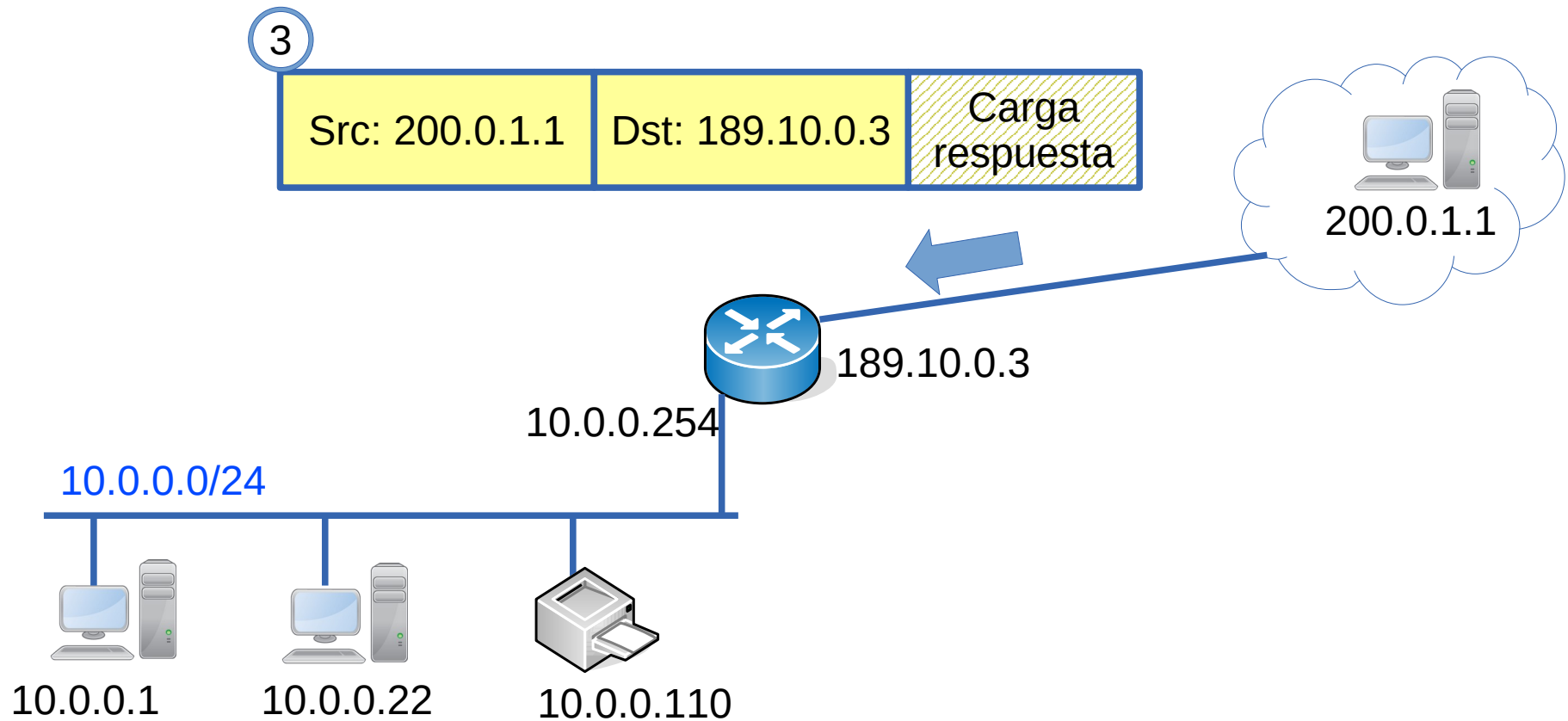
NAT: Network Address Translation

- Vuelta de los paquetes



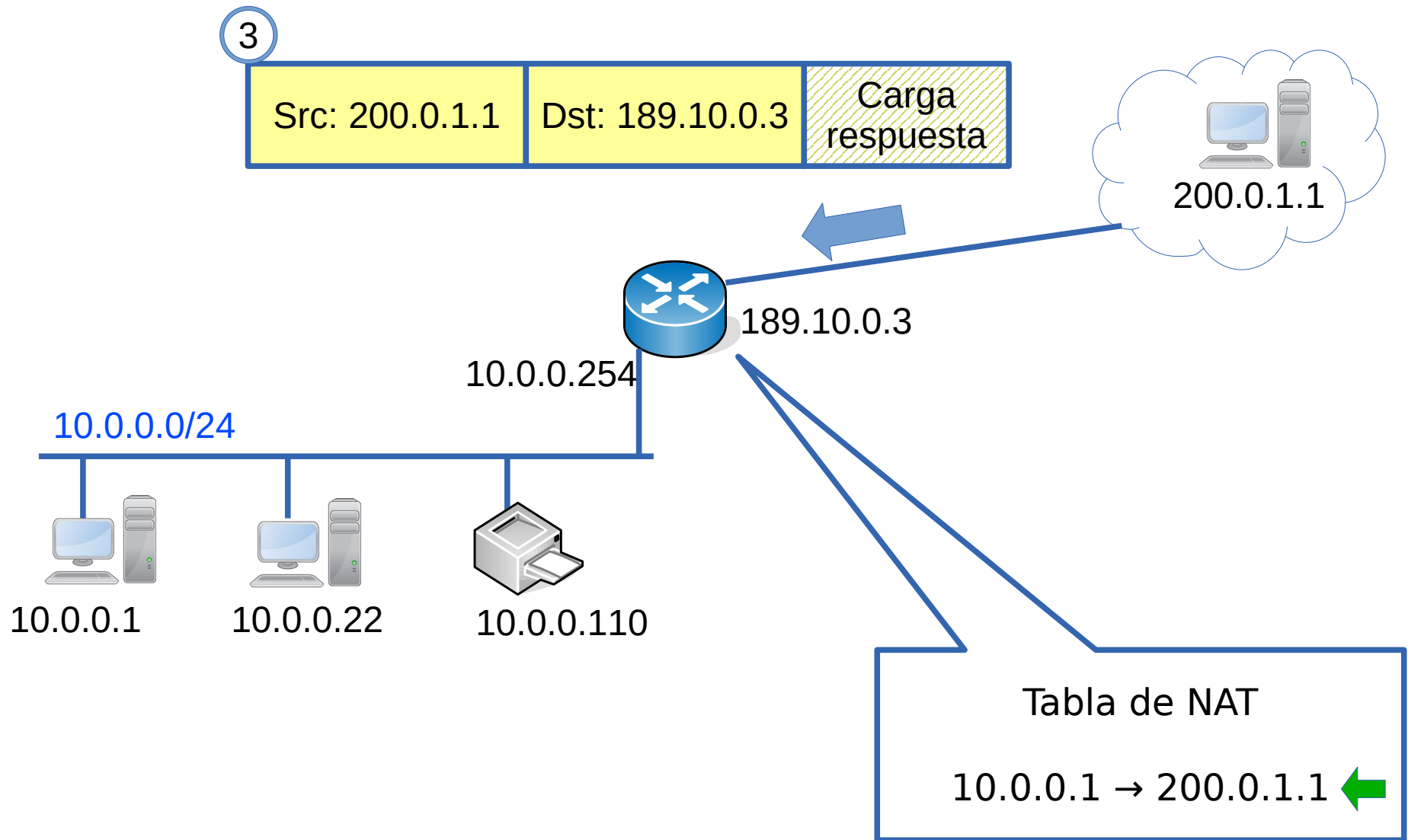
NAT: Network Address Translation

- Vuelta de los paquetes



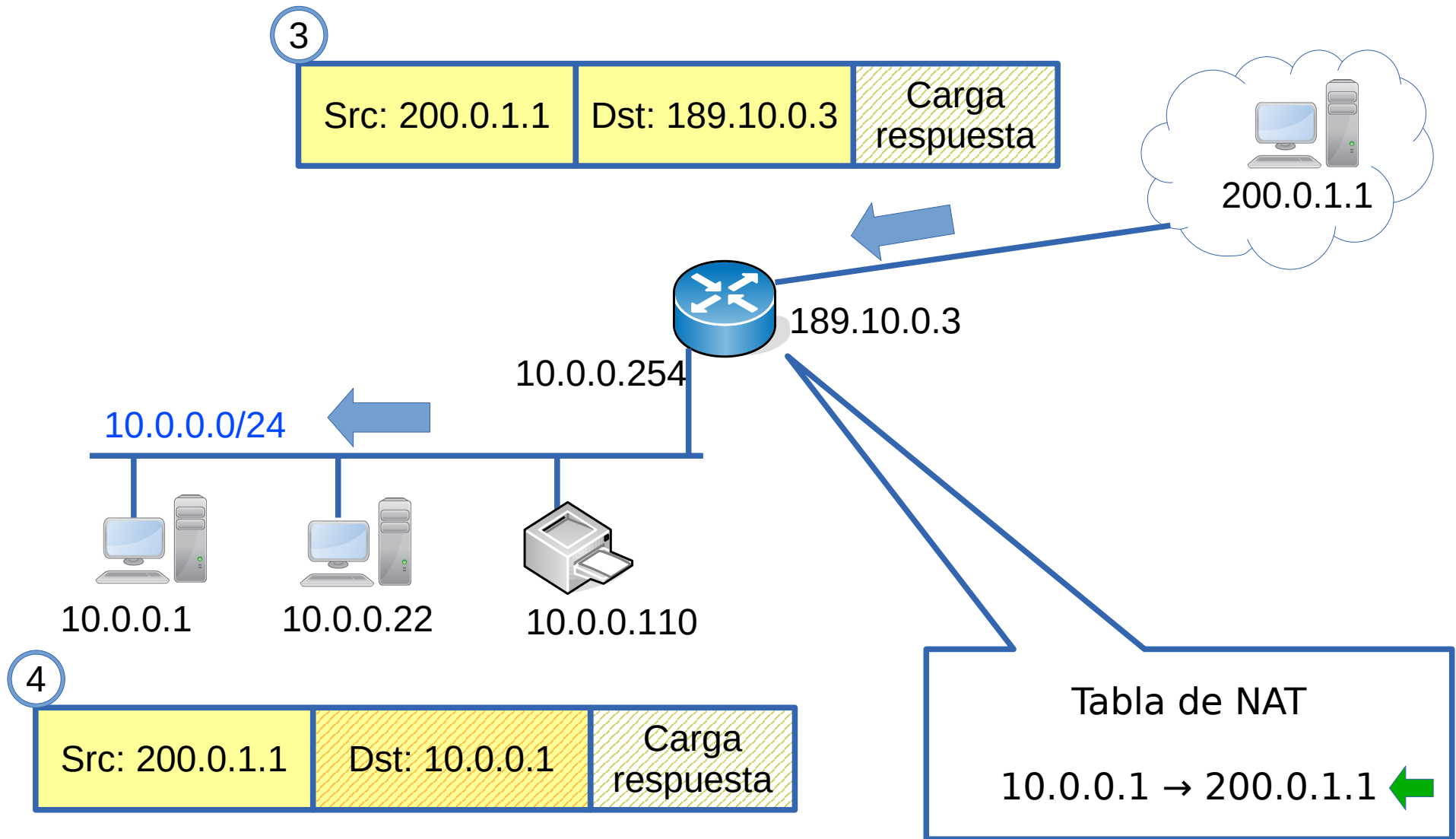
NAT: Network Address Translation

- Vuelta de los paquetes



NAT: Network Address Translation

- Vuelta de los paquetes



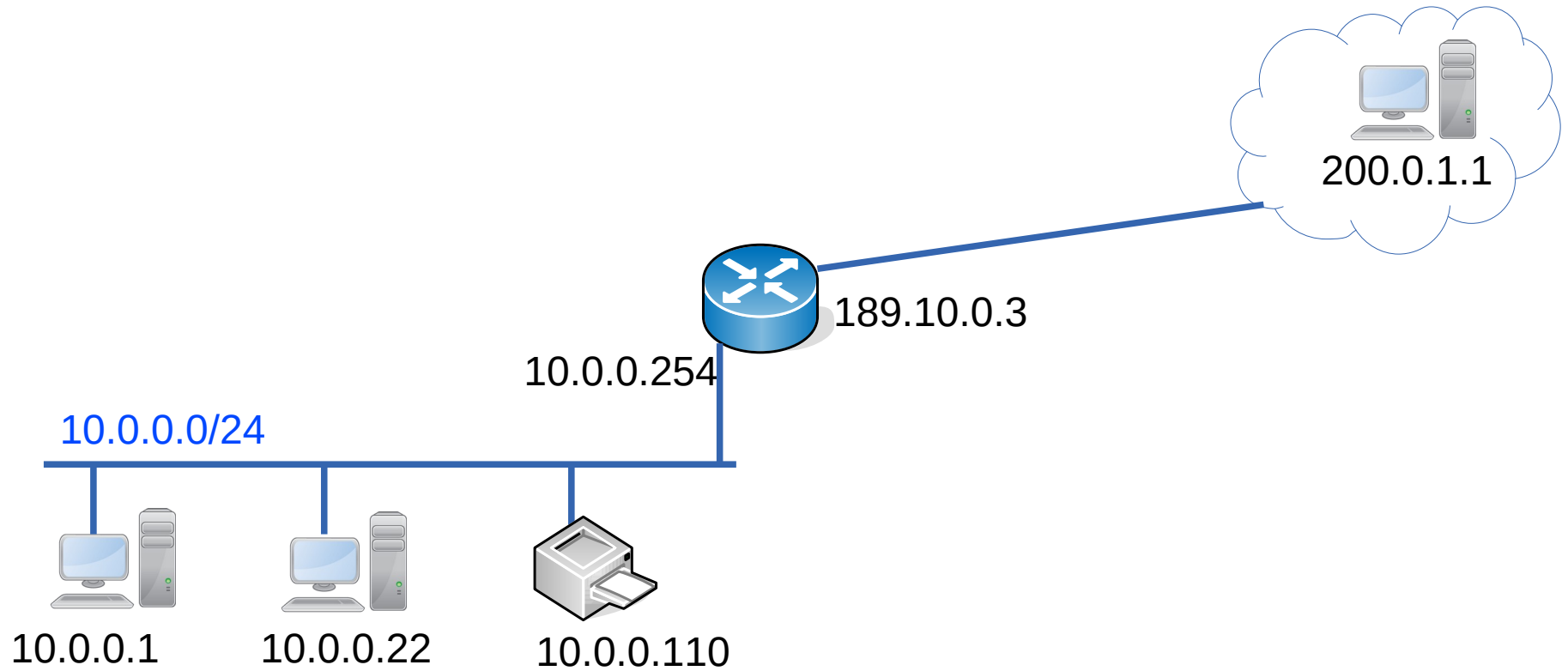
Direcciones para uso privado RFC 1918

- Rangos de direcciones para uso privado
 - 10.0.0.0/8
 - 10.0.0.0 a 10.255.255.255
 - 172.16.0.0/12
 - 172.16.0.0 a 172.31.255.255
 - 192.168.0.0/16
 - 192.168.0.0 a 192.168.255.255
- Estas direcciones no pueden aparecer (no tienen sentido) en la red pública y generalmente están filtradas por los enrutadores y los proveedores

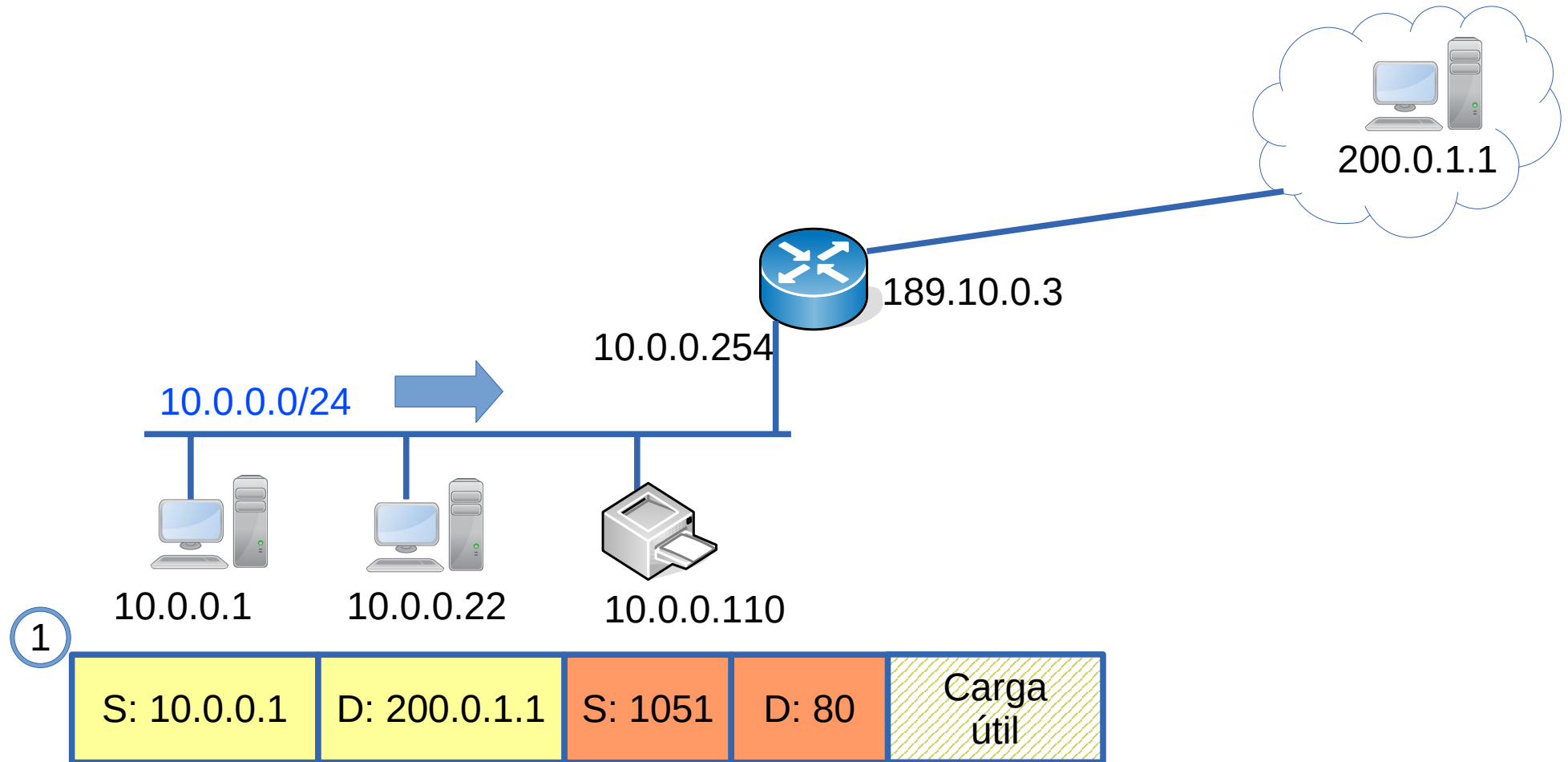
NAT: Algunas limitaciones

- Qué pasa si dos máquinas quieren salir a la vez ?
- Y si además quieren ir al mismo destino ?
 - Podría tener más de una dirección IP pública
- Solución más general:
 - NAT/PAT: Network Address Translation y Port Address Translation
 - Cambiar los puertos de capa de transporte (TCP o UDP)

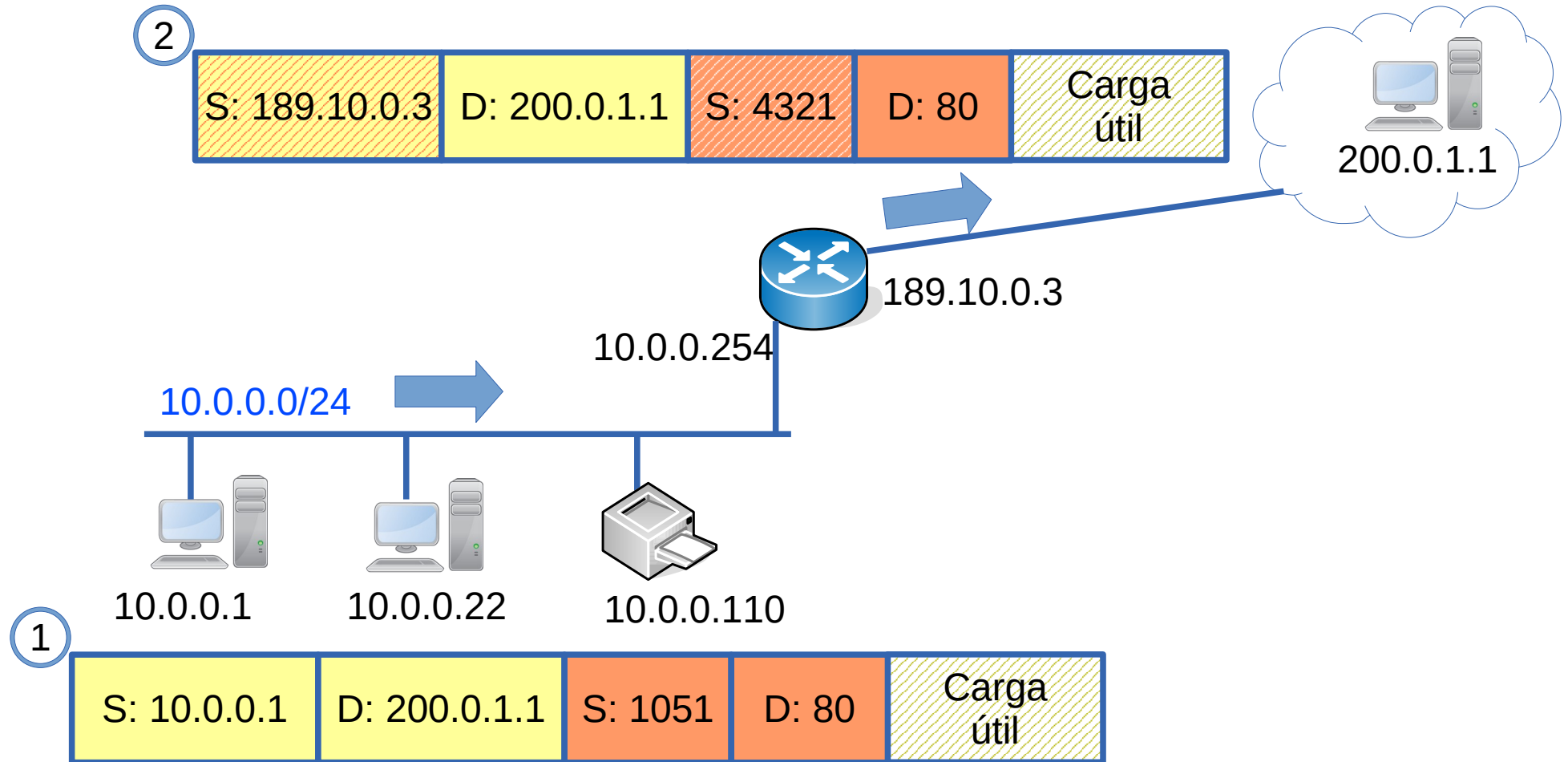
NAT/PAT



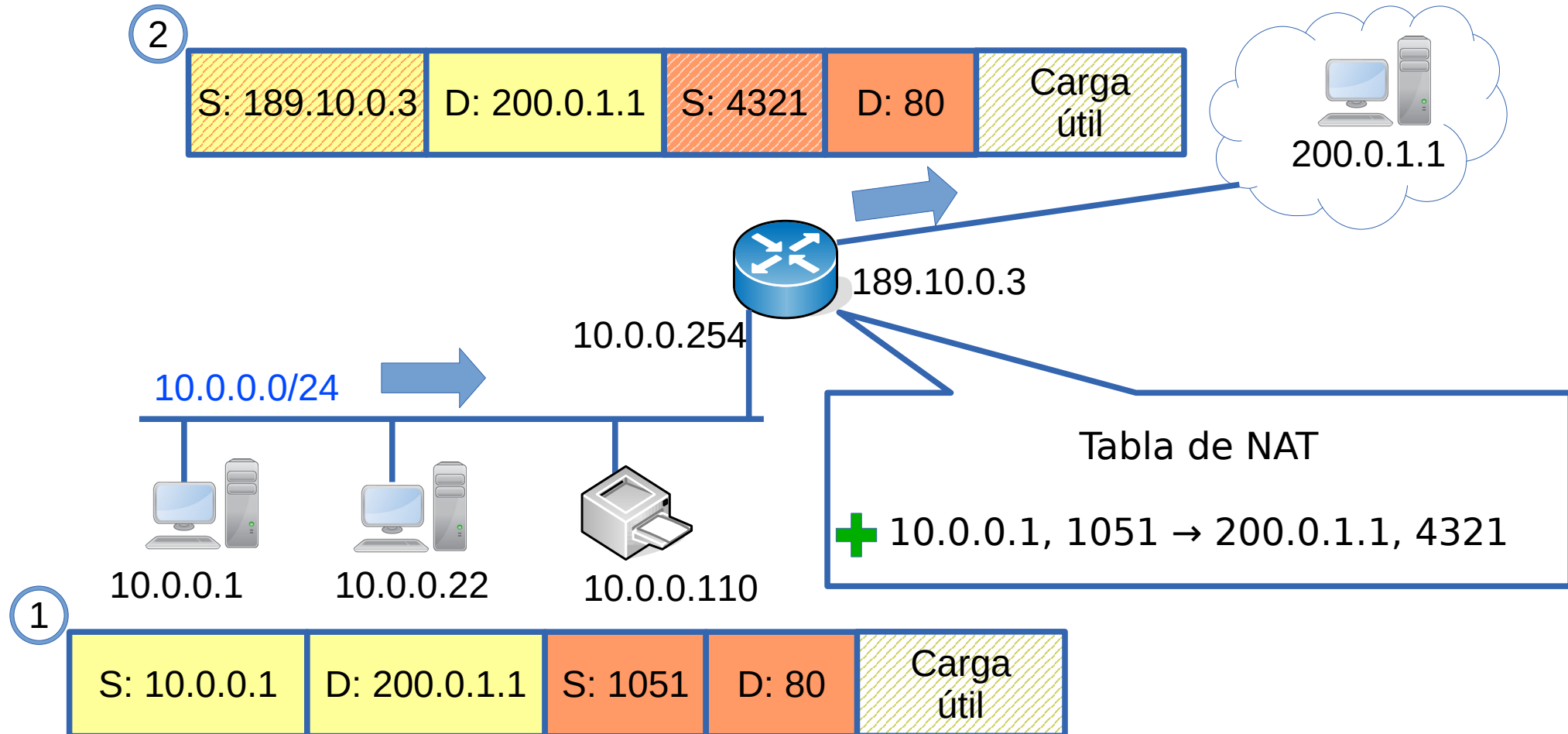
NAT/PAT



NAT/PAT

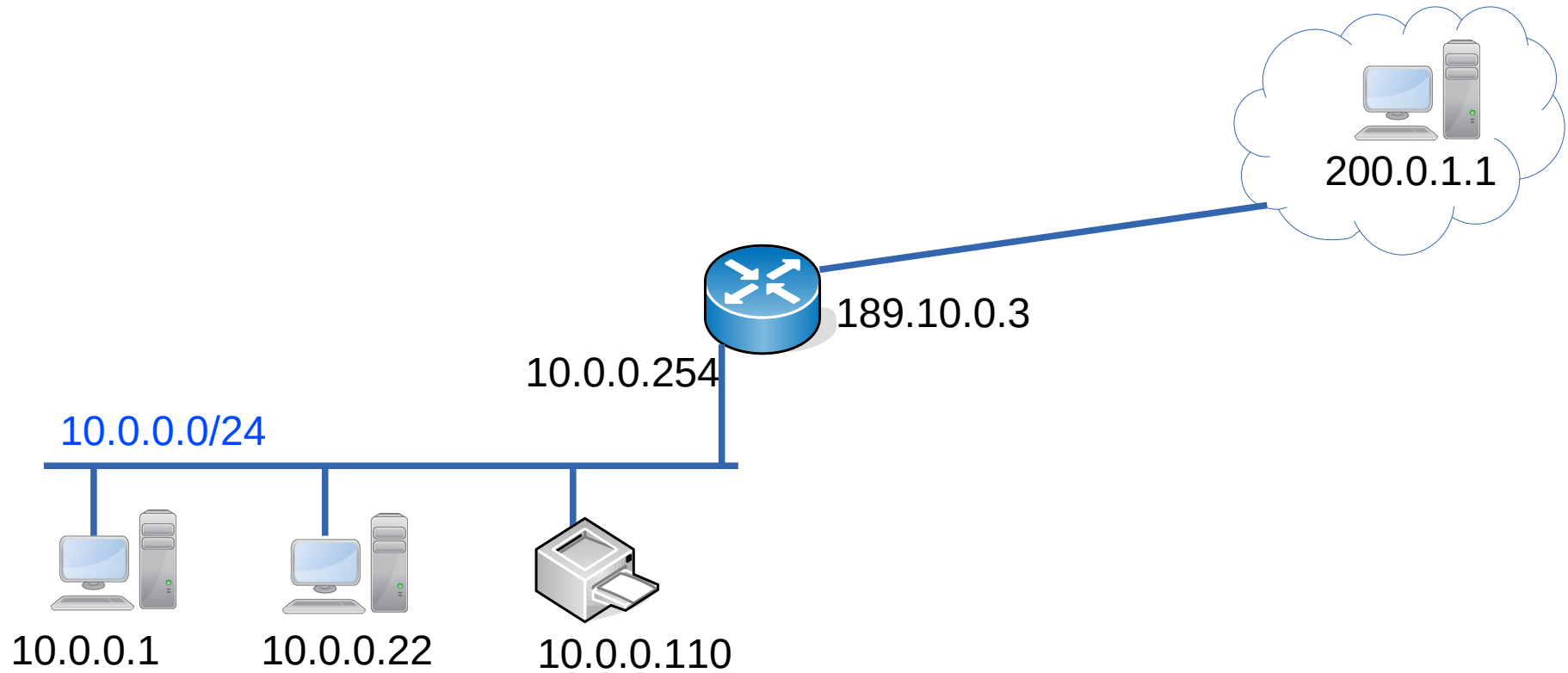


NAT/PAT



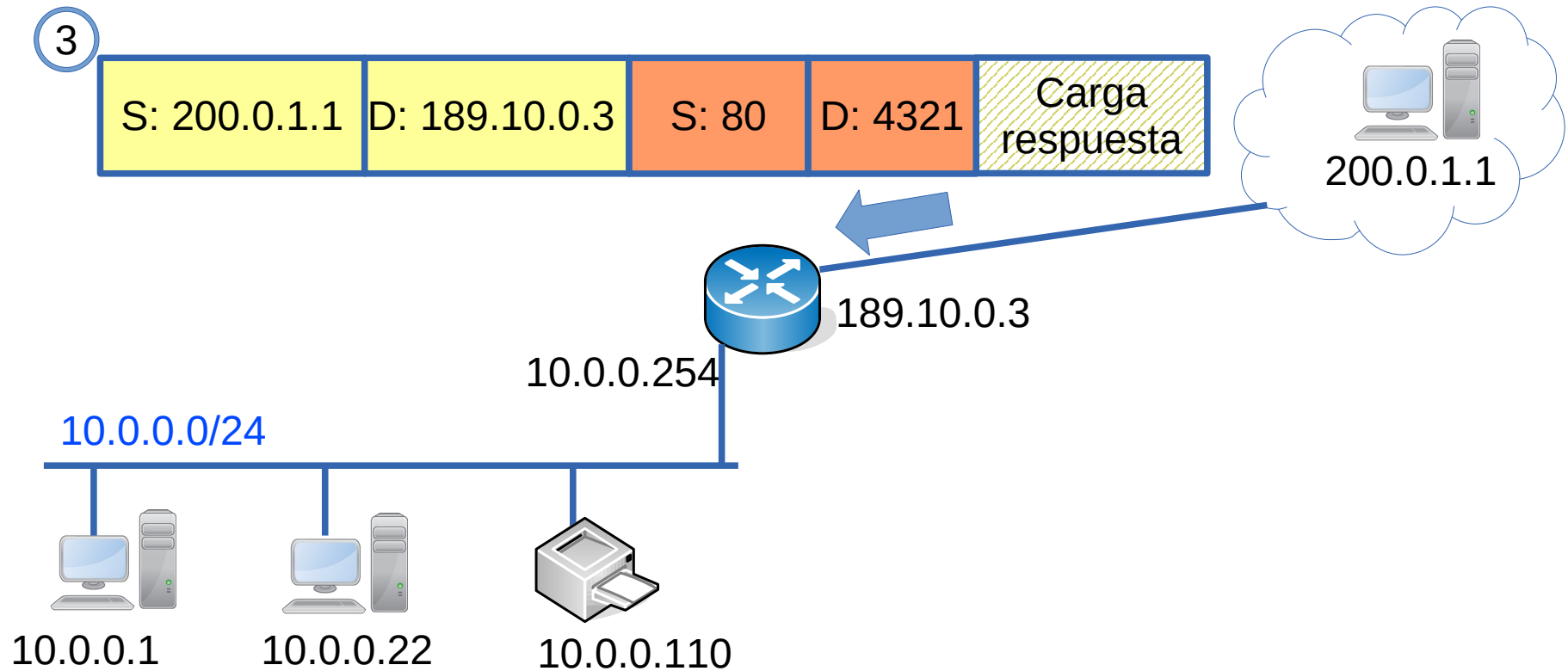
NAT/PAT

- Vuelta de los paquetes



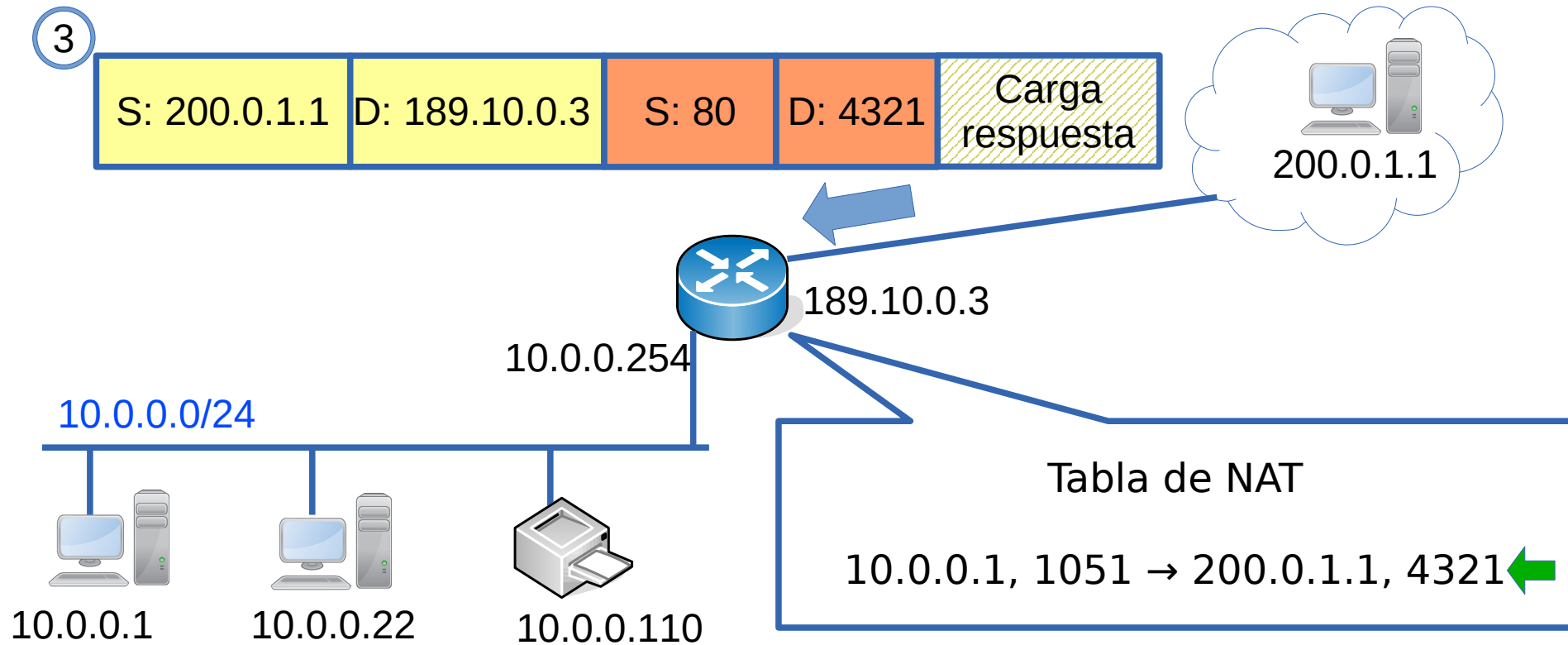
NAT/PAT

- Vuelta de los paquetes



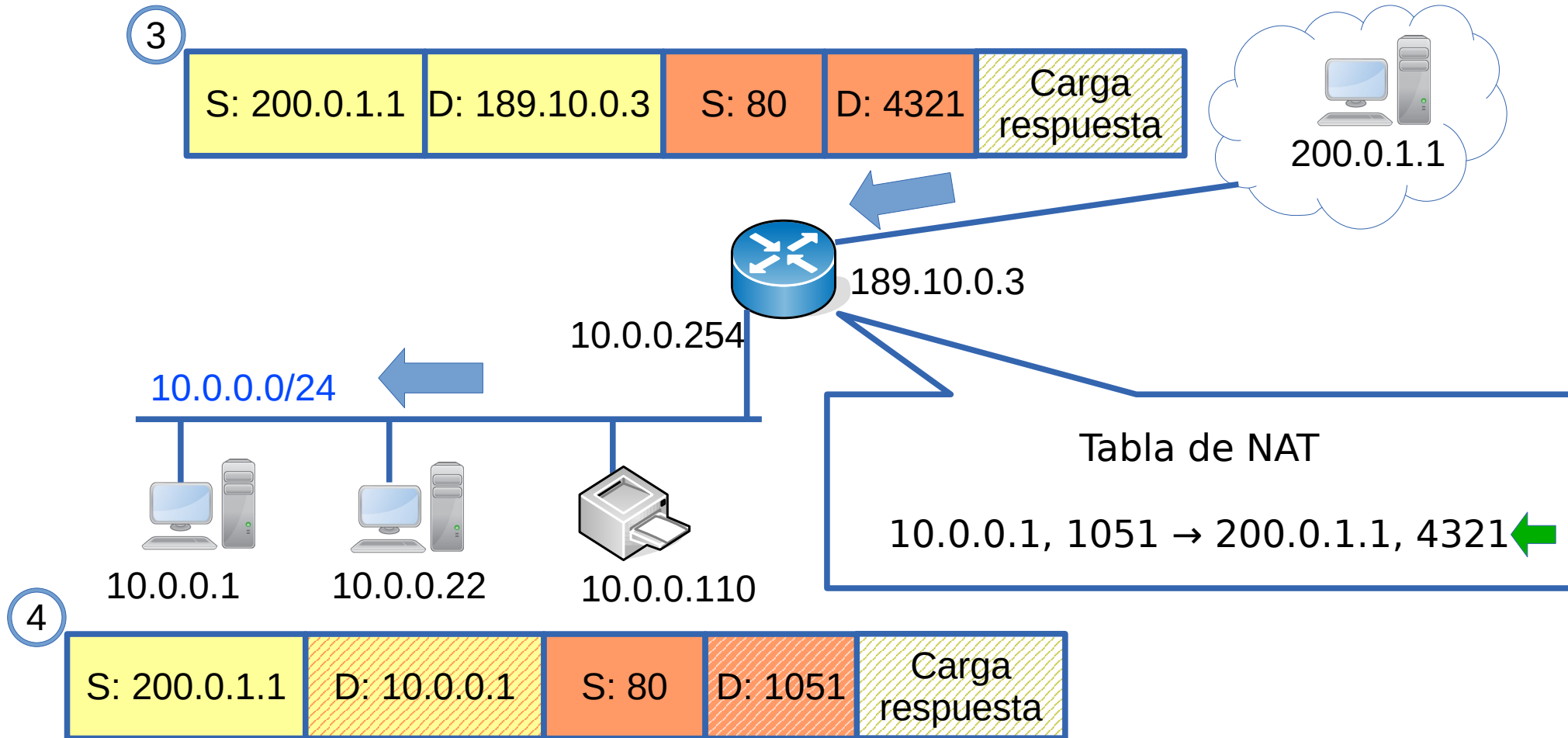
NAT/PAT

- Vuelta de los paquetes



NAT/PAT

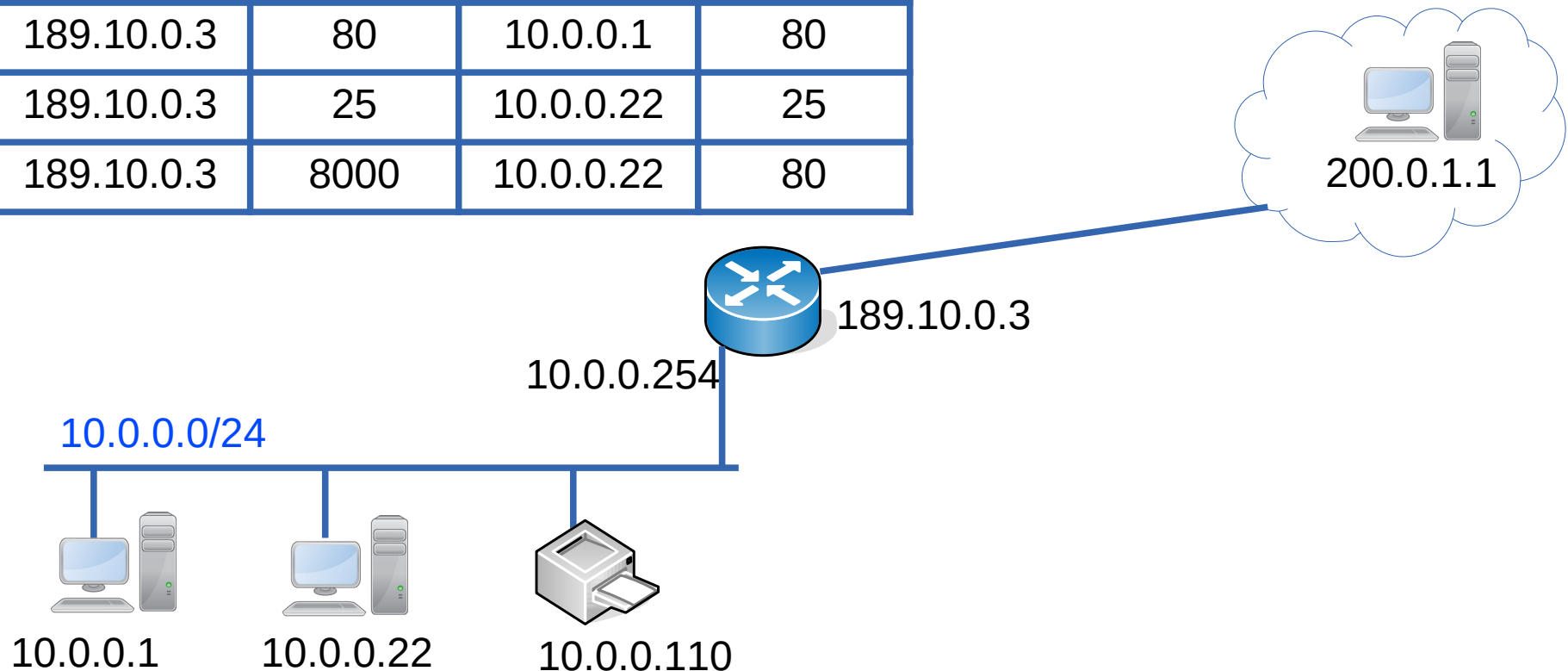
- Vuelta de los paquetes



Destination NAT

- ¿Cómo hago para brindar servicios accesibles desde Internet?
- ¿Cómo publico mi servidor web, mi servidor de correo?
- Se usa una tabla para mapear servicios públicos con servicios privados

IP Pública	Puerto Público	IP Privada	Puerto Privado
189.10.0.3	80	10.0.0.1	80
189.10.0.3	25	10.0.0.22	25
189.10.0.3	8000	10.0.0.22	80

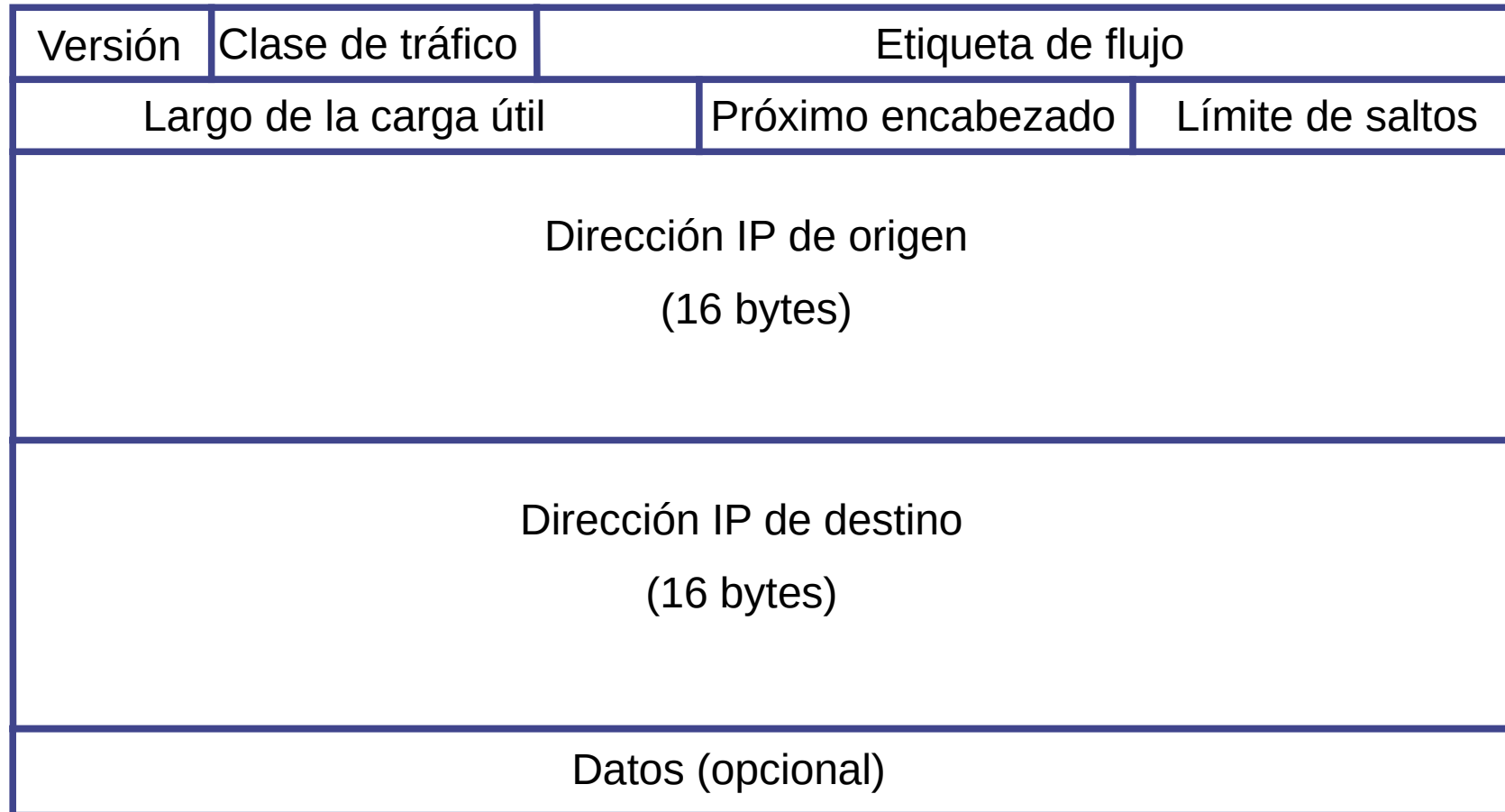


- En las redes de hogar, el router obtiene una dirección IP del proveedor y asigna un rango de IPs mediante DHCP a los equipos de la red privada
- Algunas aplicaciones tienen problemas con el NAT porque en sus mensajes de capa de aplicación incluyen las direcciones IP o puertos que están usando los equipos en los extremos de la conexión
 - Existen mecanismos de “NAT Traversal” para esos casos
- El NAT **esconde la topología** y las direcciones de la red interna
- El NAT **rompe principio end-to-end**:
 - Los nodos intermedios solamente deberían encaminar los paquetes de origen a destino sin analizarlos, modificarlos o procesarlos
 - Está vinculado al concepto de **Neutralidad de la red**
- Pero hay otras “**middle-boxes**” que también lo hacen:
 - Balance de carga
 - Priorización de tráfico
 - Firewalls
 - Intrusion detection y protection systems (IDS, IPS)

Escasez de direcciones: Protocolo IPv6

- A principios de los '90 se identificó que se necesitaba una nueva versión del protocolo IP, principalmente para aumentar el espacio de direcciones
- El NAT/PAT se incorporó como un mecanismo transitorio
- Además de aumentar el espacio de direcciones se aprovechó para mejorar otras cosas de IPv4
- Estado actual de IPv4 (<https://ipv4.potaroo.net/>, reporte de Febrero-2023):
 - IANA Unallocated Address Pool Exhaustion: 03-Feb-2011
 - RIR Address Pool Exhaustion Dates:
 - APNIC: 19-Apr-2011 (real)
 - RIPE NCC: 14-Sep-2012 (real)
 - LACNIC: 10-Jun-2014 (real)
 - ARIN: 24 Sep-2015 (real)
 - AFRINIC: 31-Dic-2021

Paquete IPv6



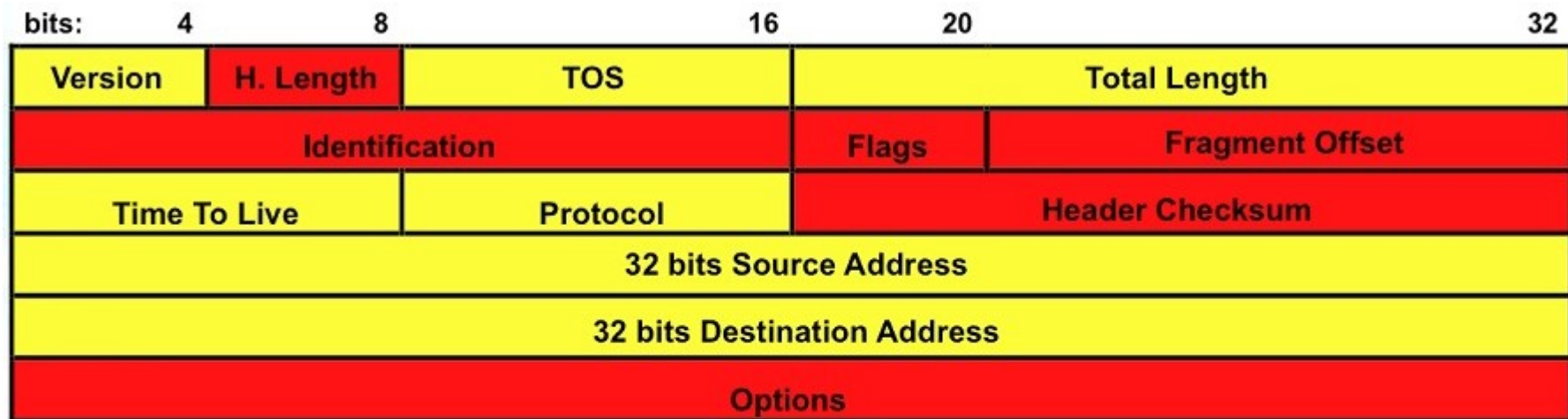
← 32 bits →

Principales cambios de IPv4 a IPv6

- Direccionamiento:
 - Se pasa de direcciones de 32 bits a **128 bits!!**
 - Hay 2^{128} direcciones (aproximadamente 3×10^{38})
 - Hay 7×10^{23} direcciones por metro cuadrado del planeta (incluyendo el agua!!)
 - Además de las direcciones unicast (una interfaz) o multicast (varias interfaces, en particular broadcast) existentes en IPv4, se definen las **direcciones anycast** que permiten direccionar a una dirección de un grupo de equipos
 - Usadas para distribución de contenido
- **Simplificación** del encabezado
 - Para encaminamiento más rápido en los enrutadores
 - Encabezado de tamaño fijo
 - Eliminación del checksum
 - No se permite fragmentación en equipos intermedios

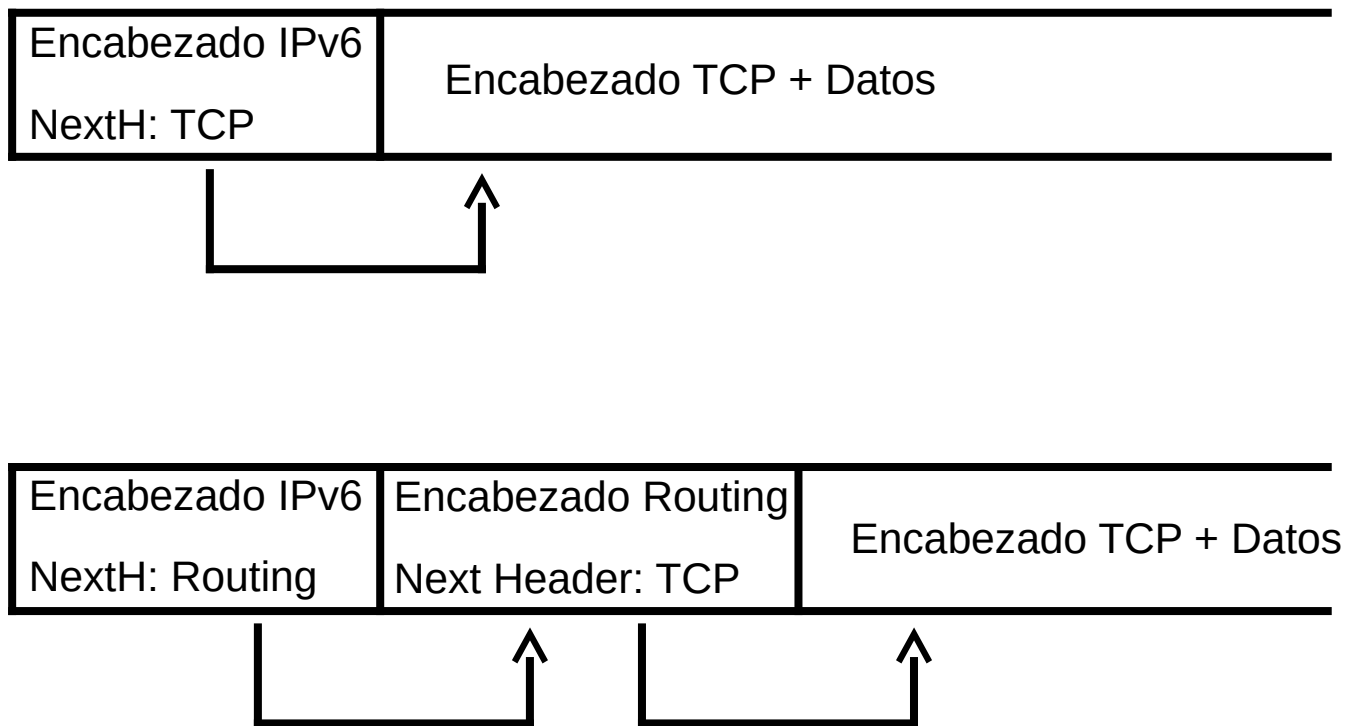
Principales cambios de IPv4 a IPv6

- Revisión de parámetros:
 - largo (de carga útil), protocolo (next header), TTL (hop limit)
 - versión
- Identificadores de flujo
 - Para permitir tráfico con calidad diferenciada
 - Nuevos campos flow label, traffic class (tipo de servicio en IPv4)



Principales cambios de IPv4 a IPv6

- El campo Opciones de IPv4 ahora se implementa con encabezados de extensión (extension headers)



Redes de datos 1

Capa de red Plano de control

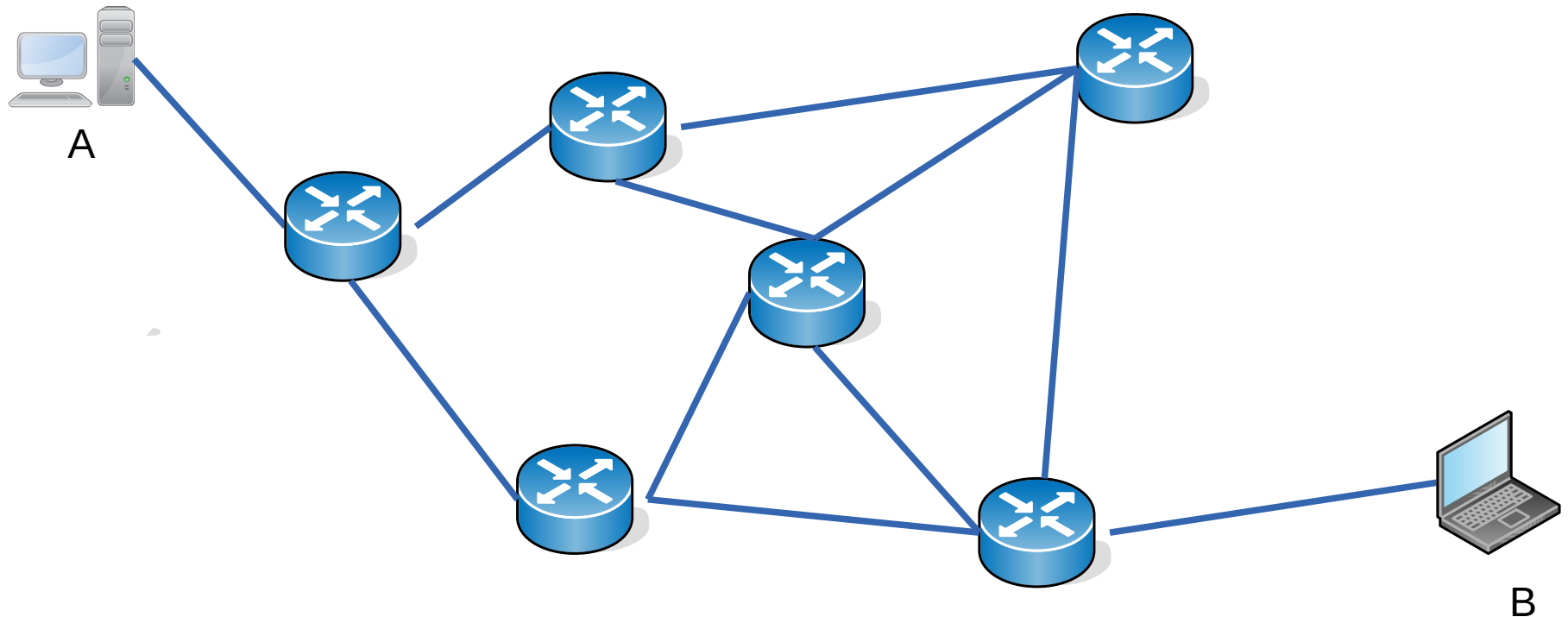
Facultad de Ingeniería – Universidad de la República
Instituto de Ingeniería Eléctrica

Agenda

- Conceptos de capa de red
- Plano de datos
- Plano de control
 - Algoritmos de ruteo
 - Clasificación de algoritmos
 - Algoritmos de estado del enlace
 - Algoritmos de vector distancia
 - Internet Control Message Protocol (ICMP)
- Redes de circuitos virtuales

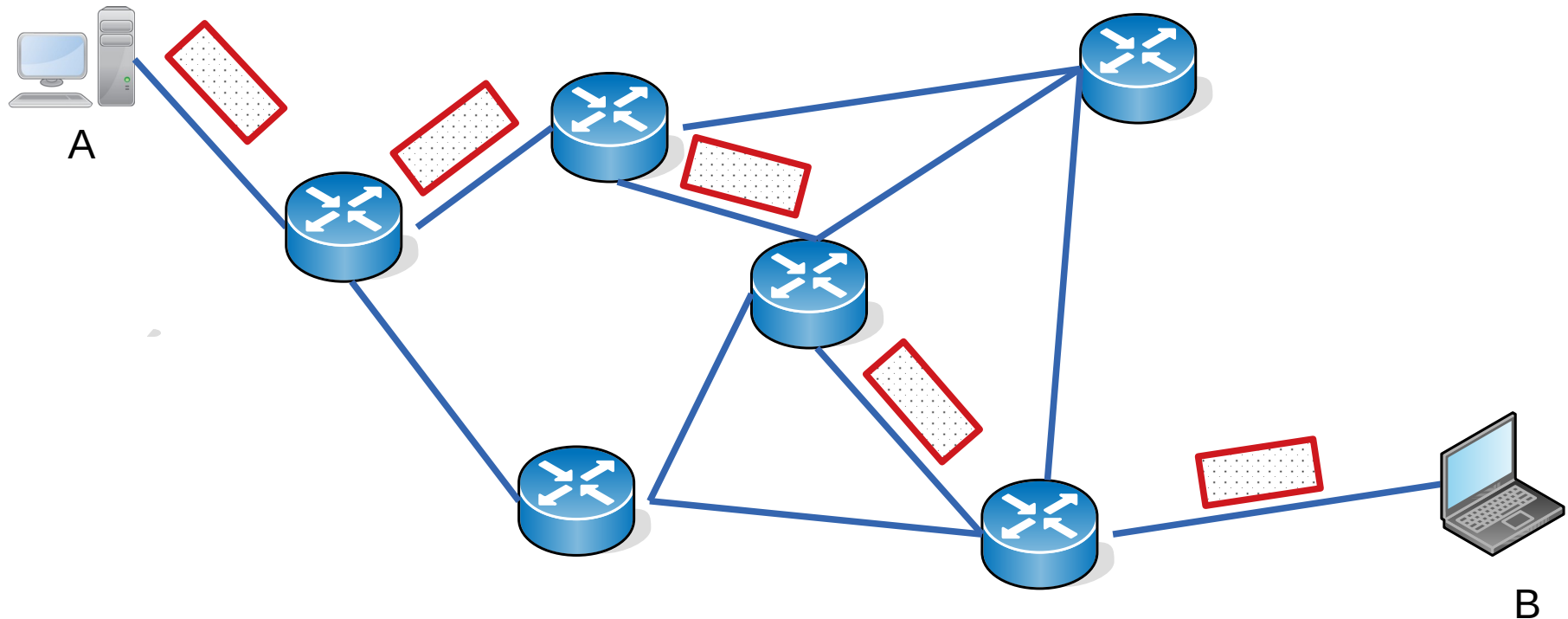
Función de Ruteo

- Cuál es el **mejor camino** para ir de A a B?



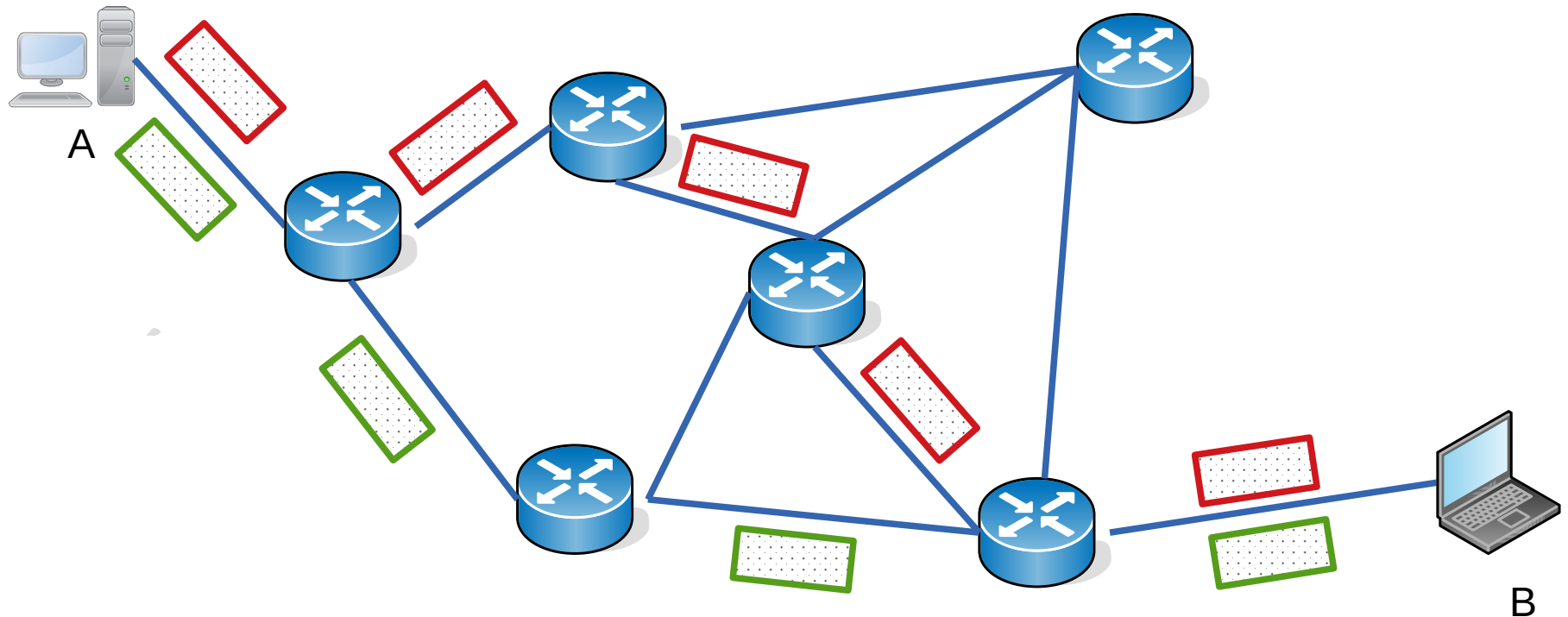
Función de Ruteo

- Cuál es el **mejor camino** para ir de A a B?



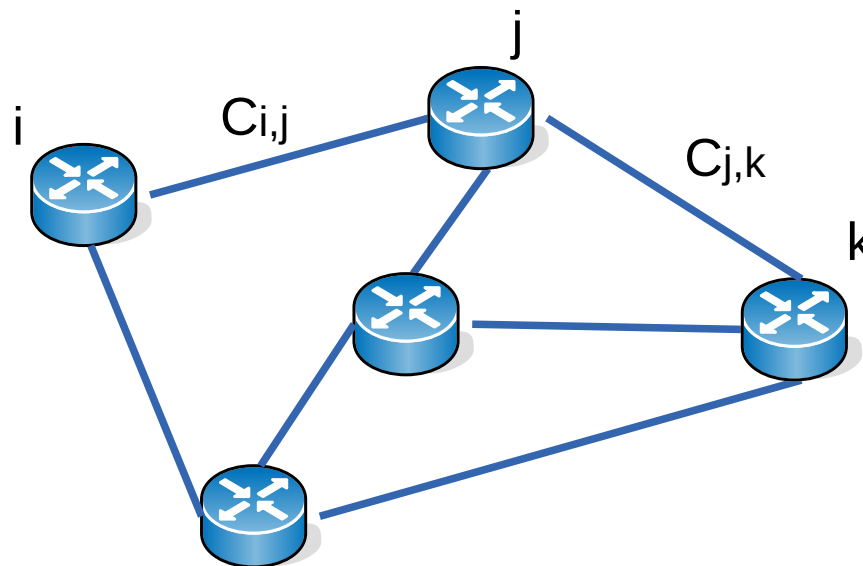
Función de Ruteo

- Cuál es el **mejor camino** para ir de A a B?



Algoritmos de ruteo

- Se utilizan **algoritmos** para encontrar los mejores caminos desde todos los posibles orígenes a todos los posibles destinos
- Se hace una abstracción de la red a un **grafo con pesos o costos**
 - Los pesos o costos pueden ser los largos de los enlaces, las velocidades, el costo en \$
- El costo del camino entre A y B será la suma de los costos de cada tramo que compone el camino
- Interesa obtener el “**camino de menor costo**” (least-cost path) o “camino más corto” (shortest path)



Clasificación de algoritmos de ruteo

- Estáticos
 - Se realiza el cálculo una vez
 - Se configuran manualmente las tablas de forwarding de los enrutadores en base a los caminos elegidos
 - No se adaptan a los cambios, requieren procedimientos manuales si cambia la topología de la red o los costos de los enlaces
- Dinámicos
 - Permiten cambiar los caminos automáticamente teniendo en cuenta los cambios de la topología o los costos de los enlaces de la red
 - Observación: Los algoritmos que se usan actualmente en Internet no son sensibles a la carga de los enlaces
 - Pueden recalcular periódicamente los caminos o recalcularlos cuando se producen cambios en la red (de topología o de costos)
 - Requieren intercambio de información para enterarse de los cambios de topología
 - Las tablas de forwarding se actualizan dinámicamente en función de los caminos calculados

Clasificación de algoritmos de ruteo

- Centralizados

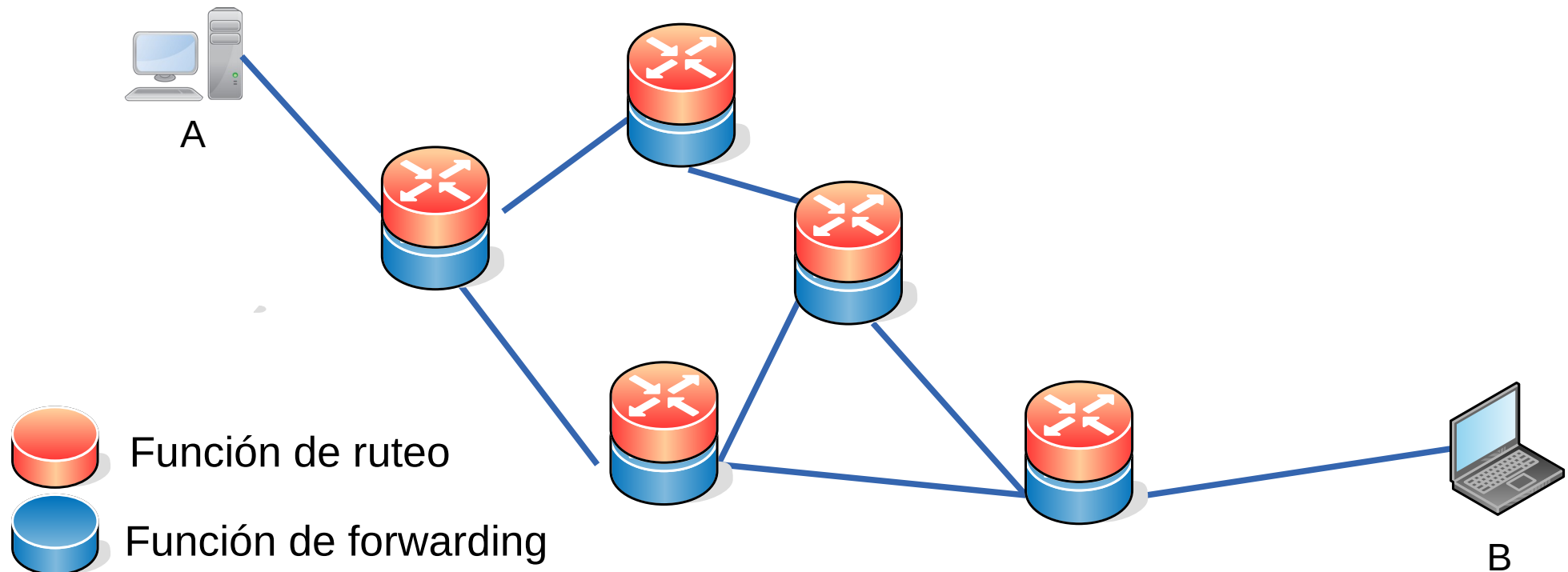
- Se conoce el grafo de la red con sus nodos, enlaces y costos
- Estos algoritmos se conocen como de “estado del enlace” o “link-state (LS)” ya que conocen el costo (o estado) de todos los enlaces de la red
- El cálculo se puede hacer en cada enrutador (arquitectura tradicional) o en un controlador central (arquitectura SDN)

- Descentralizados

- Los caminos se calculan de forma iterativa y distribuida en los enrutadores
- No hay un nodo que tenga la información completa de la red
- Cada nodo comienza sabiendo los costos de los enlaces con sus vecinos y mediante intercambios de información con ellos, iterativamente va aprendiendo cómo llegar a los demás destinos
- Se conocen como de “vector distancia” o “distance-vector (DV)” ya que cada enrutador mantiene una tabla de distancias aprendidas a los destinos de la red

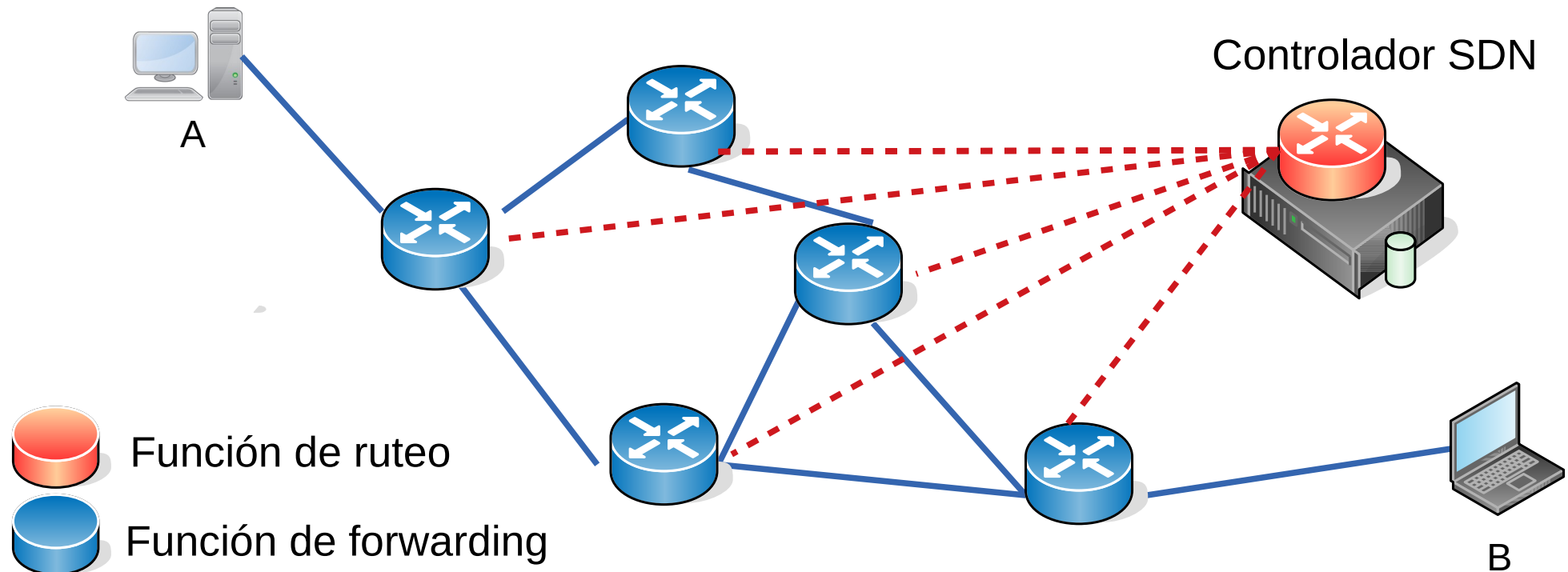
Implementación en arquitectura tradicional

- Se pueden decidir los caminos (con un algoritmo) y luego configurar las tablas de forwarding de forma manual o estática
- Los enrutadores (además de implementar la función de forwarding), pueden implementar ruteo dinámico (con un algoritmo centralizado o descentralizado)
- Además se requieren **protocolos de ruteo** para intercambiar información con los demás enrutadores



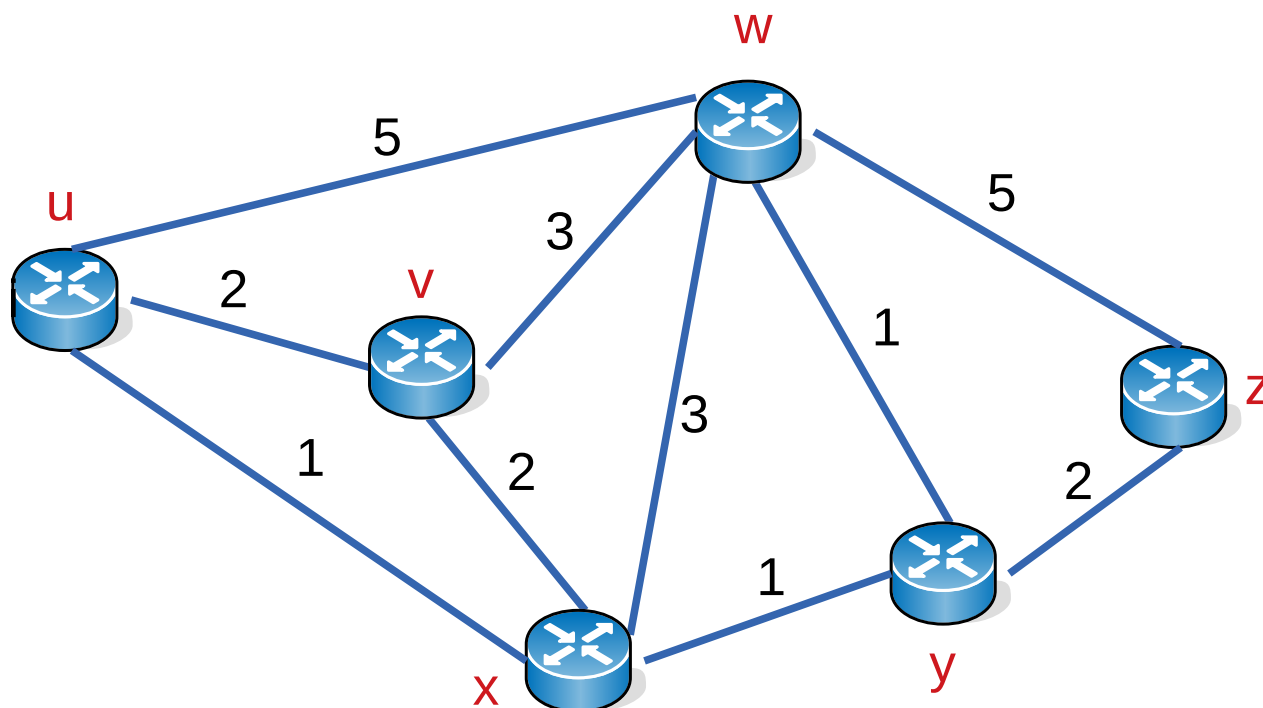
Implementación en la arquitectura SDN

- El plano de control (ruteo) y de datos (forwarding) se hace en diferentes equipos
- **Plano de Control:** En el controlador SDN (y sus aplicaciones) se deciden los caminos (con un algoritmo centralizado) y se actualizan las tablas de forwarding a los conmutadores
- **Plano de Datos:** conmutadores especializados en el forwarding de paquetes a alta velocidad



Algoritmos de estado del enlace

- [Link State routing algorithms](#) (LS)
- En la arquitectura tradicional, cada nodo conoce el grafo con nodos, enlaces y costos. Los enrutadores dialogan entre si usando protocolos específicos para intercambiar esa información
- En la arquitectura SDN, el controlador tiene la información de toda la red porque dialoga con los conmutadores y conoce la topología.
- El algoritmo más conocido y usado es el de [Dijkstra](#)
- Permite calcular el camino de menor costo desde cada origen a cada destino

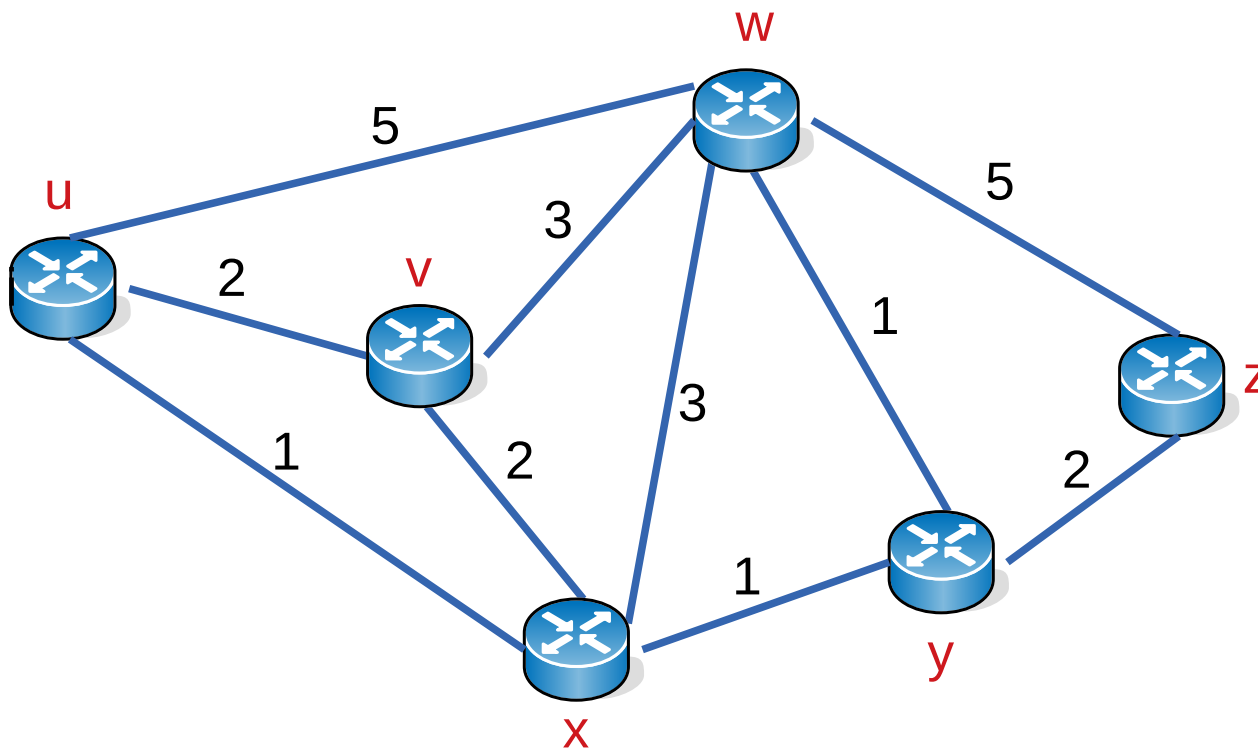


Dijkstra: Notación

- Es un algoritmo iterativo que dado un nodo origen permite, en sucesivos pasos, calcular la distancia y camino a los demás nodos
- Notación:
 - N : conjunto de nodos del grafo
 - $c(i, j)$: costo del enlace entre i y j
 - $D(i)$: distancia del nodo i al nodo origen en un paso dado
 - $p(i)$: vecino de i por el que se llega al origen con el menor costo en un paso dado
 - N' : conjunto de nodos cuya distancia ya está definitivamente conocida

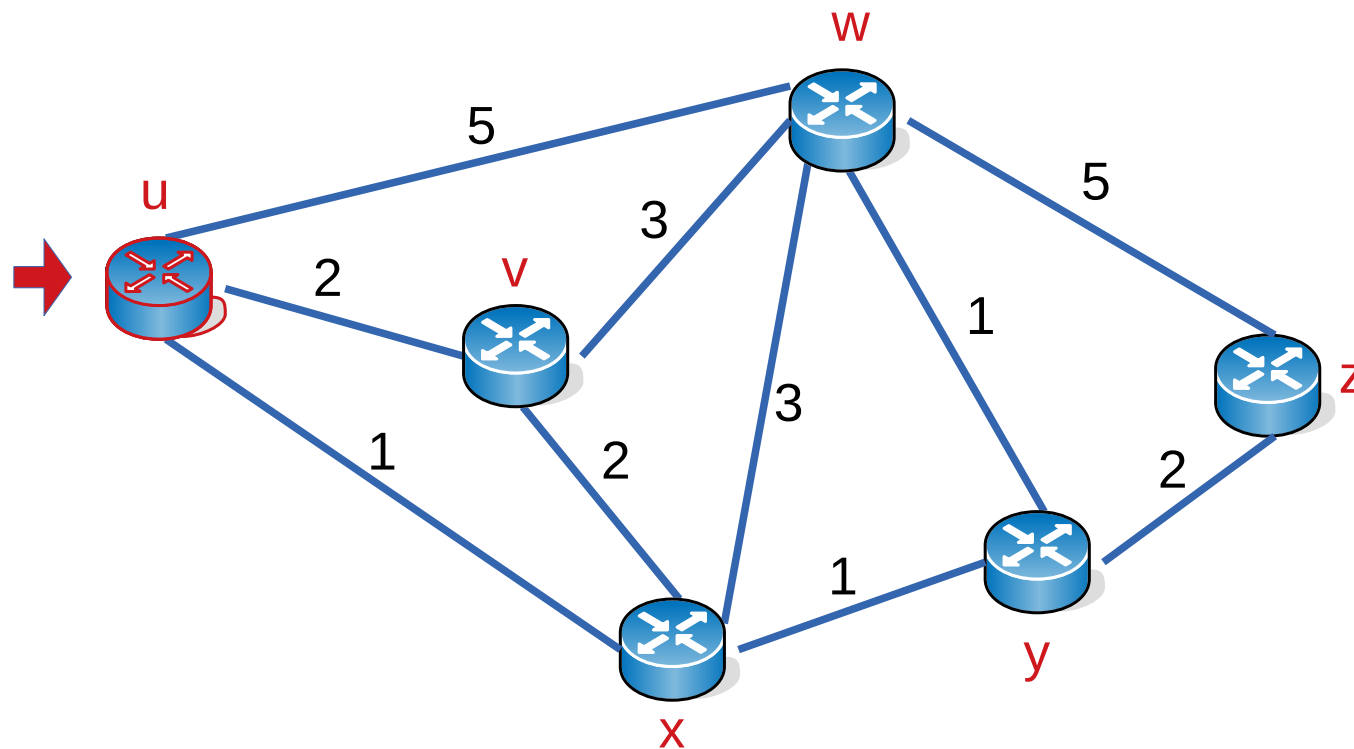
Dijkstra - Nodo origen: u

Paso	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y), p(y)	D(z), p(z)
0	u	2, u	5, u	1, u	∞	∞



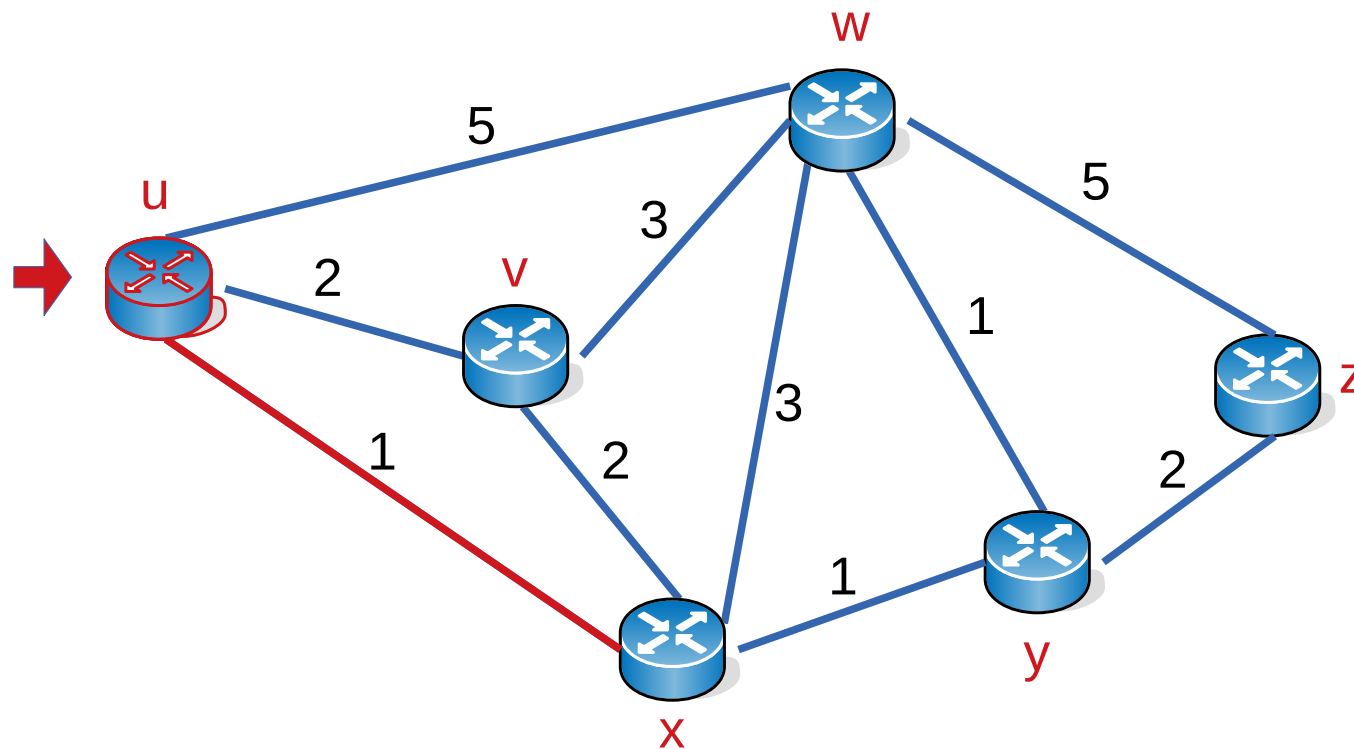
Dijkstra - Nodo origen: u

Paso	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y), p(y)	D(z), p(z)
0	u	2, u	5, u	1, u	∞	∞



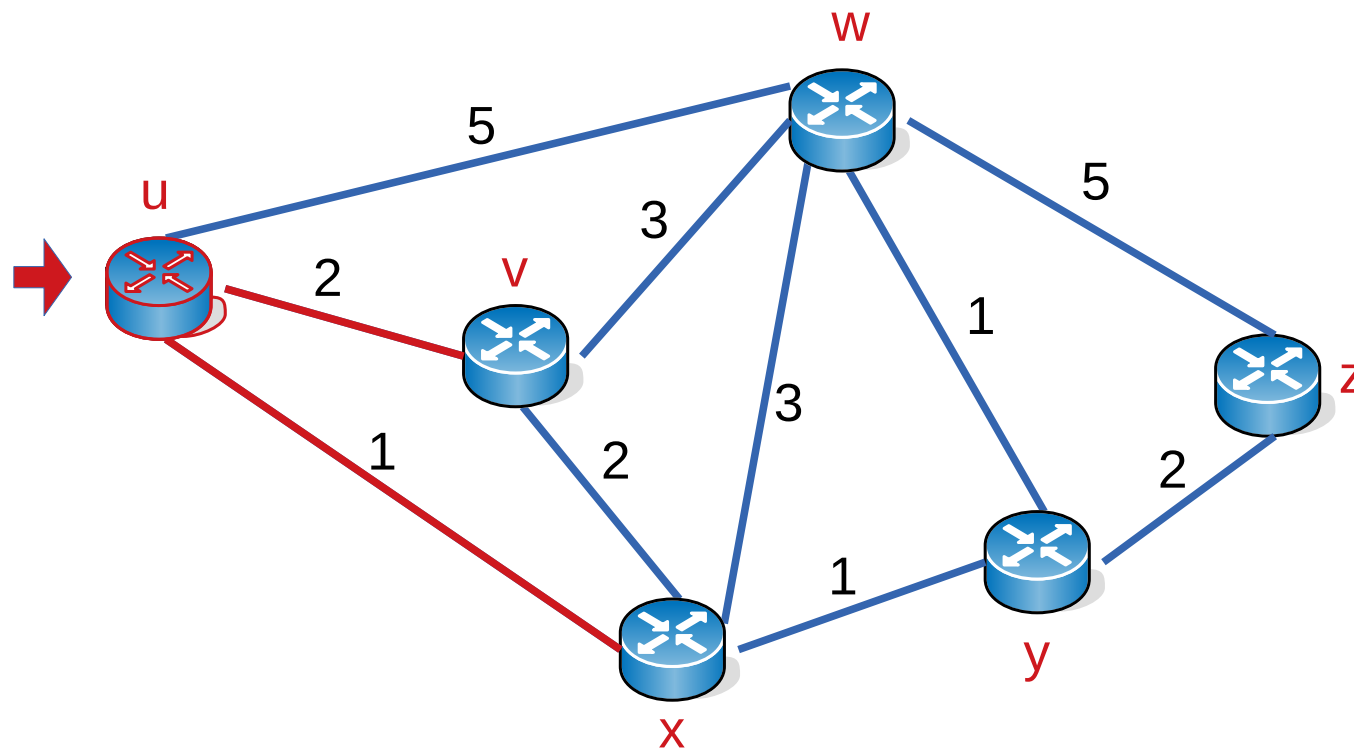
Dijkstra - Nodo origen: u

Paso	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y), p(y)	D(z), p(z)
0	u	2, u	5, u	1, u	∞	∞



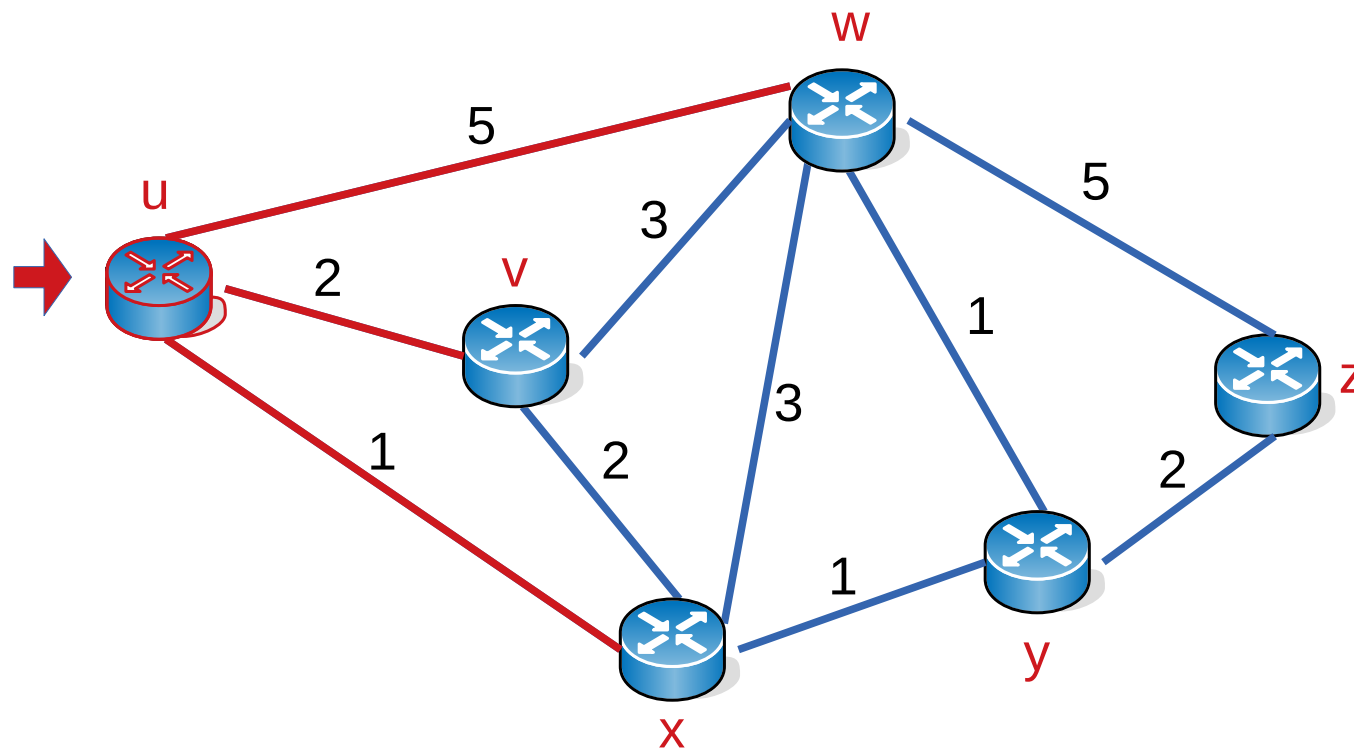
Dijkstra - Nodo origen: u

Paso	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y), p(y)	D(z), p(z)
0	u	2, u	5, u	1, u	∞	∞



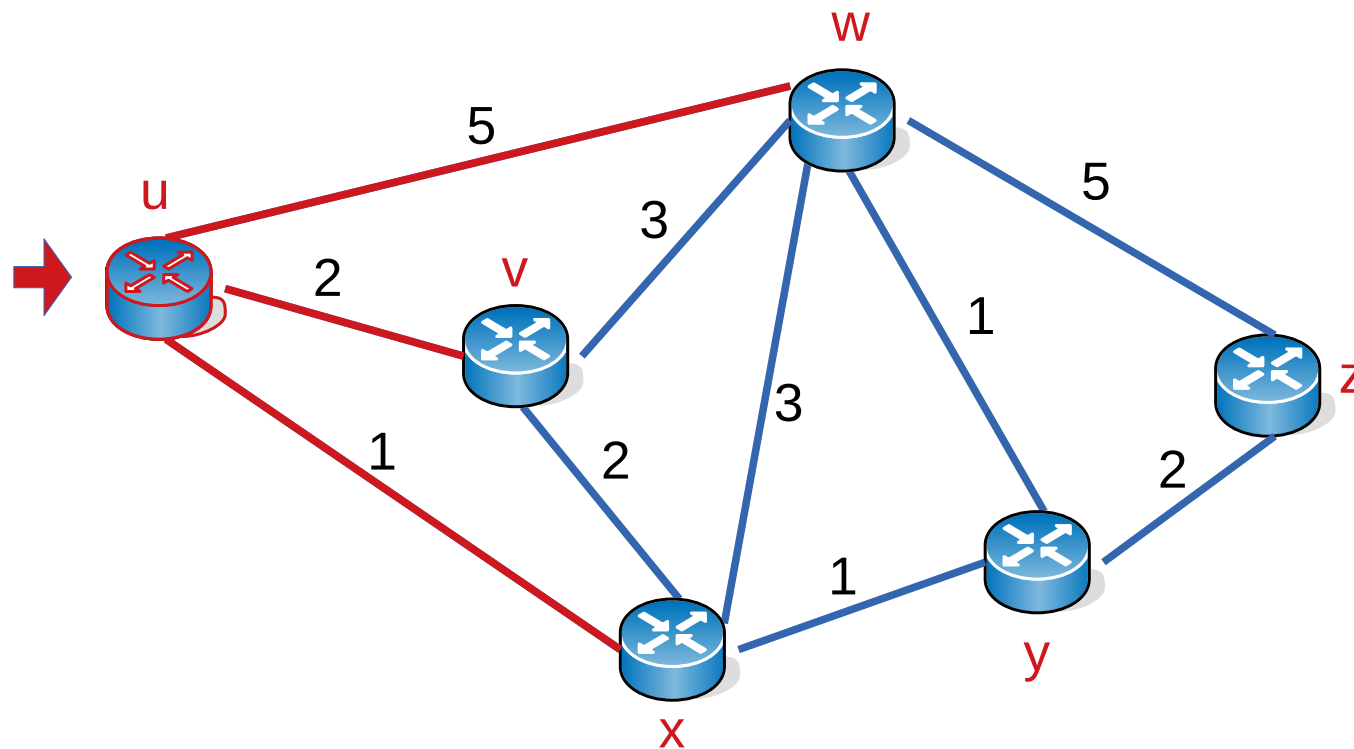
Dijkstra - Nodo origen: u

Paso	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y), p(y)	D(z), p(z)
0	u	2, u	5, u	1, u	∞	∞



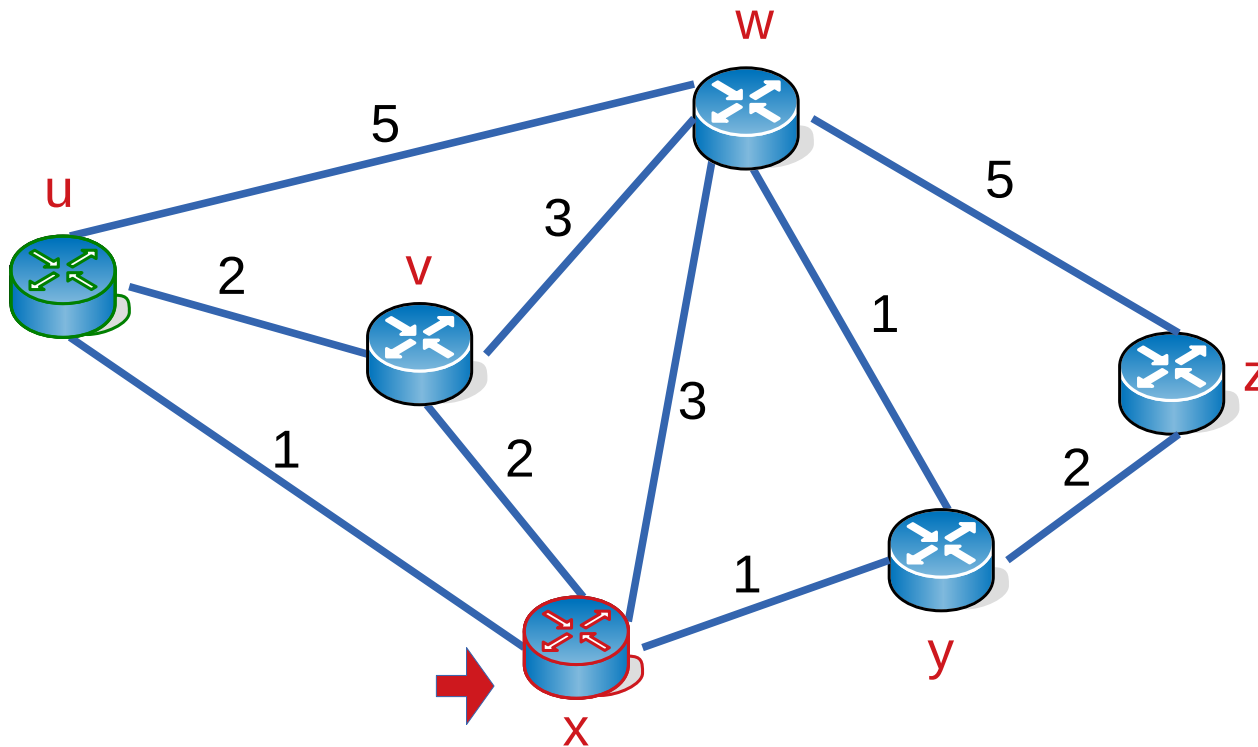
Dijkstra - Nodo origen: u

Paso	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y), p(y)	D(z), p(z)
0	u	2, u	5, u	1, u	∞	∞



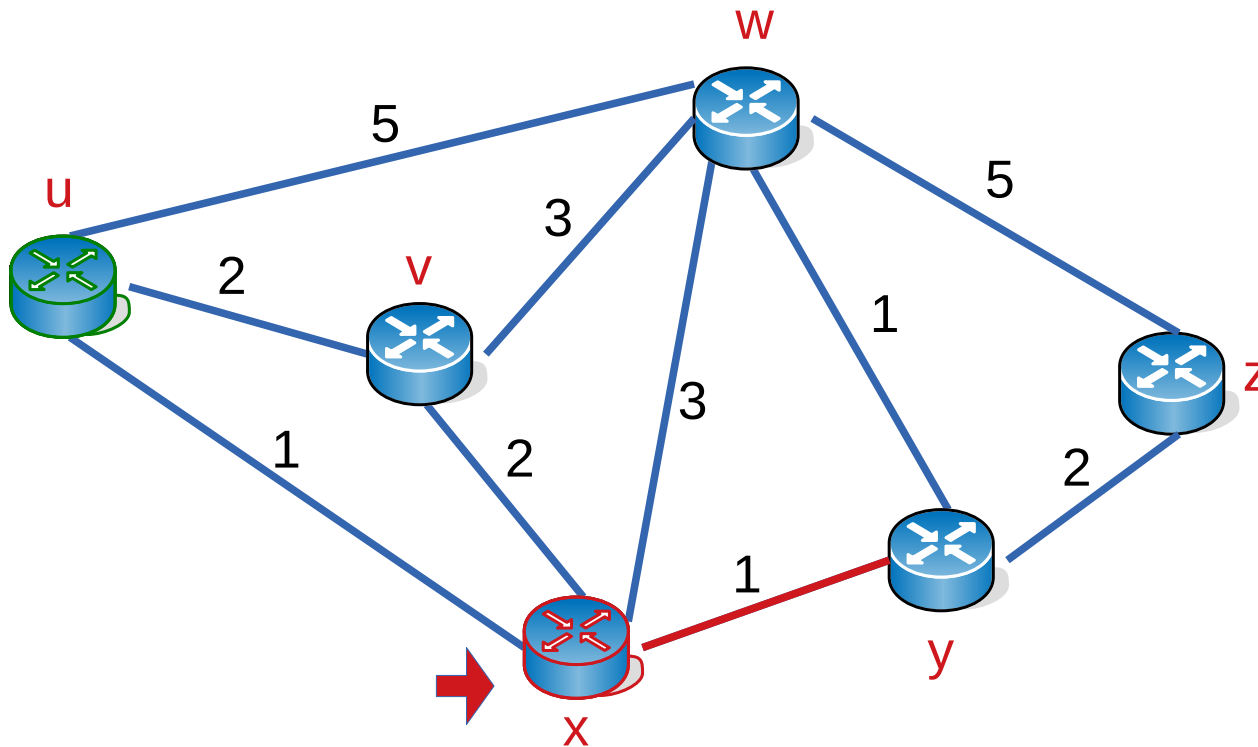
Dijkstra - Nodo origen: u

Paso	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y), p(y)	D(z), p(z)
0	u	2, u	5, u	1, u	∞	∞
1	ux	2, u	4, x		2, x	∞



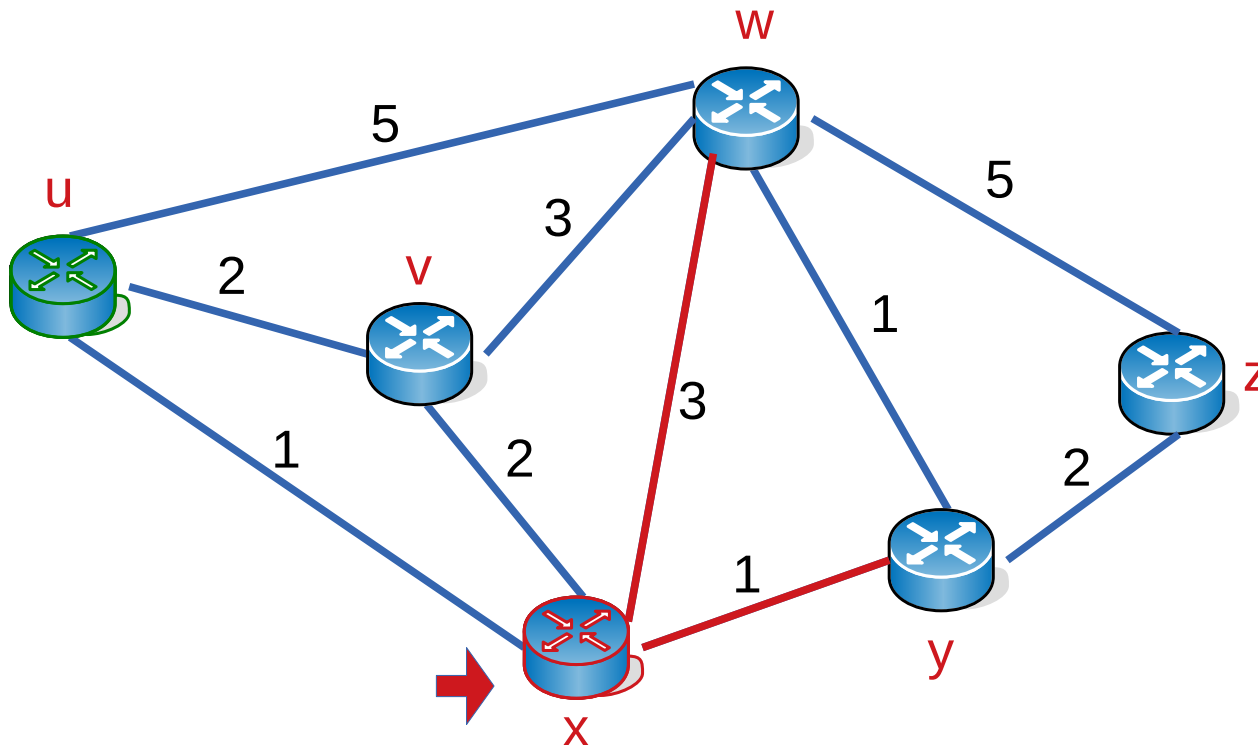
Dijkstra - Nodo origen: u

Paso	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y), p(y)	D(z), p(z)
0	u	2, u	5, u	1, u	∞	∞
1	ux	2, u	4, x		2, x	∞



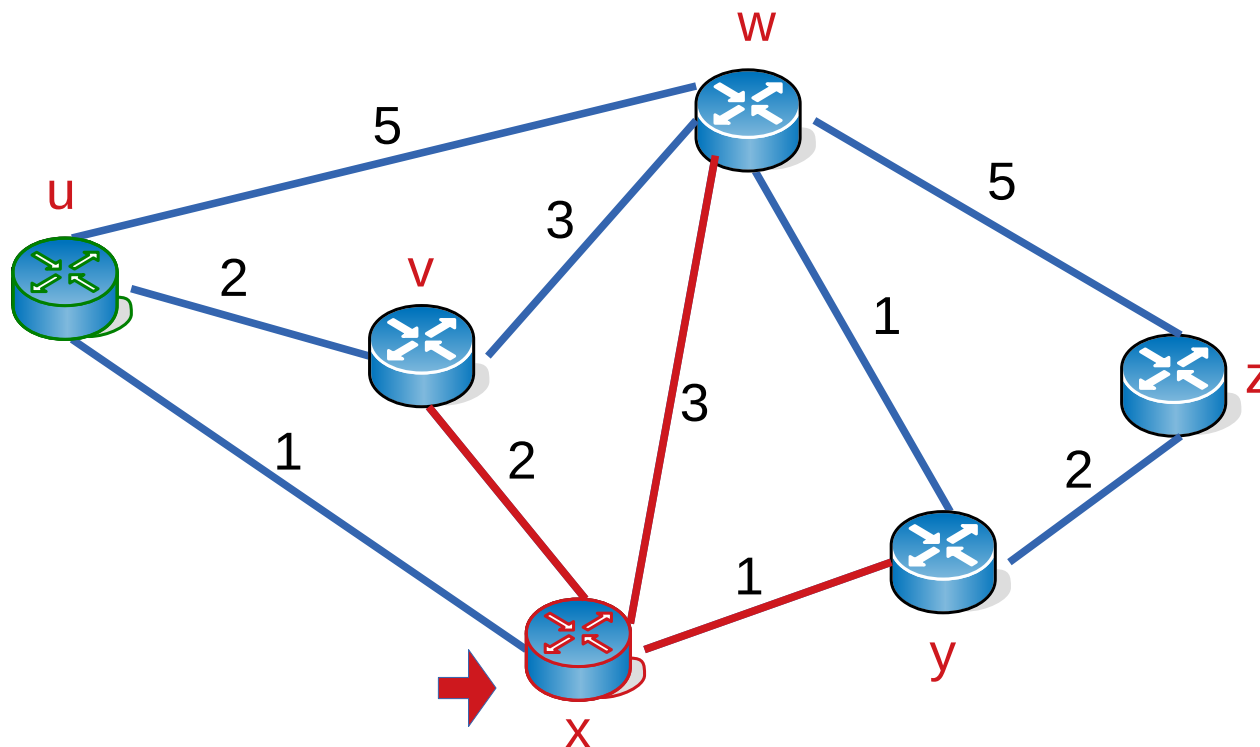
Dijkstra - Nodo origen: u

Paso	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y), p(y)	D(z), p(z)
0	u	2, u	5, u	1, u	∞	∞
1	ux	2, u	4, x		2, x	∞



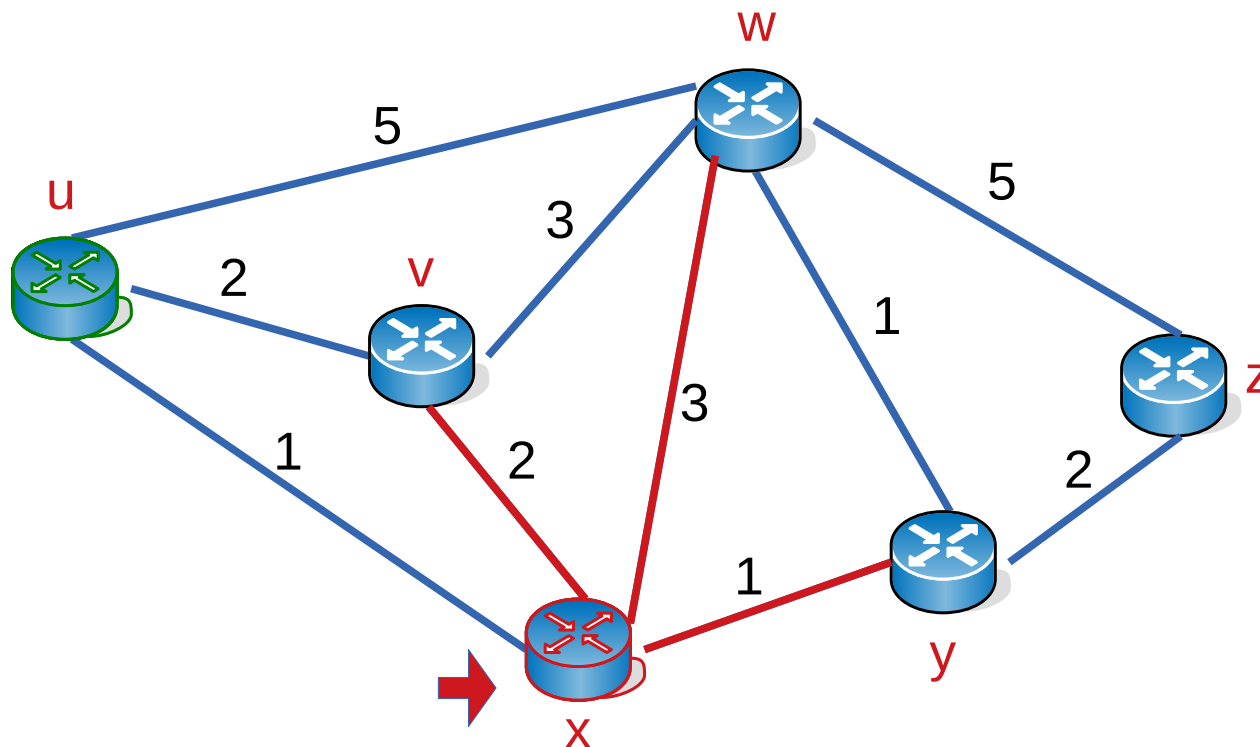
Dijkstra - Nodo origen: u

Paso	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y), p(y)	D(z), p(z)
0	u	2, u	5, u	1, u	∞	∞
1	ux	2, u	4, x		2, x	∞



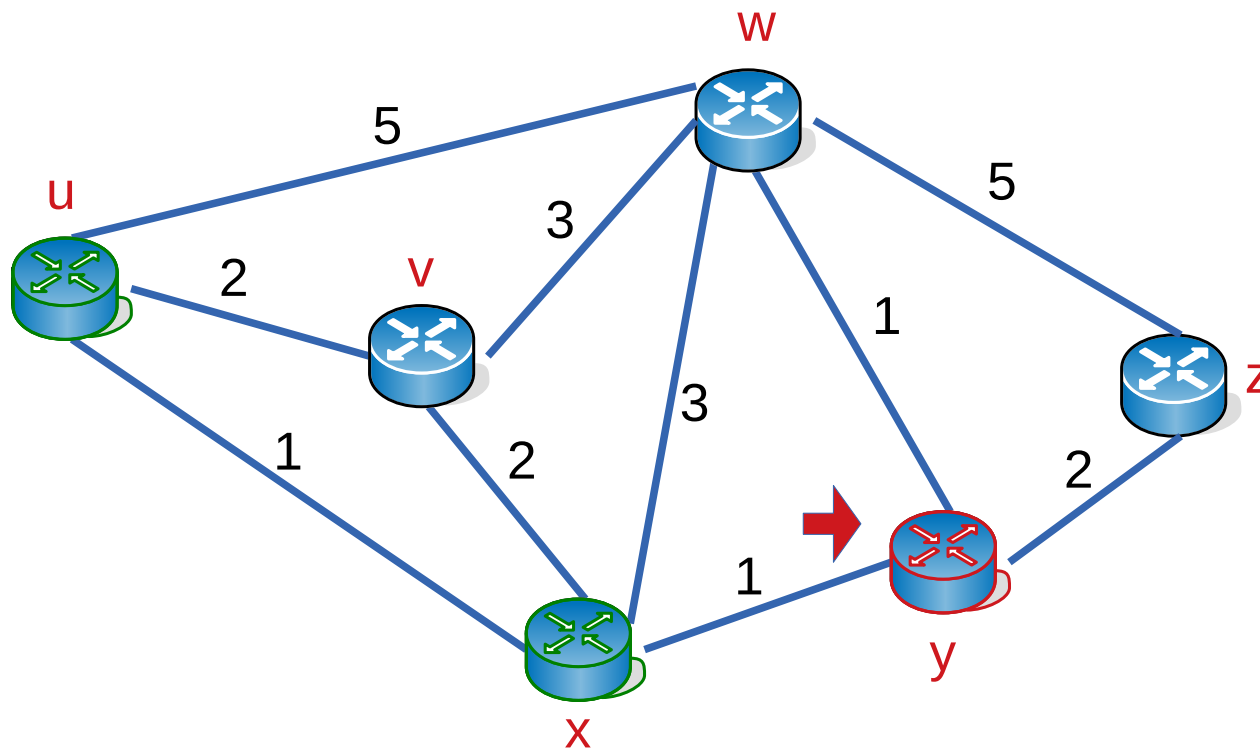
Dijkstra - Nodo origen: u

Paso	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y), p(y)	D(z), p(z)
0	u	2, u	5, u	1, u	∞	∞
1	ux	2, u	4, x		2, x	∞



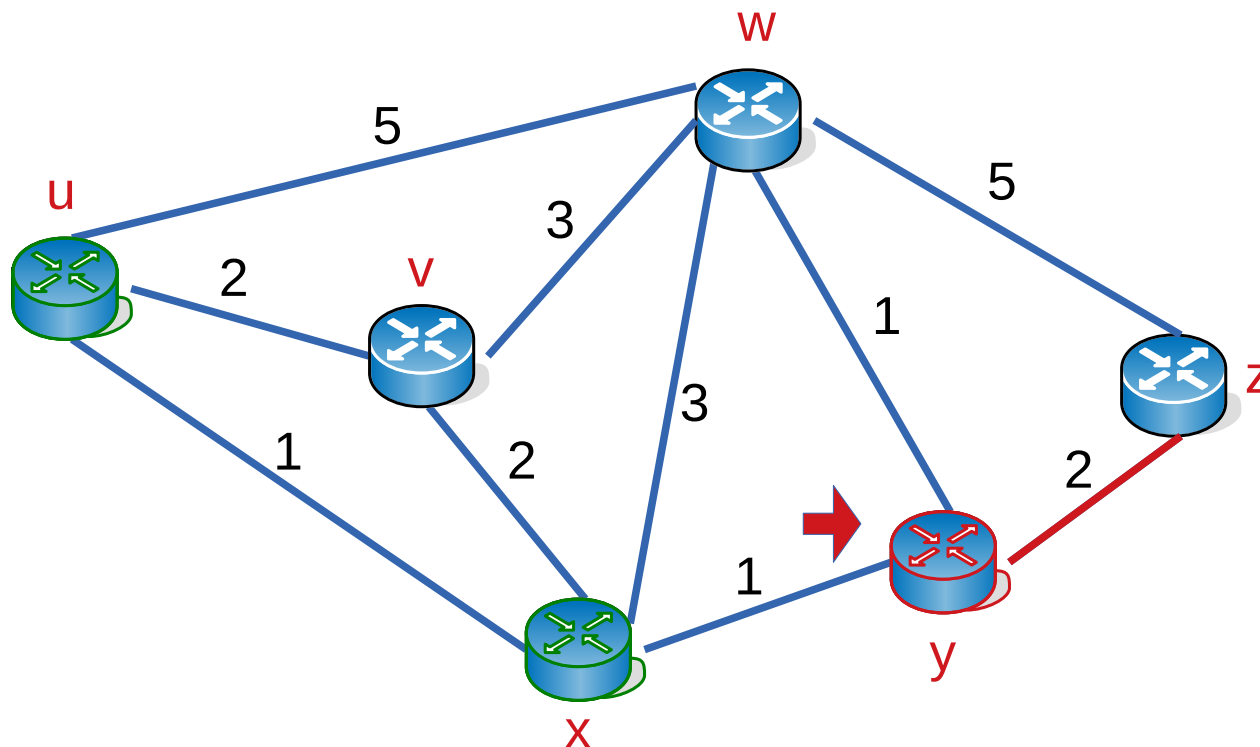
Dijkstra - Nodo origen: u

Paso	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y), p(y)	D(z), p(z)
0	u	2, u	5, u	1, u	∞	∞
1	ux	2, u	4, x		2, x	∞
2	uxy	2, u	3, y			4, y



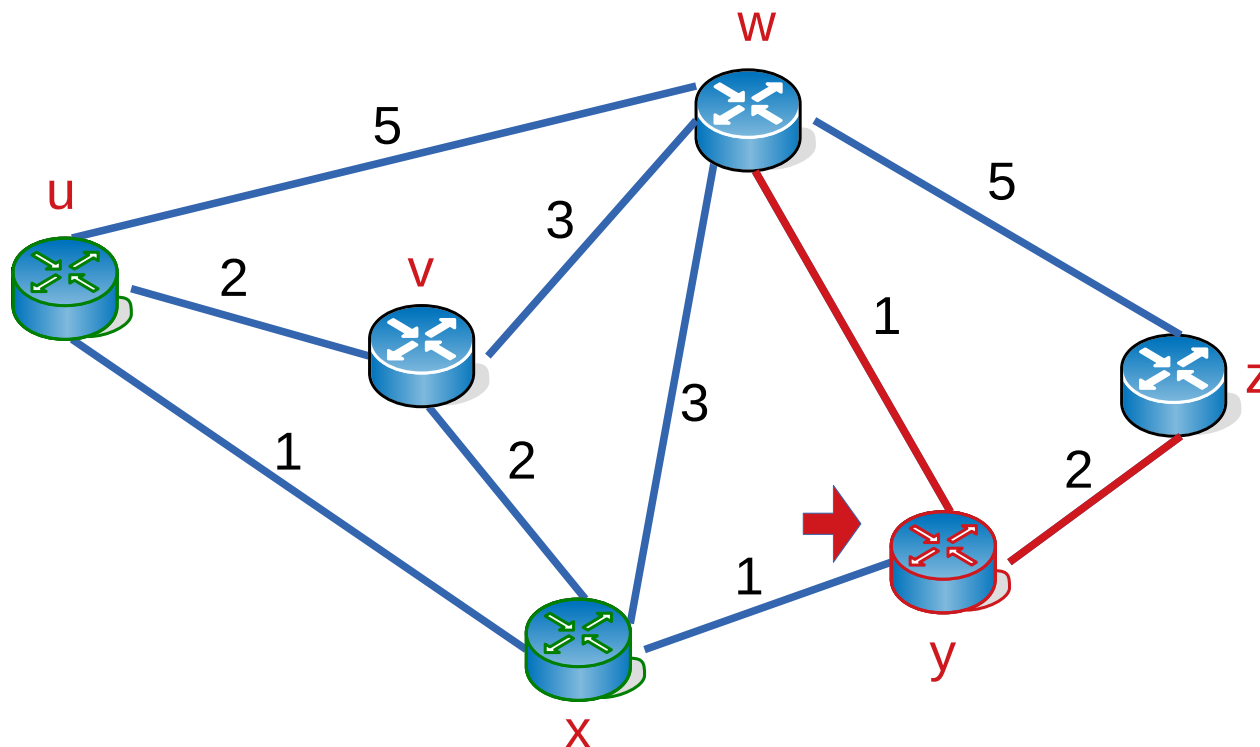
Dijkstra - Nodo origen: u

Paso	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y), p(y)	D(z), p(z)
0	u	2, u	5, u	1, u	∞	∞
1	ux	2, u	4, x		2, x	∞
2	uxy	2, u	3, y			4, y



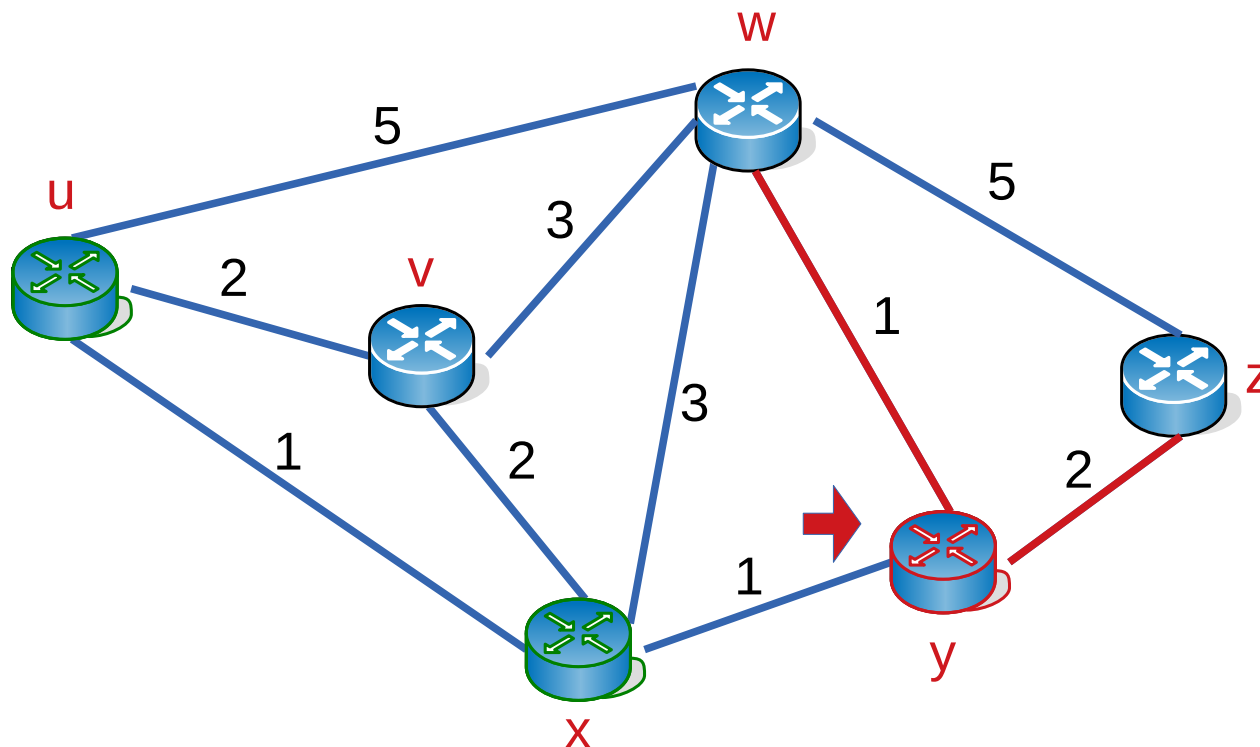
Dijkstra - Nodo origen: u

Paso	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y), p(y)	D(z), p(z)
0	u	2, u	5, u	1, u	∞	∞
1	ux	2, u	4, x		2, x	∞
2	uxy	2, u	3, y			4, y



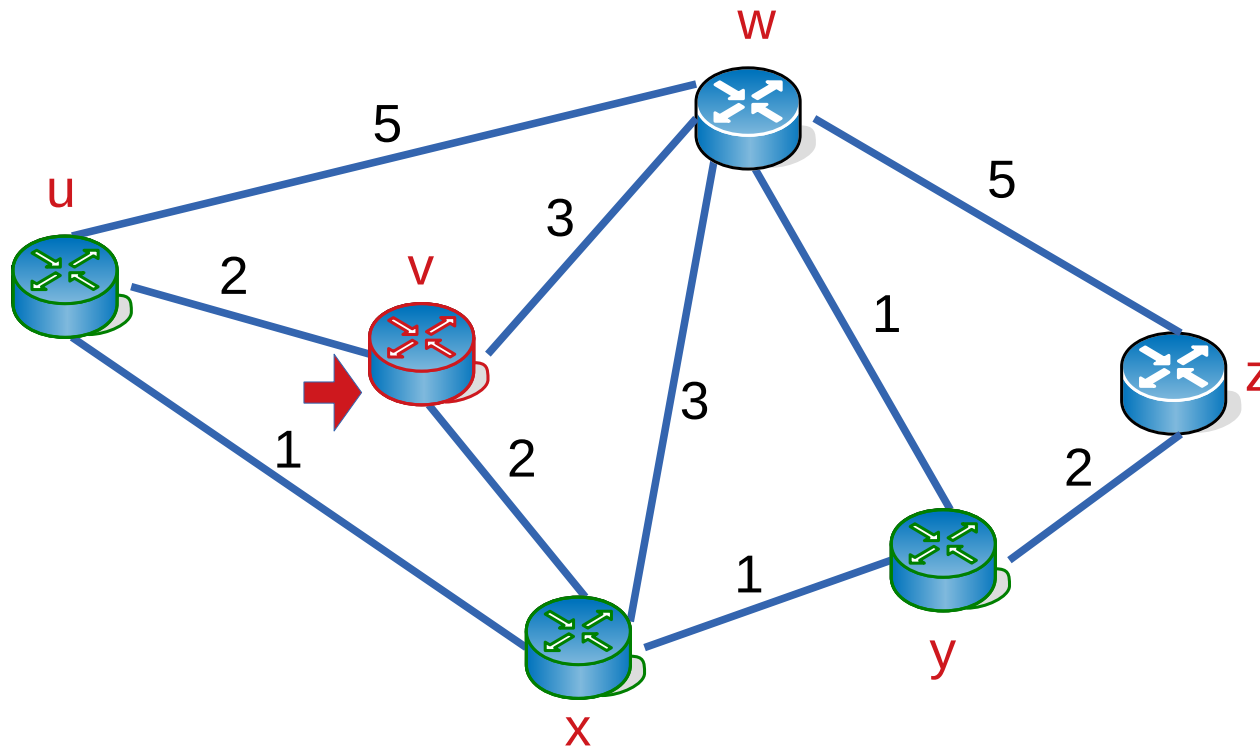
Dijkstra - Nodo origen: u

Paso	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y), p(y)	D(z), p(z)
0	u	2, u	5, u	1, u	∞	∞
1	ux	2, u	4, x		2, x	∞
2	uxy	2, u	3, y			4, y



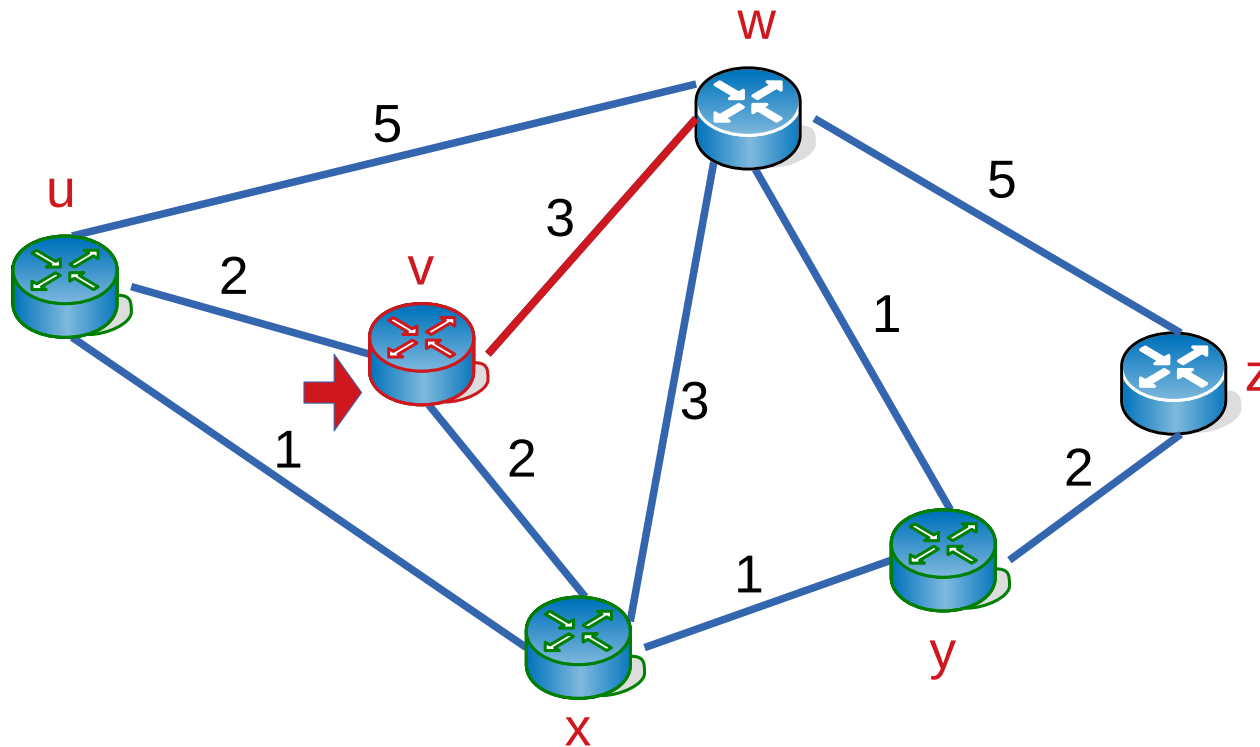
Dijkstra - Nodo origen: u

Paso	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y), p(y)	D(z), p(z)
0	u	2, u	5, u	1, u	∞	∞
1	ux	2, u	4, x		2, x	∞
2	uxy	2, u	3, y			4, y
3	uxyv		3, y			4, y



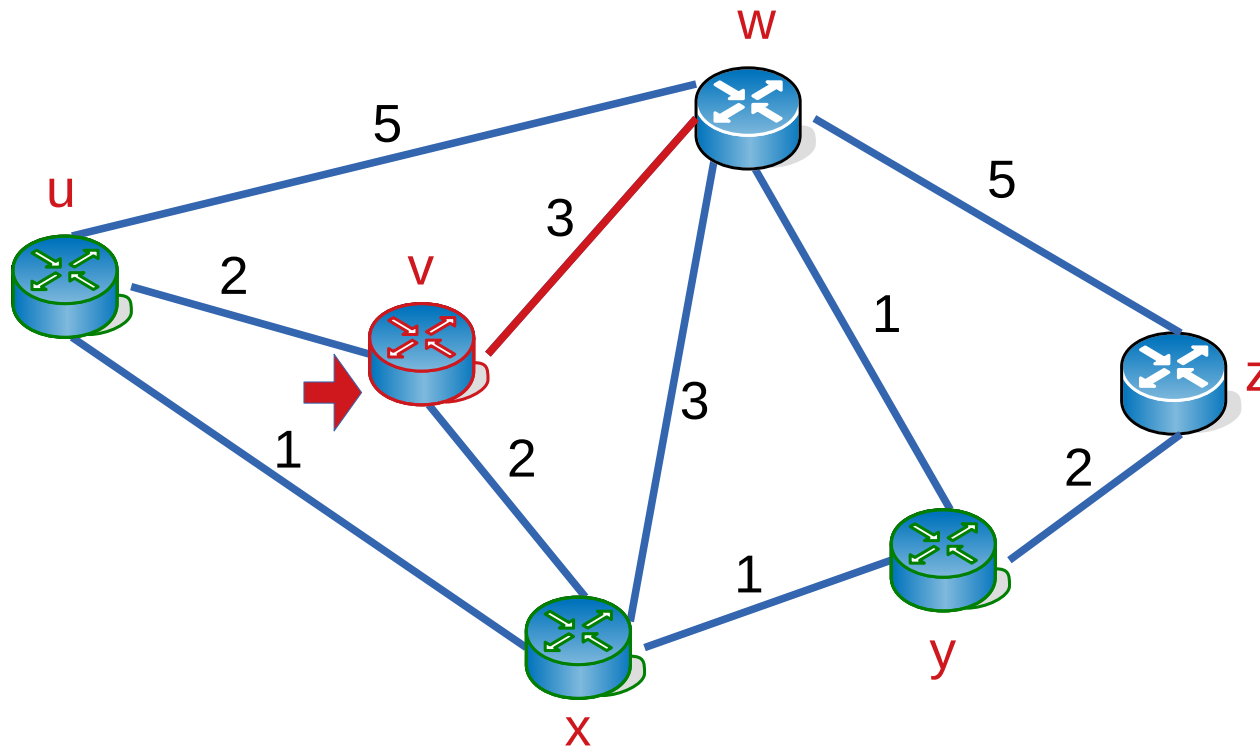
Dijkstra - Nodo origen: u

Paso	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y), p(y)	D(z), p(z)
0	u	2, u	5, u	1, u	∞	∞
1	ux	2, u	4, x		2, x	∞
2	uxy	2, u	3, y			4, y
3	uxyv		3, y			4, y



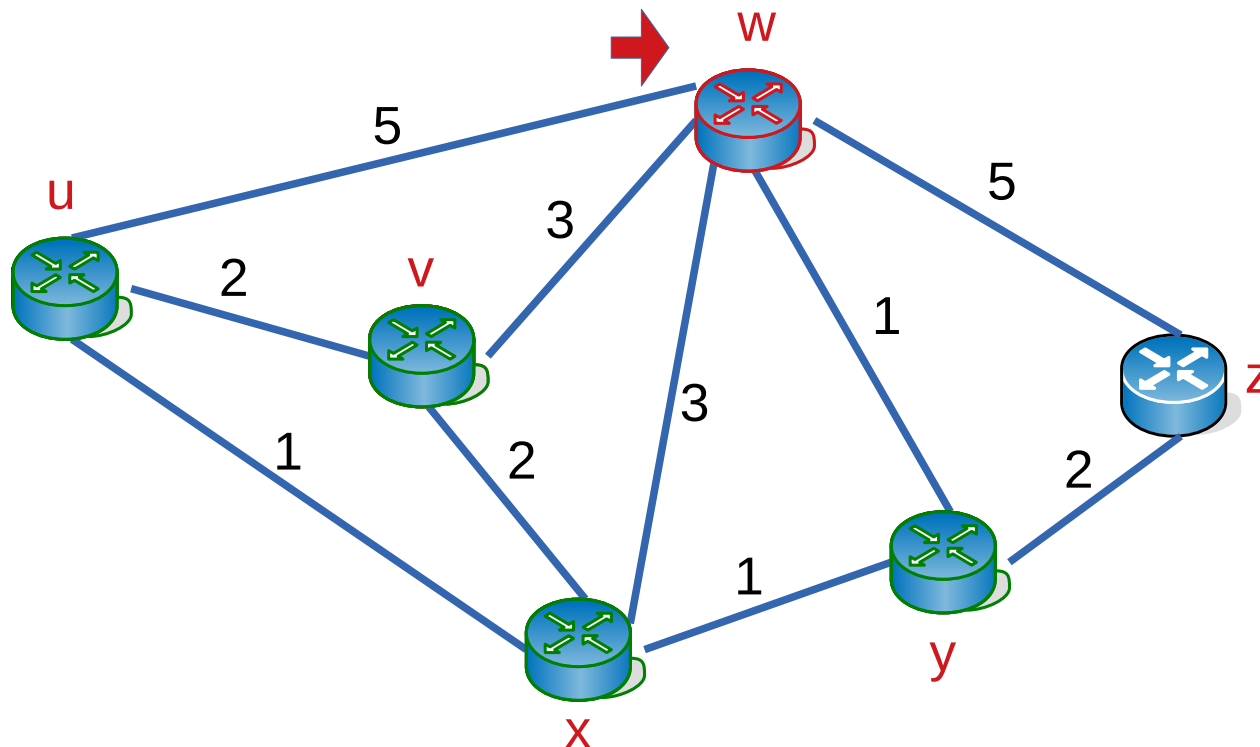
Dijkstra - Nodo origen: u

Paso	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y), p(y)	D(z), p(z)
0	u	2, u	5, u	1, u	∞	∞
1	ux	2, u	4, x		2, x	∞
2	uxy	2, u	3, y			4, y
3	uxyv		3, y			4, y



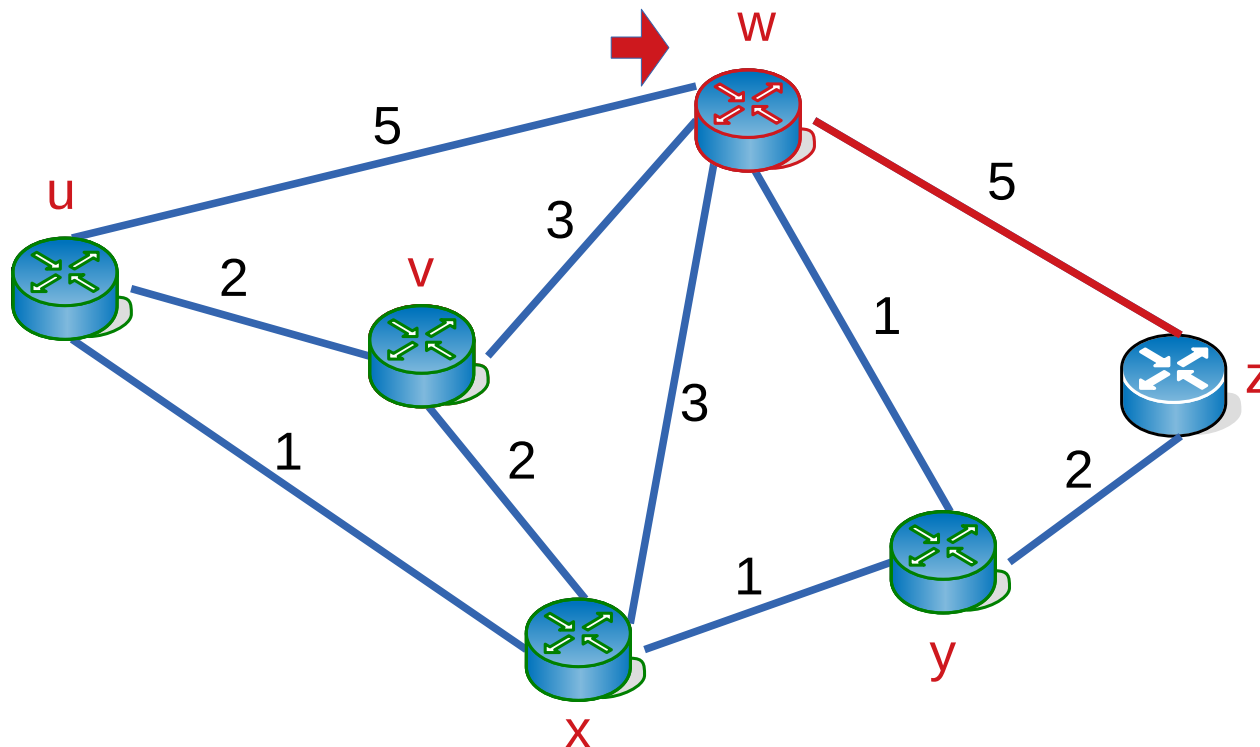
Dijkstra - Nodo origen: u

Paso	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y), p(y)	D(z), p(z)
0	u	2, u	5, u	1, u	∞	∞
1	ux	2, u	4, x		2, x	∞
2	uxy	2, u	3, y			4, y
3	uxyv		3, y			4, y
4	uxyvw					4, y



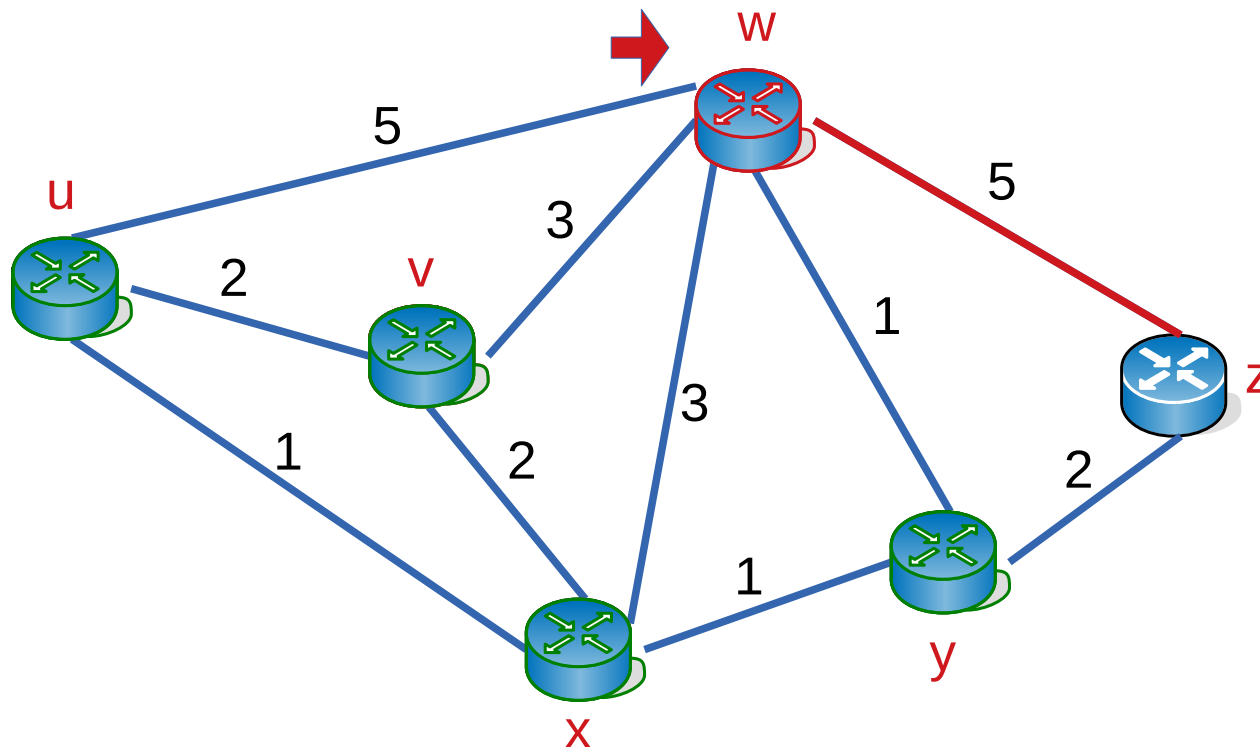
Dijkstra - Nodo origen: u

Paso	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y), p(y)	D(z), p(z)
0	u	2, u	5, u	1, u	∞	∞
1	ux	2, u	4, x		2, x	∞
2	uxy	2, u	3, y			4, y
3	uxyv		3, y			4, y
4	uxyvw					4, y



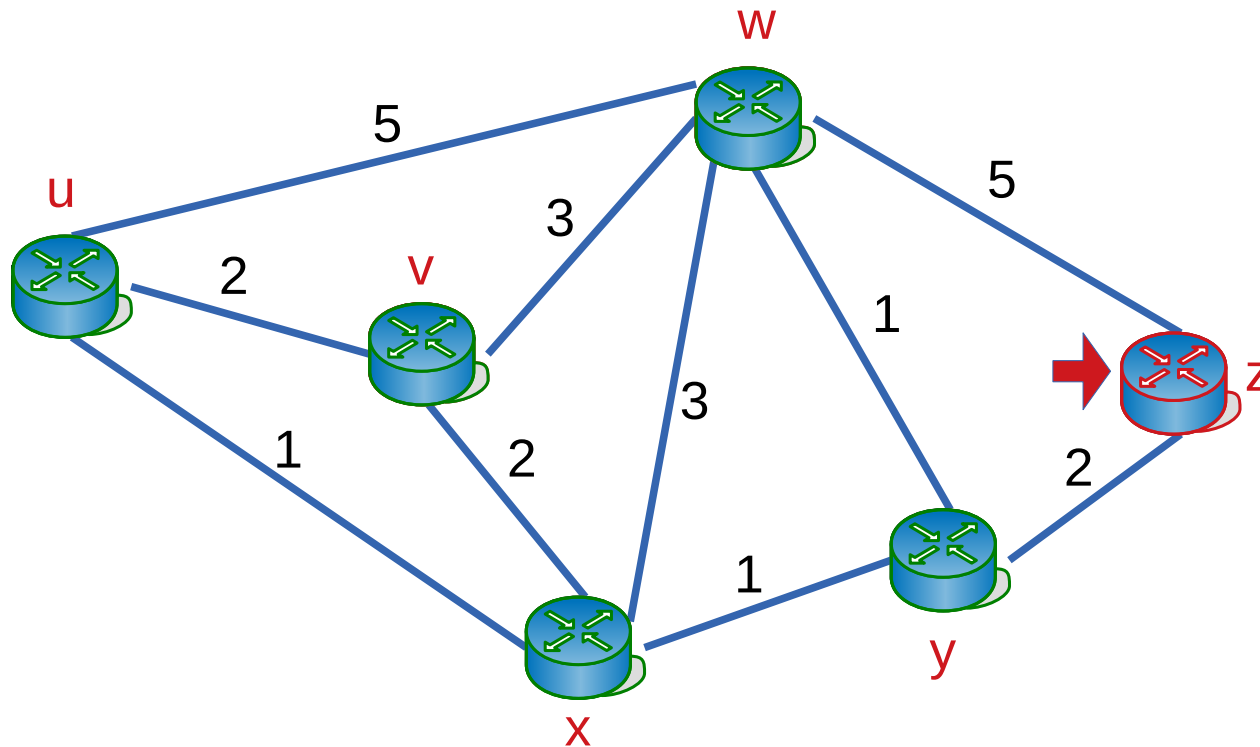
Dijkstra - Nodo origen: u

Paso	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y), p(y)	D(z), p(z)
0	u	2, u	5, u	1, u	∞	∞
1	ux	2, u	4, x		2, x	∞
2	uxy	2, u	3, y			4, y
3	uxyv		3, y			4, y
4	uxyvw					4, y



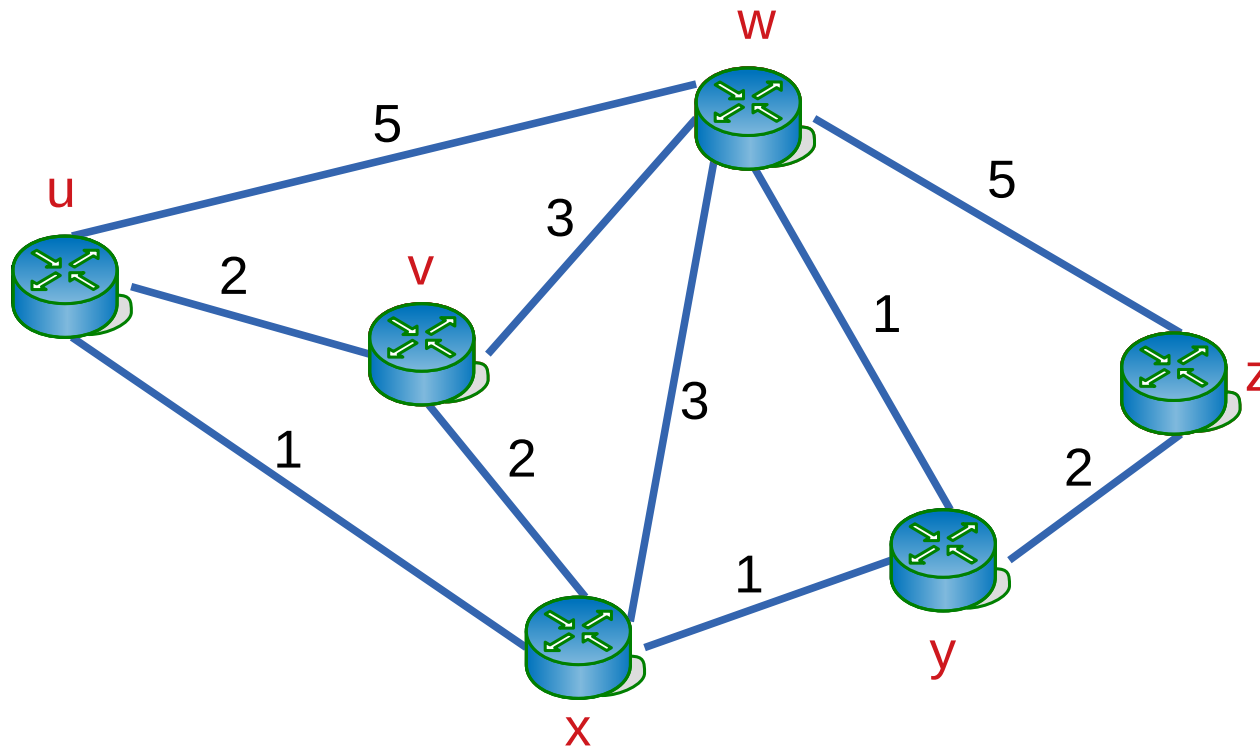
Dijkstra - Nodo origen: u

Paso	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y), p(y)	D(z), p(z)
0	u	2, u	5, u	1, u	∞	∞
1	ux	2, u	4, x		2, x	∞
2	uxy	2, u	3, y			4, y
3	uxyv		3, y			4, y
4	uxyvw					4, y
5	uxyvwz					



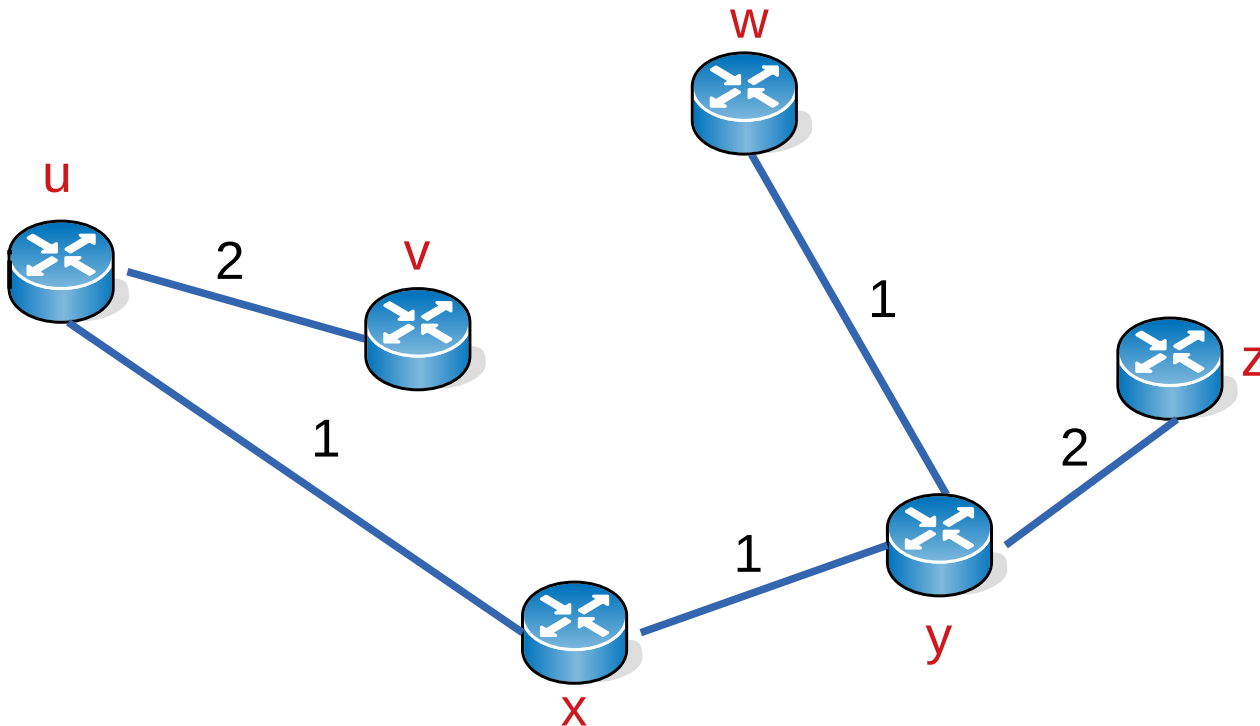
Dijkstra - Nodo origen: u

Paso	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y), p(y)	D(z), p(z)
0	u	2, u	5, u	1, u	∞	∞
1	ux	2, u	4, x		2, x	∞
2	uxy	2, u	3, y			4, y
3	uxyv		3, y			4, y
4	uxyvw					4, y
5	uxyvwz					



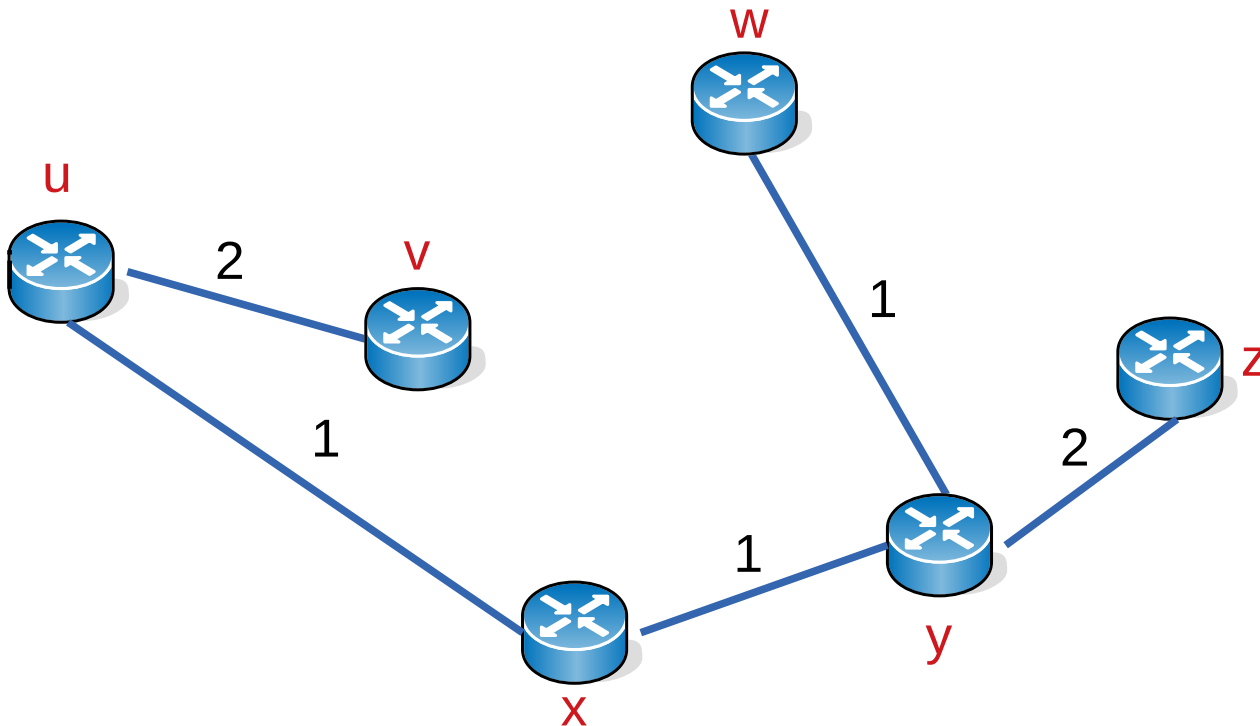
Arbol de mejores caminos y tabla de forwarding de u

Paso	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y), p(y)	D(z), p(z)
0	u	2, u	5, u	1, u	∞	∞
1	ux	2, u	4, x		2, x	∞
2	uxy	2, u	3, y			4, y
3	uxyv		3, y			4, y
4	uxyvw					4, y
5	uxyvwz					



Arbol de mejores caminos y tabla de forwarding de u

Paso	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y), p(y)	D(z), p(z)
0	u	2, u	5, u	1, u	∞	∞
1	ux	2, u	4, x		2, x	∞
2	uxy	2, u	3, y			4, y
3	uxyv		3, y			4, y
4	uxyvw					4, y
5	uxyvwz					



Destino	Enlace
v	(u, v)
w	(u, x)
x	(u, x)
y	(u, x)
z	(u, x)

Dijkstra: pseudo código, nodo u

- Inicialización

$$N' = \{ u \}$$

para todos los nodos v

si v es vecino de u

entonces $D(v) = c(u,v)$

sino $D(v) = \text{infinito}$

- Loop hasta que $N' = N$

encontrar w no incluido en N' tal que $D(w)$ sea mínima

agregar w a N'

actualizar $D(v)$ para todo vecino v de w no incluido en N' :

$$D(v) = \min\{ D(v), D(w) + c(w,v) \}$$

Algoritmos de vector distancia

- Distance Vector routing algorithms (DV)
- Algoritmo iterativo, asíncrono y distribuido
- Cada nodo recibe información de sus nodos vecinos directamente conectados, hace cálculos y redistribuye esa información a sus vecinos
- Se basa en la ecuación de Bellman-Ford:

$$d_x(y) = \min_j \{ c(x,j) + d_j(y) \}$$

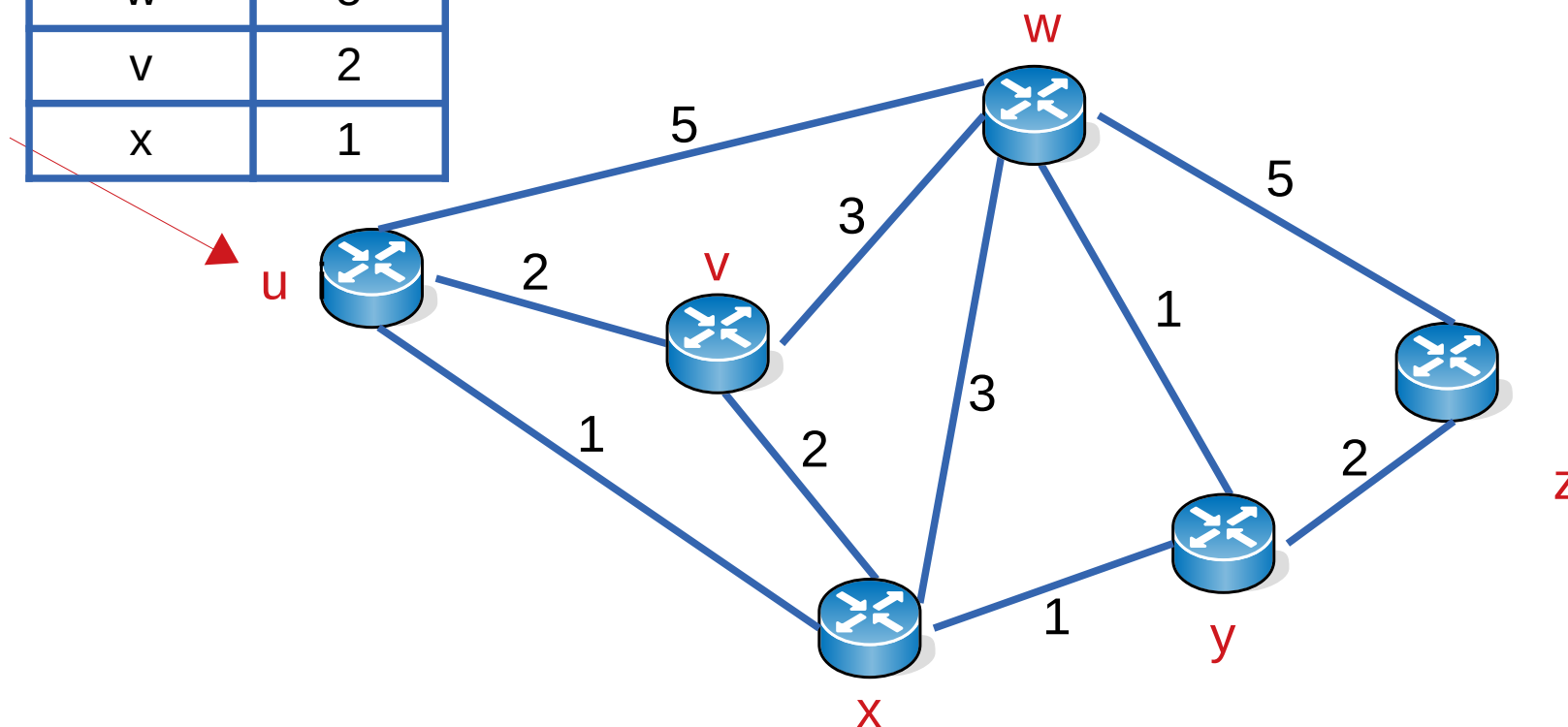
- j son los vecinos directamente conectados a x
 - La distancia de x a y pasando por j resulta de sumar la distancia de x a j más la distancia de j a y
 - El nodo j^* que minimice la distancia, será el siguiente nodo en el mejor camino hacia y
- Cada nodo x , recibe un vector de cada vecino j con sus distancias $D_j(y)$ para todo y
- Construye su propio vector

$$D_x(y) = \min_j \{ c(x,j) + D_j(y) \} \text{ para todo } y$$

Dj(y) para cada j vecino de u

- Inicialmente cada nodo arma su tabla con los vecinos directamente conectados

Tabla de u	
Destino	Costo
w	5
v	2
x	1

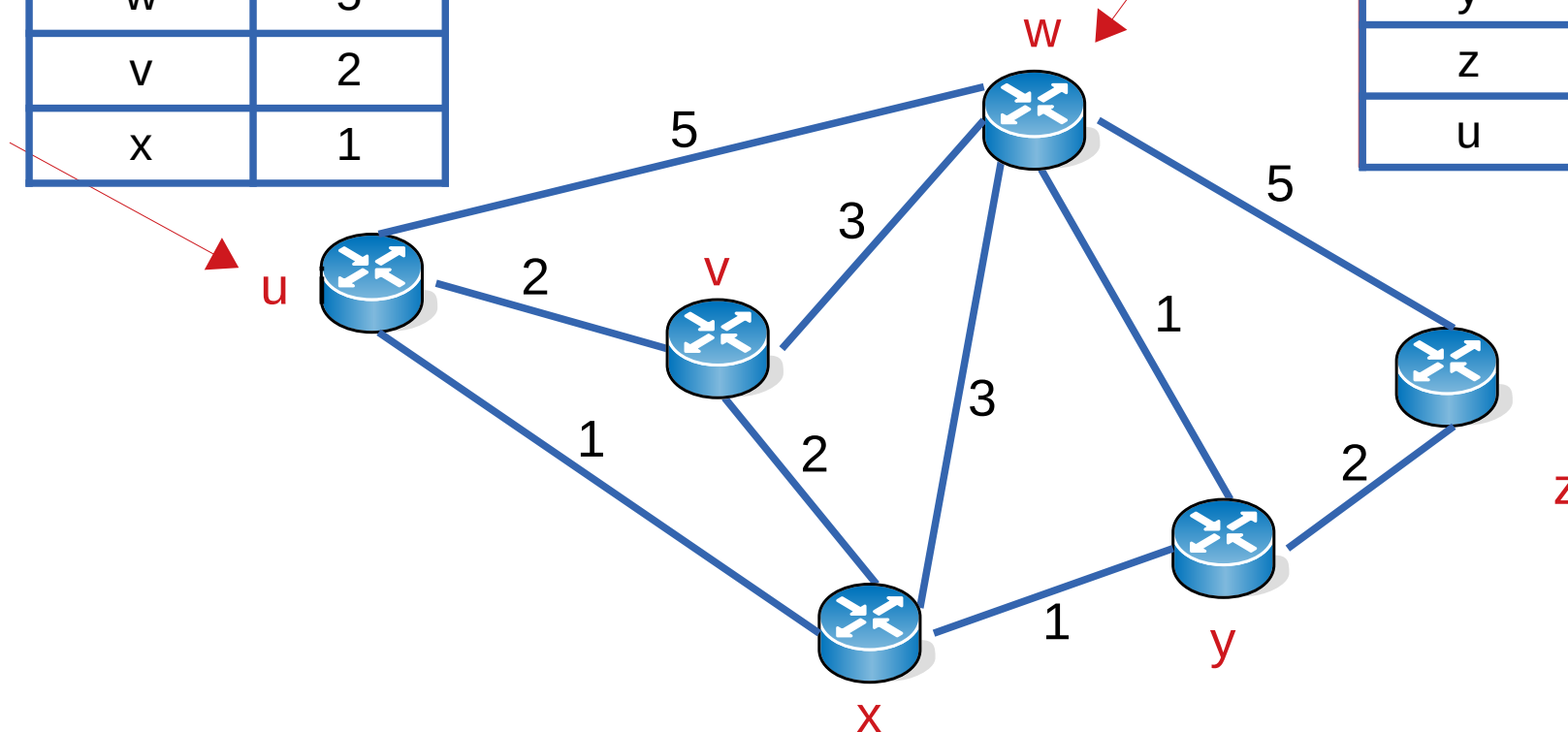


Dj(y) para cada j vecino de u

- Inicialmente cada nodo arma su tabla con los vecinos directamente conectados

Tabla de u	
Destino	Costo
w	5
v	2
x	1

Tabla de w	
Destino	Costo
v	3
x	3
y	1
z	5
u	5

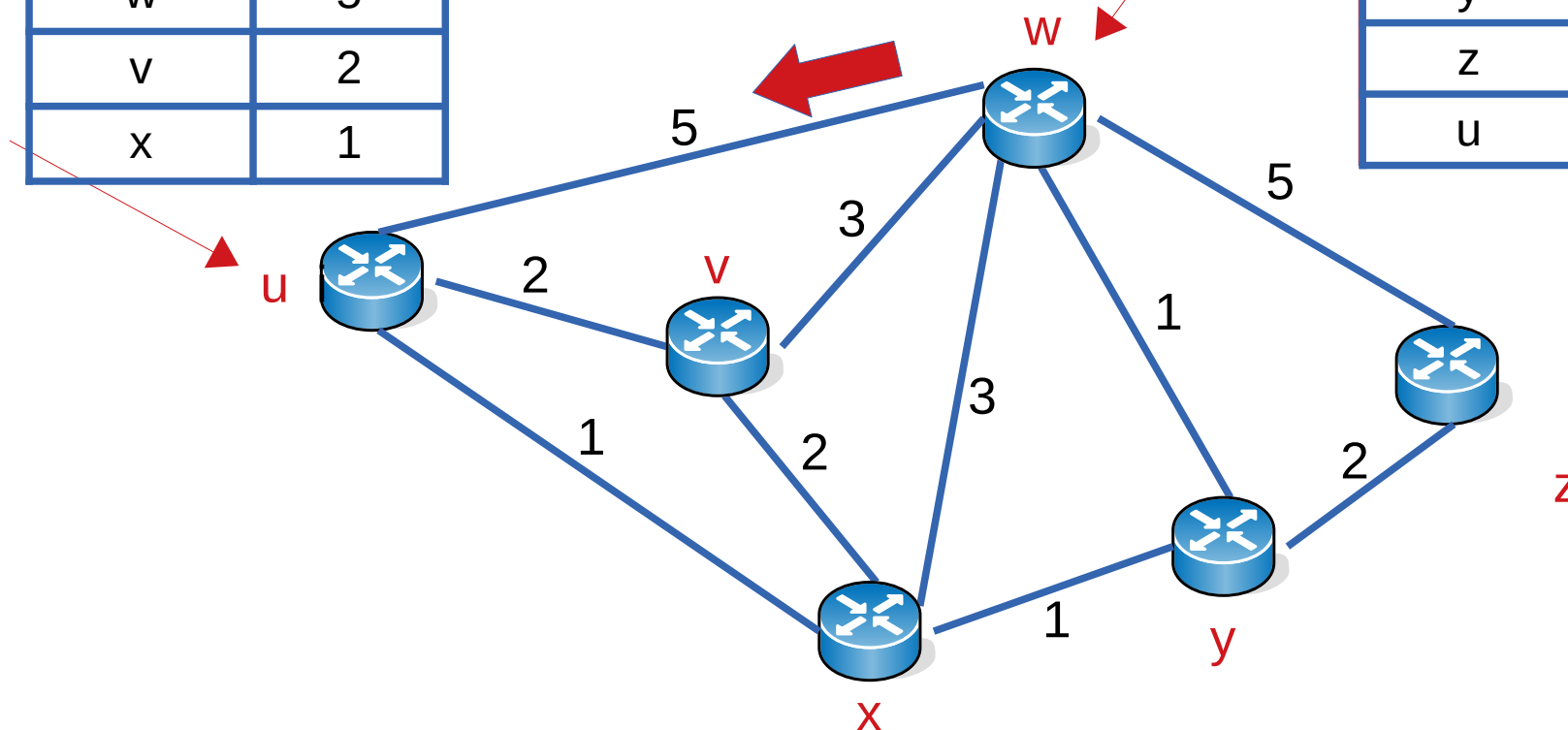


Dj(y) para cada j vecino de u

- Inicialmente cada nodo arma su tabla con los vecinos directamente conectados

Tabla de u	
Destino	Costo
w	5
v	2
x	1

Tabla de w	
Destino	Costo
v	3
x	3
y	1
z	5
u	5



Dj(y) para cada j vecino de u

- Inicialmente cada nodo arma su tabla con los vecinos directamente conectados

Tabla de u	
Destino	Costo
w	5
v	2
x	1

Tabla de w	
Destino	Costo
v	3
x	3
y	1
z	5
u	5

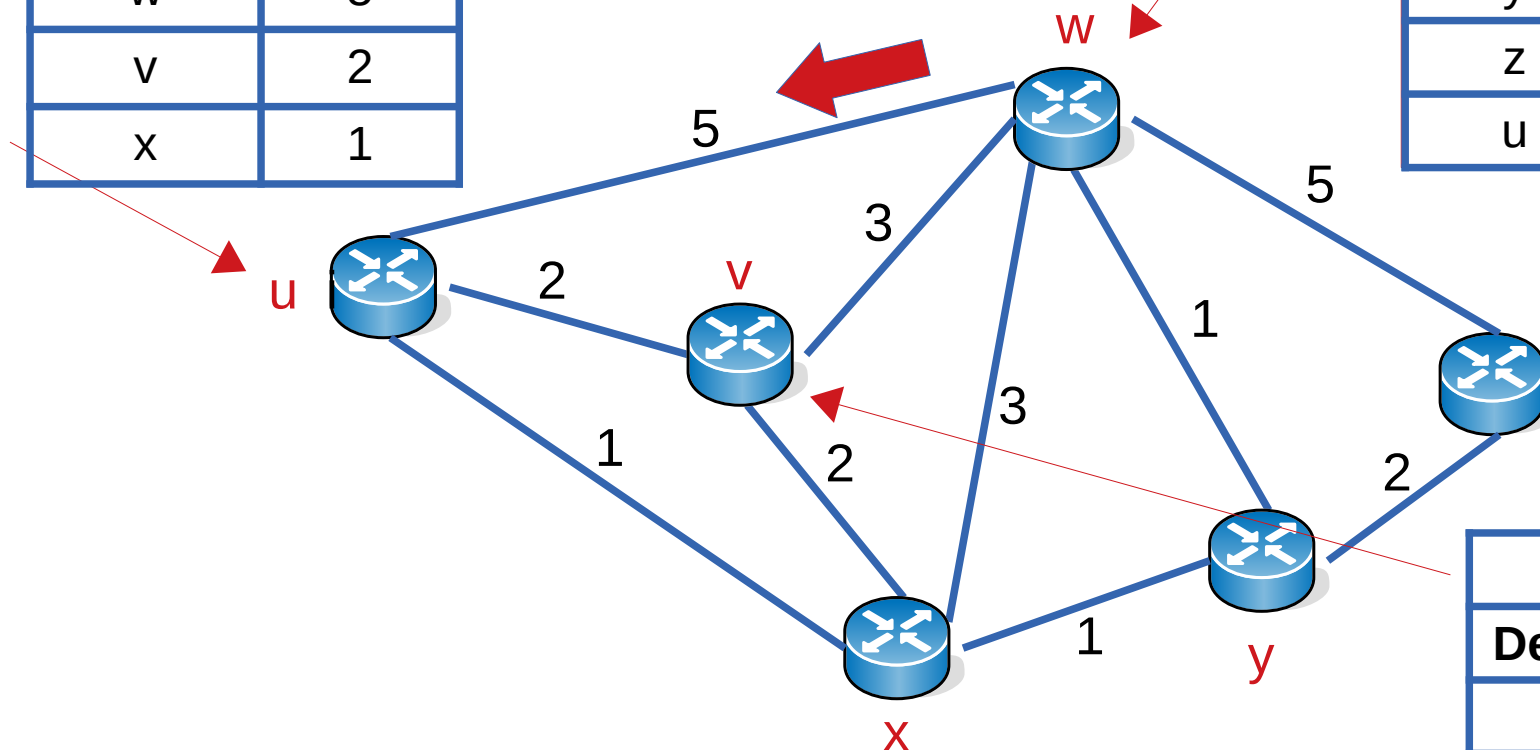


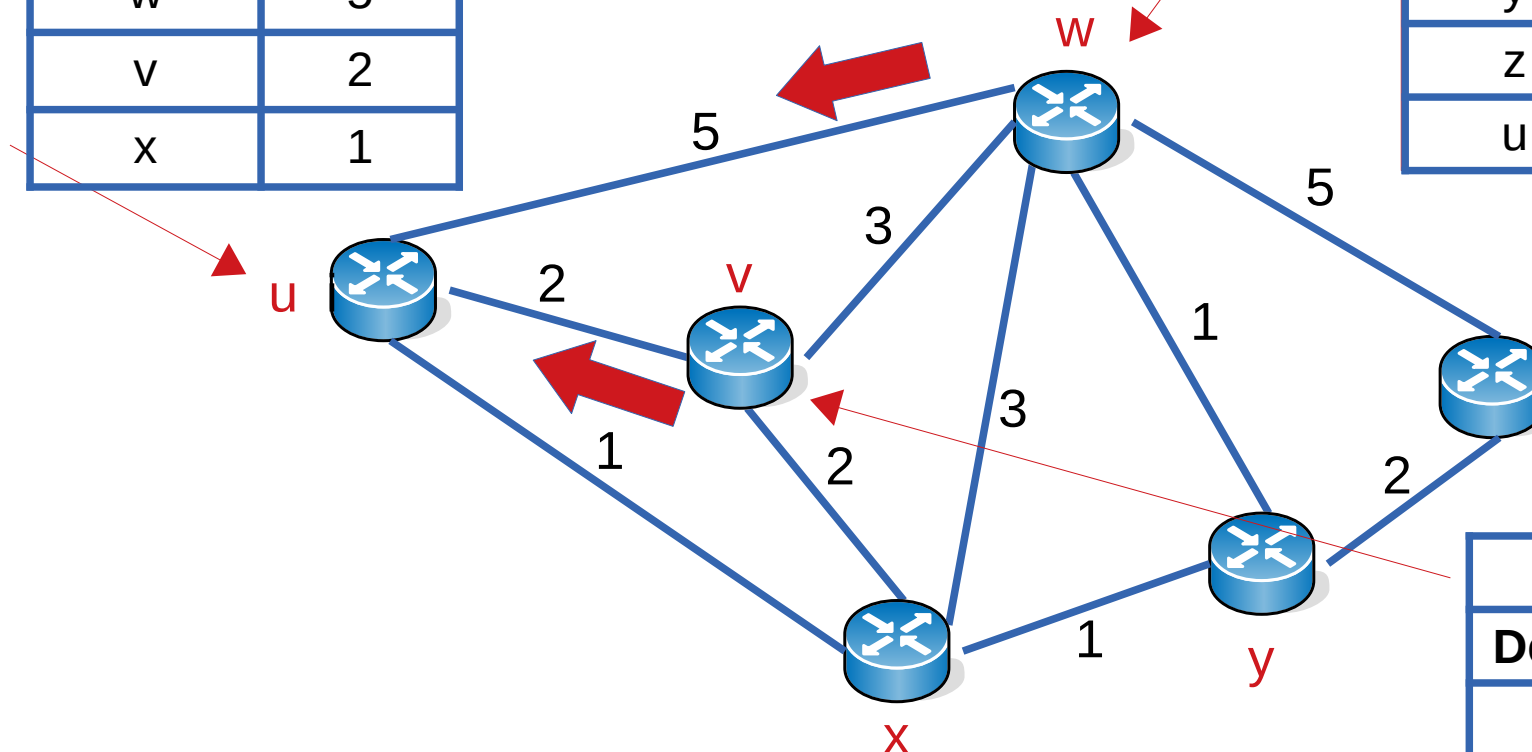
Tabla de v	
Destino	Costo
w	3
x	2
u	2

Dj(y) para cada j vecino de u

- Inicialmente cada nodo arma su tabla con los vecinos directamente conectados

Destino	Costo
w	5
v	2
x	1

Destino	Costo
v	3
x	3
y	1
z	5
u	5



Destino	Costo
w	3
x	2
u	2

Dj(y) para cada j vecino de u

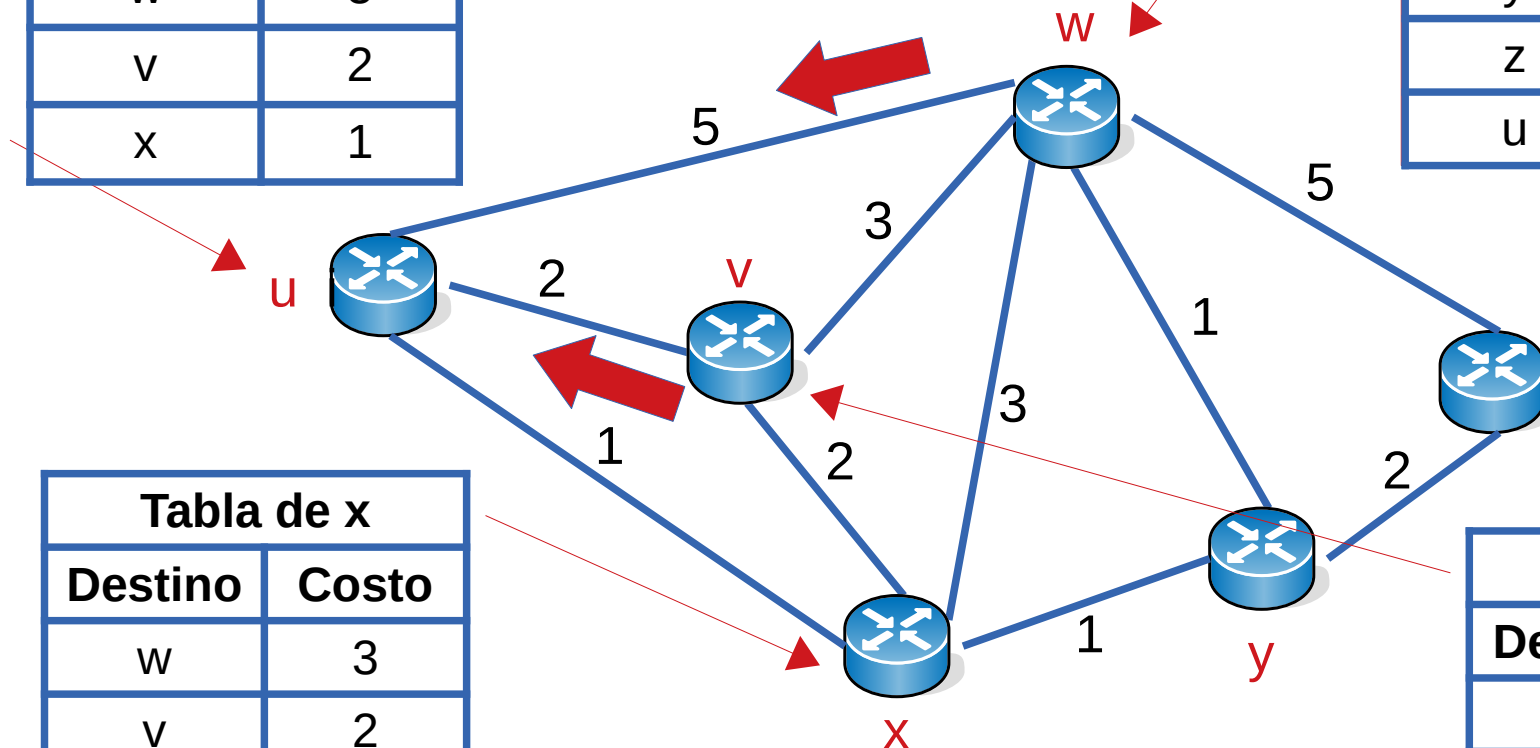
- Inicialmente cada nodo arma su tabla con los vecinos directamente conectados

Destino	Costo
w	5
v	2
x	1

Destino	Costo
v	3
x	3
y	1
z	5
u	5

Destino	Costo
w	3
v	2
y	1
u	1

Destino	Costo
w	3
x	2
u	2



Dj(y) para cada j vecino de u

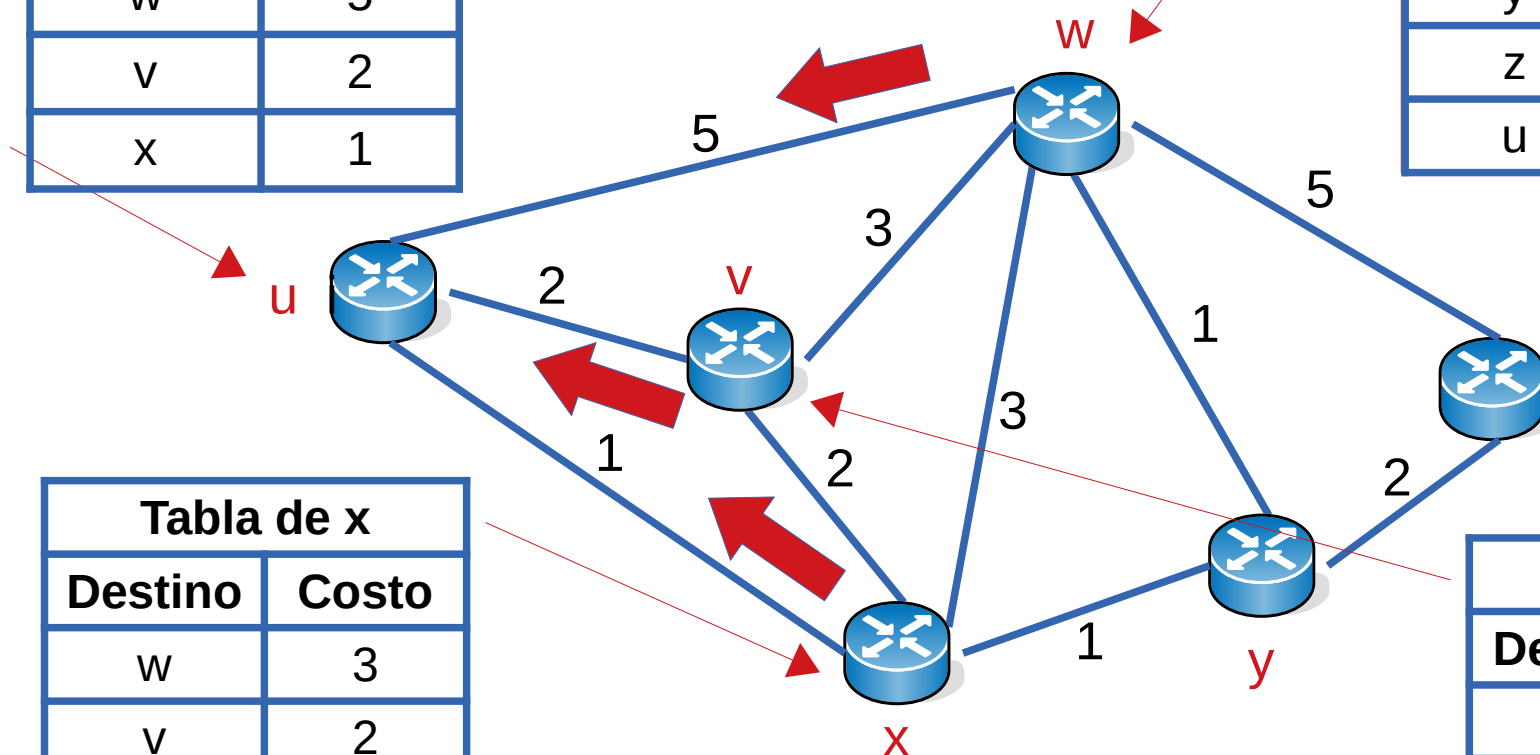
- Inicialmente cada nodo arma su tabla con los vecinos directamente conectados

Destino	Costo
w	5
v	2
x	1

Destino	Costo
v	3
x	3
y	1
z	5
u	5

Destino	Costo
w	3
v	2
y	1
u	1

Destino	Costo
w	3
x	2
u	2



¿ Cómo calcula u su $D_u(y)$?

- Tabla inicial de u

Tabla de u	
Destino	Costo
w	5
v	2
x	1

Tabla de w	
Destino	Costo
v	3
x	3
y	1
z	5
u	5

- Nueva tabla de u en base a la información recibida de w

Destino	Costo inicial	Costo por w	Decisión mínimo	Hacia (j^*)
w	5	∞	5	w
v	2	$5 + 3$	2	v
x	1	$5 + 3$	1	x
y	∞	$5 + 1$	6	w
z	∞	$5 + 5$	10	w
u	0	$5 + 5$	0	-

¿ Cómo calcula u su $D_u(y)$?

- Tabla inicial de u

Tabla de u	
Destino	Costo
w	5
v	2
x	1

Tabla de w	
Destino	Costo
v	3
x	3
y	1
z	5
u	5

- Nueva tabla de u en base a la información recibida de w

Destino	Costo inicial	Costo por w	Decisión mínimo	Hacia (j^*)
w	5	∞	5	w
v	2	$5 + 3$	2	v
x	1	$5 + 3$	1	x
y	∞	$5 + 1$	6	w
z	∞	$5 + 5$	10	w
u	0	$5 + 5$	0	-

¿ Cómo calcula u su $D_u(y)$?

- Tabla inicial de u

Tabla de u	
Destino	Costo
w	5
v	2
x	1

Tabla de w	
Destino	Costo
v	3
x	3
y	1
z	5
u	5

- Nueva tabla de u en base a la información recibida de w

Destino	Costo inicial	Costo por w	Decisión mínimo	Hacia (j^*)
w	5	∞	5	w
v	2	$5 + 3$	2	v
x	1	$5 + 3$	1	x
y	∞	$5 + 1$	6	w
z	∞	$5 + 5$	10	w
u	0	$5 + 5$	0	-

¿ Cómo calcula u su $D_u(y)$?

- Ahora recibe la información de v

Tabla de u		
Destino	Costo anterior	Hacia (j^*)
w	5	w
v	2	v
x	1	x
y	6	w
z	10	w
u	0	-

Tabla de v	
Destino	Costo
w	3
x	2
u	2

- Nueva tabla de u en base a la información recibida de v

Destino	Costo anterior	Costo por v	Decisión mínimo	Hacia (j^*)
w	5	$2 + 3$	5	w
v	2	∞	2	v
x	1	$2 + 2$	1	x
y	6	∞	6	w
z	10	∞	10	w
u	0	$2 + 2$	0	-

¿ Cómo calcula u su $D_u(y)$?

- Ahora recibe la información de x

Tabla de u		
Destino	Costo anterior	Hacia (j^*)
w	5	w
v	2	v
x	1	x
y	6	w
z	10	w
u	0	-

Tabla de x	
Destino	Costo
w	3
v	2
y	1
u	1

- Nueva tabla de u en base a la información recibida de x

Destino	Costo anterior	Costo por x	Decisión mínimo	Hacia (j^*)
w	5	$1 + 3$	4	x
v	2	$1 + 2$	2	v
x	1	∞	1	x
y	6	$1 + 1$	2	x
z	10	∞	10	w
u	0	$1 + 1$	0	-

¿ Cómo calcula u su $D_u(y)$?

- Ahora recibe la información de x

Tabla de u		
Destino	Costo anterior	Hacia (j^*)
w	5	w
v	2	v
x	1	x
y	6	w
z	10	w
u	0	-

Tabla de x	
Destino	Costo
w	3
v	2
y	1
u	1

- Nueva tabla de u en base a la información recibida de x

Destino	Costo anterior	Costo por x	Decisión mínimo	Hacia (j^*)
w	5	$1 + 3$	4	x
v	2	$1 + 2$	2	v
x	1	∞	1	x
y	6	$1 + 1$	2	x
z	10	∞	10	w
u	0	$1 + 1$	0	-

¿ Cómo calcula u su $D_u(y)$?

- Ahora recibe la información de x

Tabla de u		
Destino	Costo anterior	Hacia (j^*)
w	5	w
v	2	v
x	1	x
y	6	w
z	10	w
u	0	-

Tabla de x	
Destino	Costo
w	3
v	2
y	1
u	1

- Nueva tabla de u en base a la información recibida de x

Destino	Costo anterior	Costo por x	Decisión mínimo	Hacia (j^*)
w	5	$1 + 3$	4	x
v	2	$1 + 2$	2	v
x	1	∞	1	x
y	6	$1 + 1$	2	x
z	10	∞	10	w
u	0	$1 + 1$	0	-

Vector distancia: pseudo código nodo x

- Inicialización

para todos los destinos y en N :

si v es vecino de u

entonces $D_x(v) = c(u,v)$

sino $D_x(v) = \text{infinito}$

para cada vecino w

enviar vector $D_x = \{ D_x(y) \text{ para todo } y \}$

- Loop para siempre

esperar a:

recibir vectores de los vecinos w

o ver si costo a algún vecino cambia

para cada y en N :

$$D_x(y) = \min_v \{ c(x,v) + D_v(y) \}$$

si $D_x(y)$ cambió para algún destino y

para cada vecino w

enviar vector $D_x = \{ D_x(y) \text{ para todo } y \}$

Protocolos de ruteo dinámico

- En la arquitectura tradicional los algoritmos vistos (vector distancia, DV o estado del enlace, LS) necesitan que los nodos **intercambien información** entre si
 - En la arquitectura SDN, el controlador conoce la topología de la red mediante su vínculo con cada conmutador SDN
- Para intercambiar esa información (vector de distancias o distancias a los vecinos respectivamente) se utilizan **protocolos específicos**
- Esos protocolos pueden usar UDP o viajar directamente sobre IP o sobre Ethernet
- Algunos ejemplos usados en Internet
 - **RIP** (Routing Information Protocol): Vector distancia
 - **OSPF** (Open Shortest Path First): Estado del enlace
 - **ISIS** (Intermediate System to intermediate System): Estado del enlace
 - **BGP** (Border Gateway Protocol): Vector distancia (vector de caminos)
- En este curso no vamos a estudiar estos protocolos (Redes de Datos 2)

Clasificación de algoritmos de ruteo

- En la realidad la red global no es un conjunto de nodos homogéneo en la que todos los nodos puedan intercambiar información de ruteo
 - Escala
 - Sería imposible que los cientos de millones de enrutadores que hay en Internet almacenaran y compartieran información de ruteo
 - Habría un altísimo tráfico de información de ruteo circulando por la red
 - Con la dinámica de cambio de estado de los enlaces los algoritmos nunca convergerían
 - Proveedores de servicio
 - Los proveedores de servicio ([ISP](#), Internet Service Providers) son dueños y administran sus enrutadores
 - Cada proveedor de servicio utilizará el algoritmo de ruteo que prefiera
 - Querrá ocultar información interna de su red a otros proveedores

Sistemas Autónomos

- Los enrutadores de la red se organizan en **Sistemas Autónomos** (ASs)
 - Un **AS** es un conjunto de enrutadores bajo control de una **misma administración**
- Un ISP puede dividir su red en varios ASs o tener un solo AS para toda su red
- Cada AS tiene un **número único** asignado por ICANN
- Los protocolos y algoritmos de ruteo que se usan dentro de un AS se llaman **intra-AS** o de **ruteo interno**
- Los protocolos y algoritmos de ruteo que intercambian información entre ASs se llaman **inter-AS** o de **ruteo externo**
 - Permiten aplicar políticas para decidir qué quiero informar a los otros ASs o qué quiero aceptar de lo que otros ASs me informen
 - El protocolo de ruteo externo usado en Internet es **BGP** (Border Gateway Protocol)

ICMP: Internet Control Message Protocol

- Protocolo para mensajes de control, principalmente para reportar errores (RFC 792)
- Los mensajes **ICMP** viajan como carga de paquetes IP
- Usos destacados:
 - Cuando un enrutador procesa un paquete decrementa el TTL en 1 y si llega a 0,
 - Descarta el paquete
 - Envía al originador un mensaje **ICMP “TTL expired”**
 - En el mensaje se indica la IP del enrutador que descartó el paquete
 - Cuando un enrutador no encuentra en su tabla de forwarding una ruta hacia la IP destino del paquete (caso en que no hay ruta por defecto),
 - Descarta el paquete
 - Envía al originador (IP origen del paquete) un mensaje **ICMP “destination network unreachable”**
 - En el mensaje se indica la IP del enrutador que descartó el paquete

ICMP: Internet Control Message Protocol

- Usos destacados:
 - Cuando un enrutador tiene un paquete destinado a una IP que no está disponible dentro de un rango de direcciones en una red de difusión,
 - Descarta el paquete
 - Envía al originador un mensaje ICMP “destination host unreachable”
 - En el mensaje se indica la IP del enrutador que descartó el paquete
 - Cuando hay puertos filtrados por temas de seguridad, el enrutador o firewall,
 - Descarta el segmento
 - Envía al originador un mensaje ICMP “destination port unreachable”
 - En el mensaje se indica la IP del equipo que descartó el paquete

Comandos que usan ICMP

- Comando “ping”
 - Saber si un equipo está activo en la red
 - Usa mensajes ICMP “echo request” y “echo reply”
- Comando “traceroute” o “tracert”
 - Permite descubrir la secuencia de enrutadores en el camino hacia un destino
 - Envía una secuencia de mensajes UDP a la IP del destino de interés y a un puerto destino (raro)
 - El sistema operativo Windows lo hace con ICMP
 - En cada envío el campo de TTL de IP se incrementa en 1
 - Los mensajes ICMP “TTL expired” se usan para determinar los enrutadores en el camino al destino
 - Como se usa un puerto destino raramente usado, cuando se llega a la IP de destino, el mensaje recibido será un ICMP “port unreachable”

Algunos tipos de mensajes ICMP

ICMP Type	ICMP Code	Descripción
0	0	Echo reply
3	0	Destination network unreachable
3	1	Destination host unreachable
3	2	Destination protocol unreachable
3	3	Destination port unreachable
4	0	Source Quench (control de congestión, no usado)
5	0	Redirect
8	0	Echo request
11	0	TTL expired
12	0	IP header bad

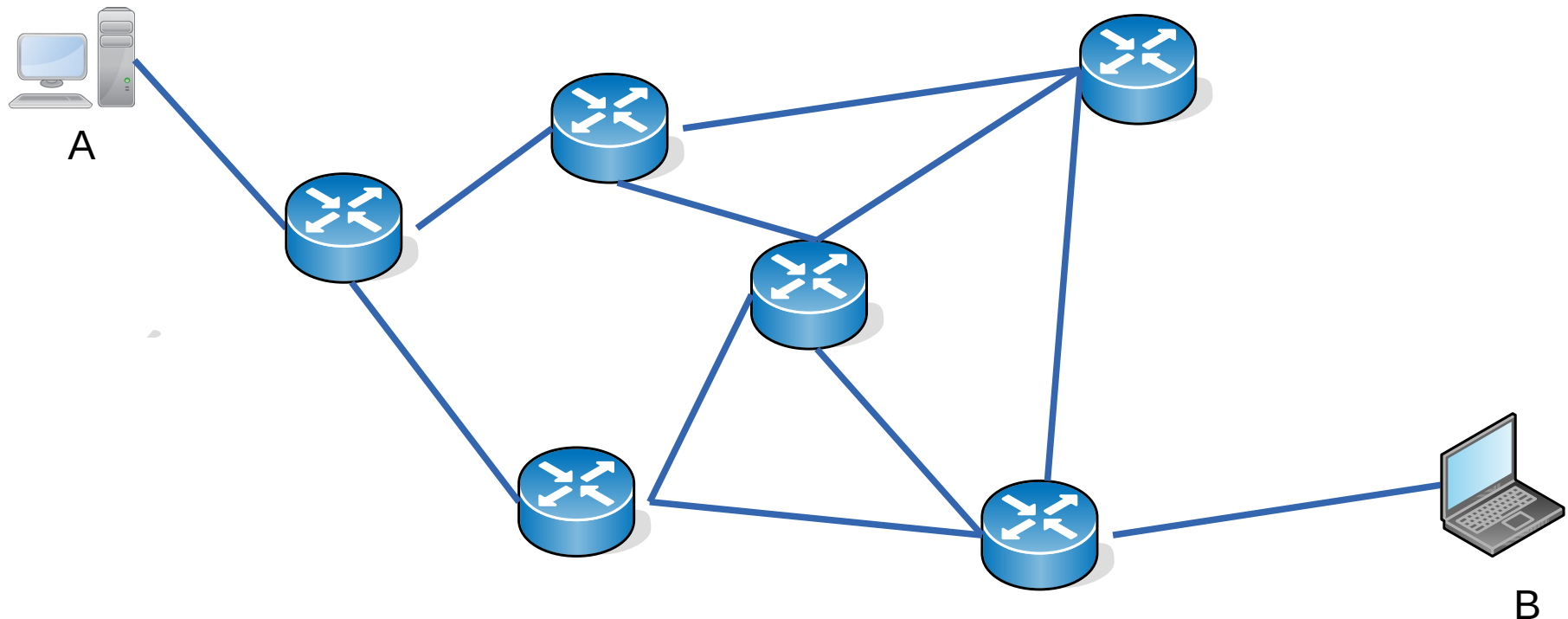
Redes de datos 1

Capa de red Circuitos Virtuales

Facultad de Ingeniería – Universidad de la República
Instituto de Ingeniería Eléctrica

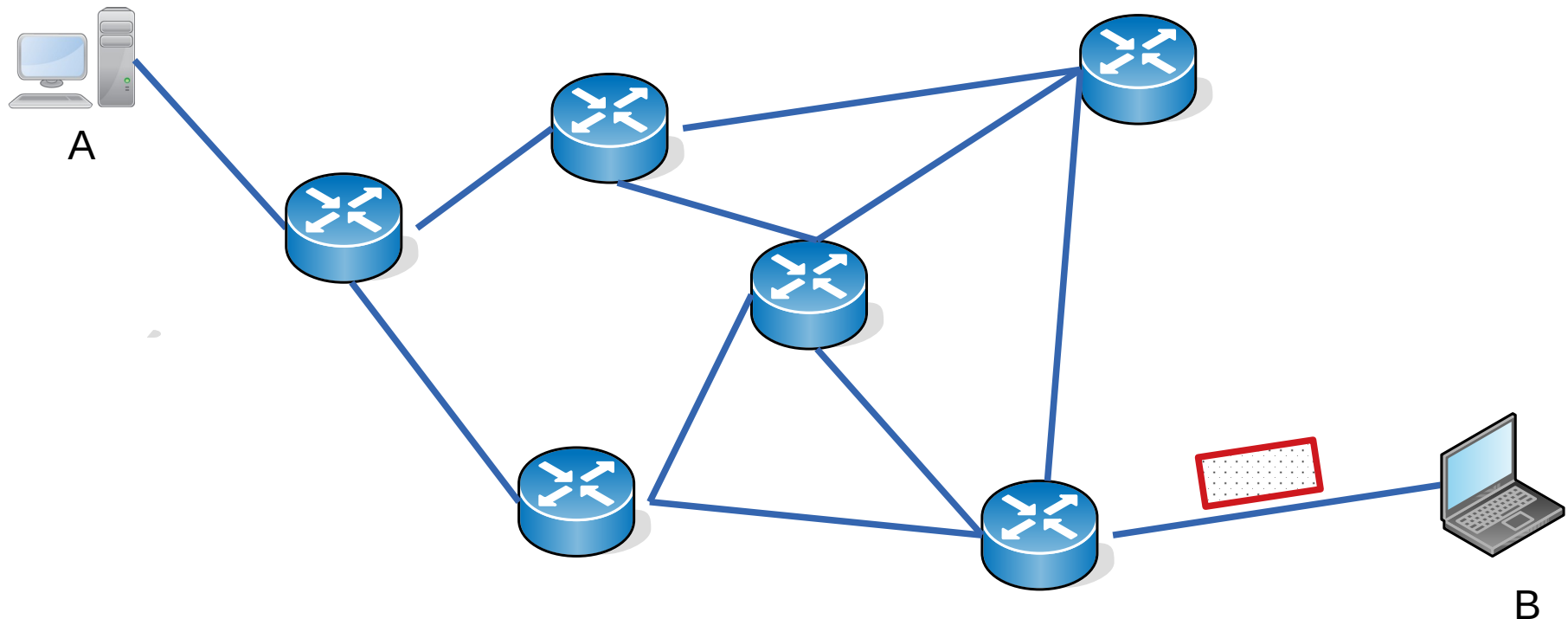
Redes de datagramas

- En la arquitectura de red de datagramas vista
 - Los paquetes se encaminan por separado en base a su dirección de destino y consultando la tabla de forwarding
 - El camino lo define la función de ruteo y podría cambiar en el curso de una comunicación



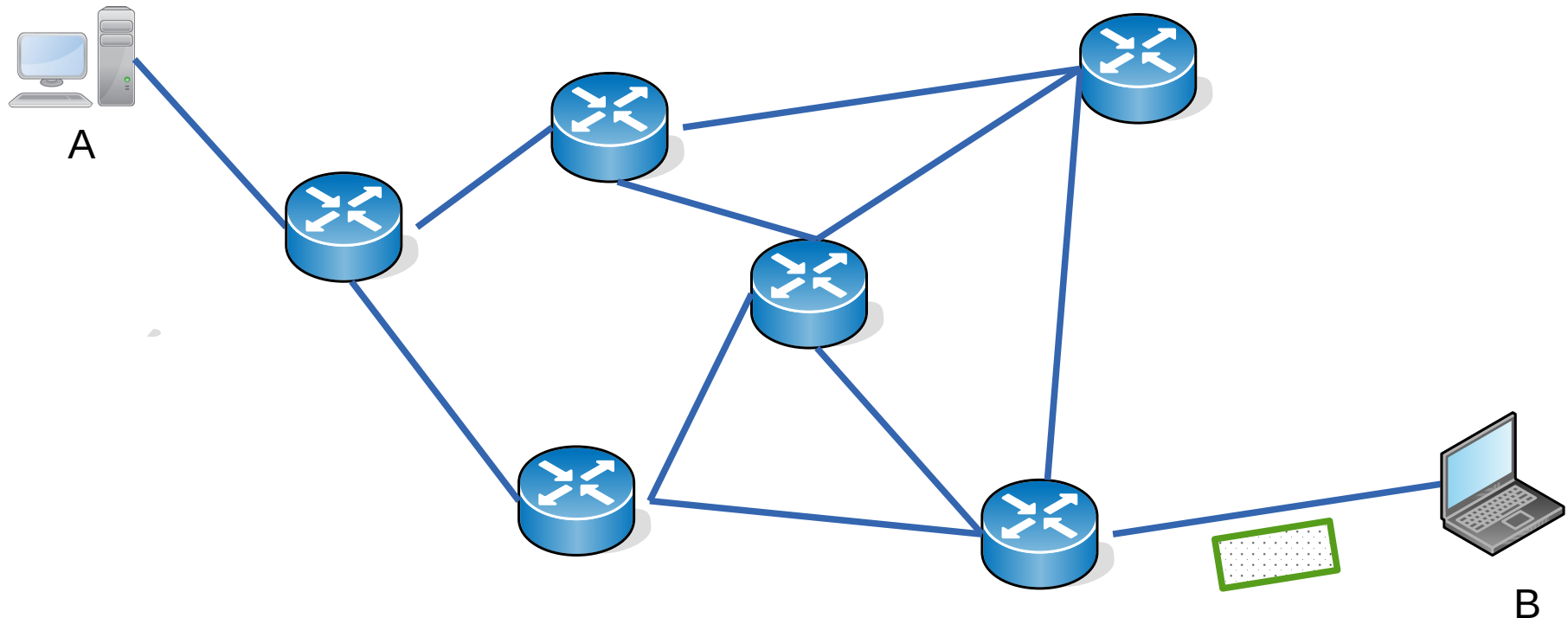
Redes de datagramas

- En la arquitectura de red de datagramas vista
 - Los paquetes se encaminan por separado en base a su dirección de destino y consultando la tabla de forwarding
 - El camino lo define la función de ruteo y podría cambiar en el curso de una comunicación



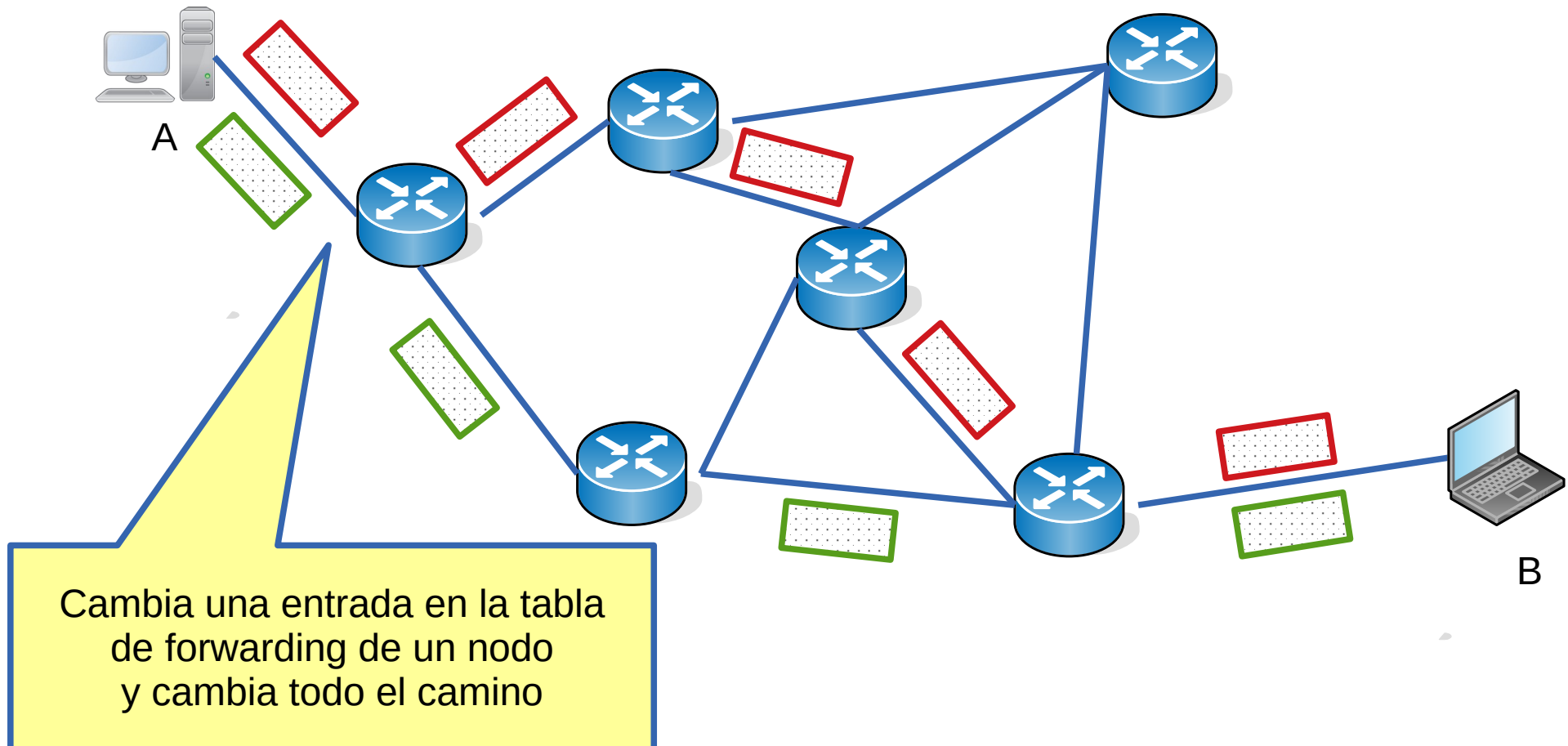
Redes de datagramas

- En la arquitectura de red de datagramas vista
 - Los paquetes se encaminan por separado en base a su dirección de destino y consultando la tabla de forwarding
 - El camino lo define la función de ruteo y podría cambiar en el curso de una comunicación



Redes de datagramas

- En la arquitectura de red de datagramas vista
 - Los paquetes se encaminan por separado en base a su dirección de destino y consultando la tabla de forwarding
 - El camino lo define la función de ruteo y podría cambiar en el curso de una comunicación



Limitaciones de la arquitectura de datagramas

- Algunas aplicaciones son más sensibles que otras a las pérdidas, retardos y variaciones de retardo (jitter)
- Una capa de red basada en encaminamiento de datagramas con criterio de “best effort” posee escasas herramientas para proveer **calidad de servicio** de extremo a extremo
 - No se puede hacer un **control de congestión** adecuado
 - Un escenario de congestión aumenta las pérdidas, el retardo y las variaciones de retardo
 - No se puede garantizar **calidad de servicio entre proveedores**
 - Es complicado implementar **ingeniería de tráfico**
 - Para que diferentes tipos de tráfico circulen por caminos diferentes
- En general las soluciones tradicionales apuntan a aumentar los recursos de la red de modo de minimizar los problemas de congestión, pero ese sobredimensionamiento es **caro**

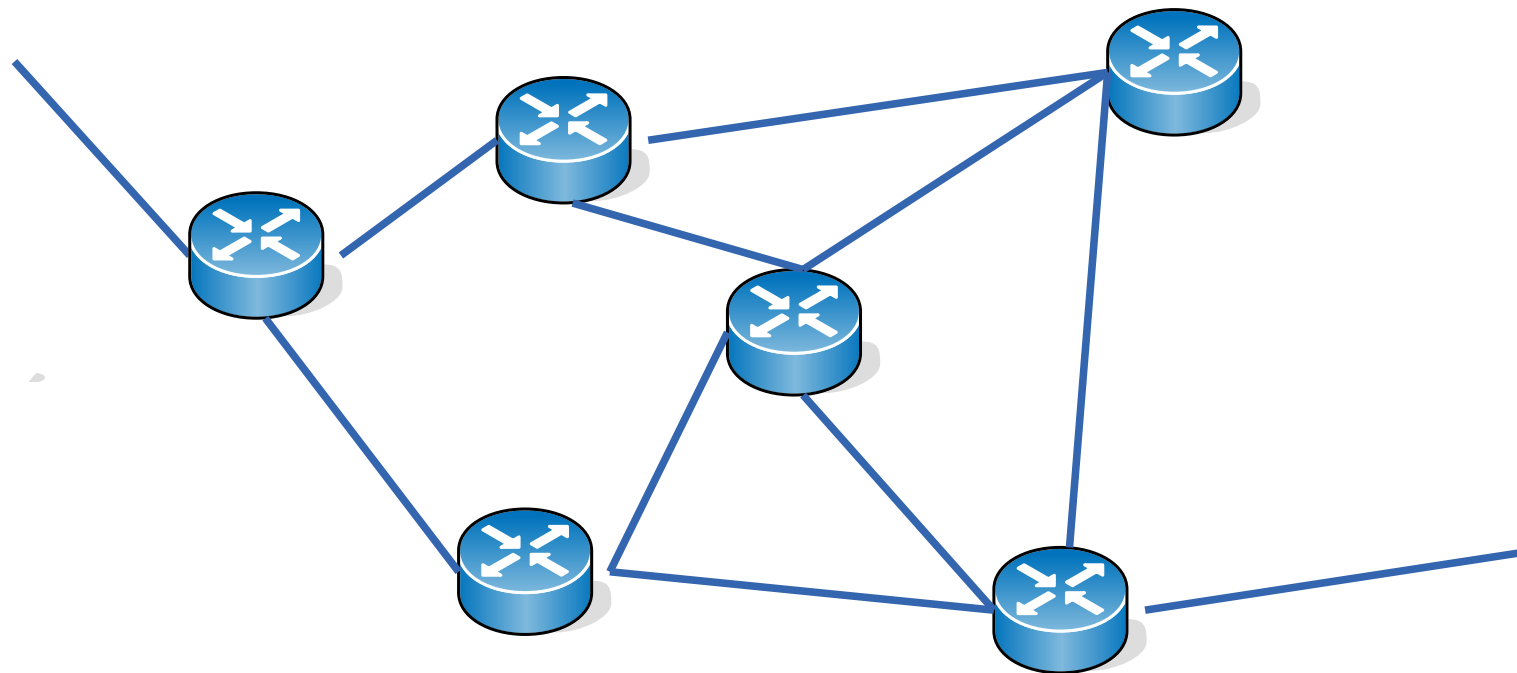
Redes de Circuitos Virtuales

- Circuitos Virtuales: Arquitectura de red que intenta abordar esos problemas
- La idea es los paquetes sigan siempre un mismo camino
- Se elige un camino (**circuito virtual**, **CV**) y se usa durante toda la comunicación
 - La elección del camino lo hace la función de ruteo
- Se identifican 3 fases:
 - **Establecimiento** del CV
 - **Uso** del CV (envío de datos)
 - **Liberación** del CV

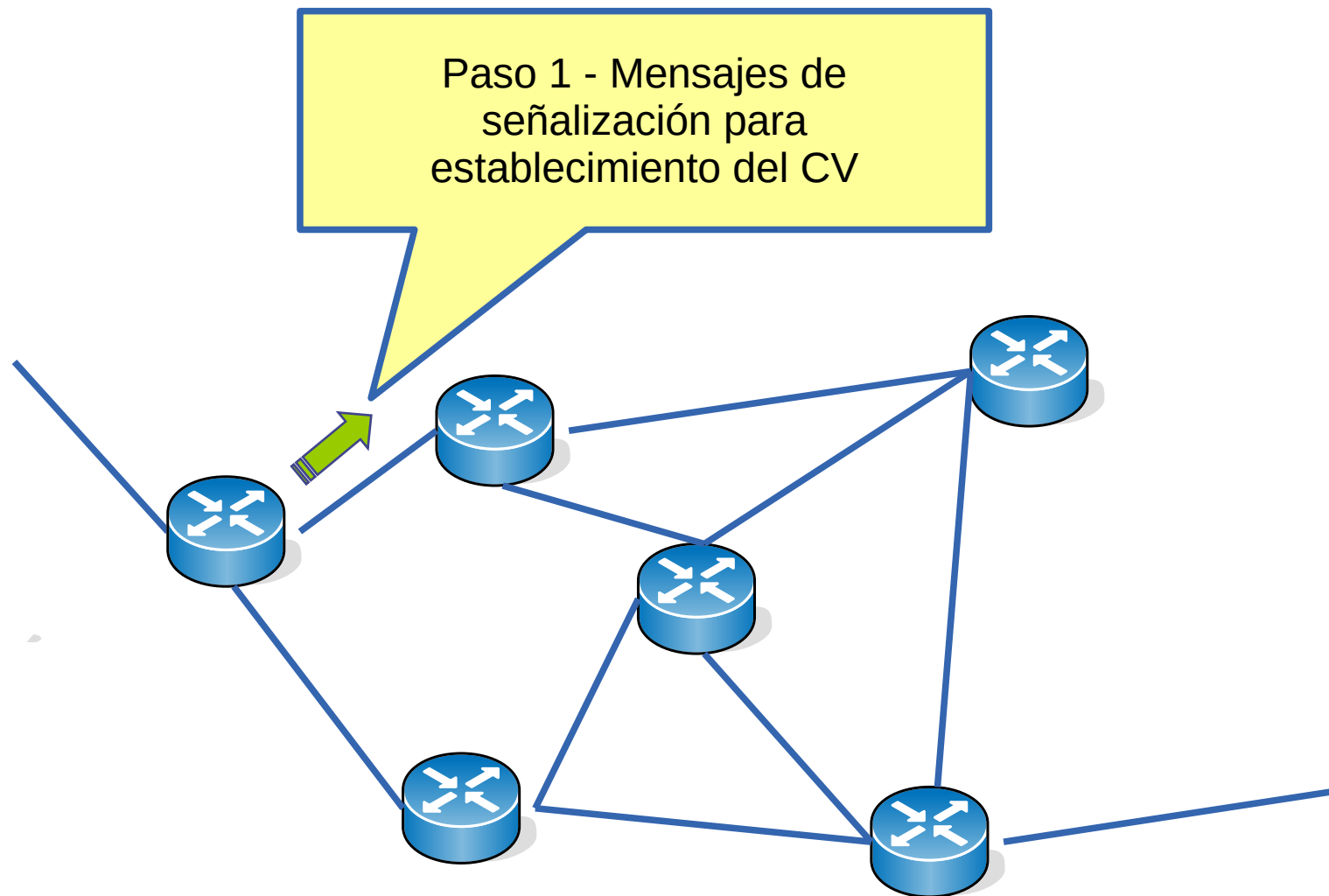
Redes de Circuitos Virtuales

- Ventajas:
 - En el establecimiento del CV se pueden identificar las zonas congestionadas
 - Se podrían eludir las zonas congestionadas usando caminos alternativos
 - Se podría no aceptar el establecimiento de un CV si no se se puede garantizar la calidad requerida
 - Se pueden negociar y reservar recursos de capacidad en nodos y enlaces
 - Todos los paquetes siguen la misma ruta por lo que los retardos y la variación de retardos será **más predecible**

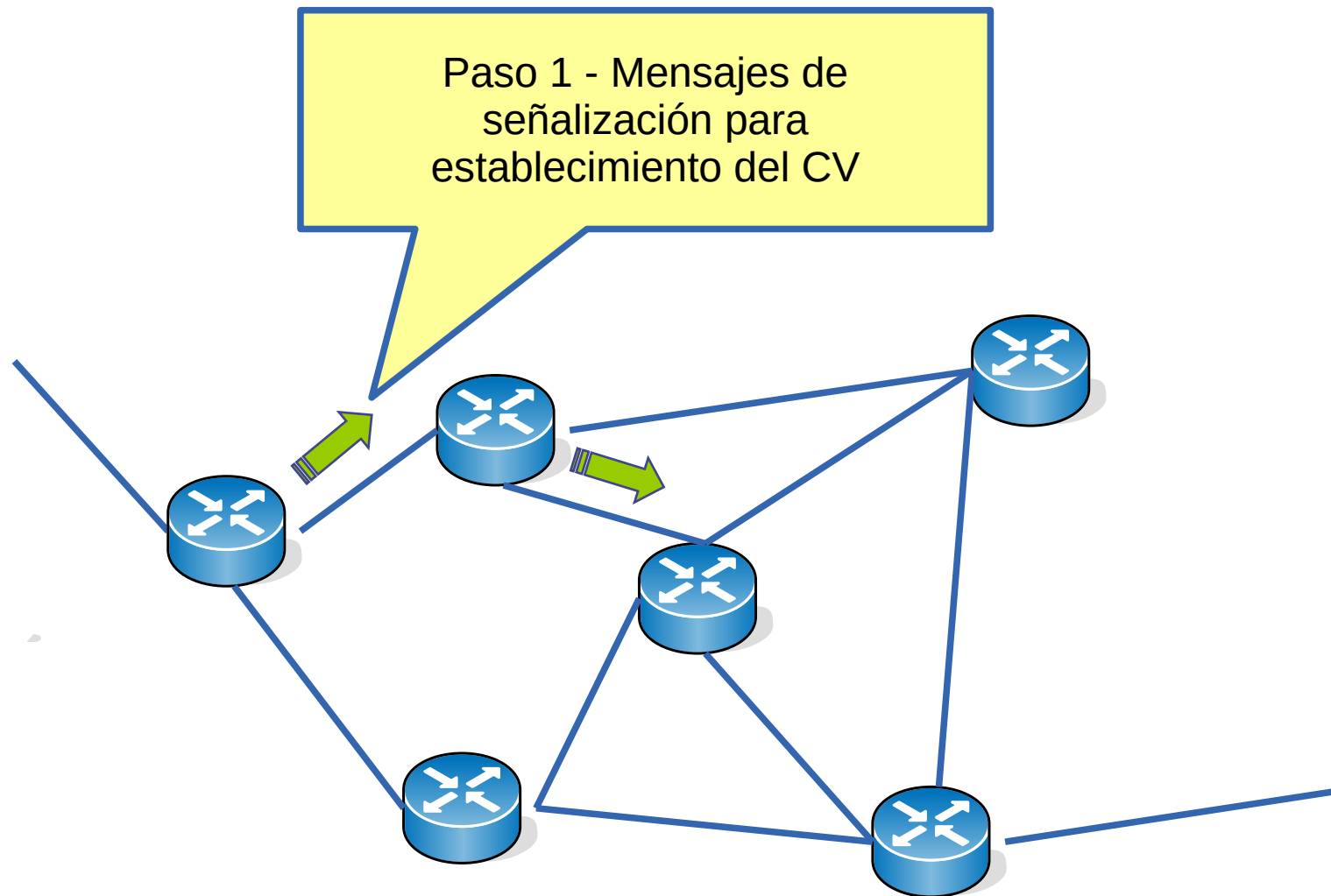
Redes de Circuitos Virtuales



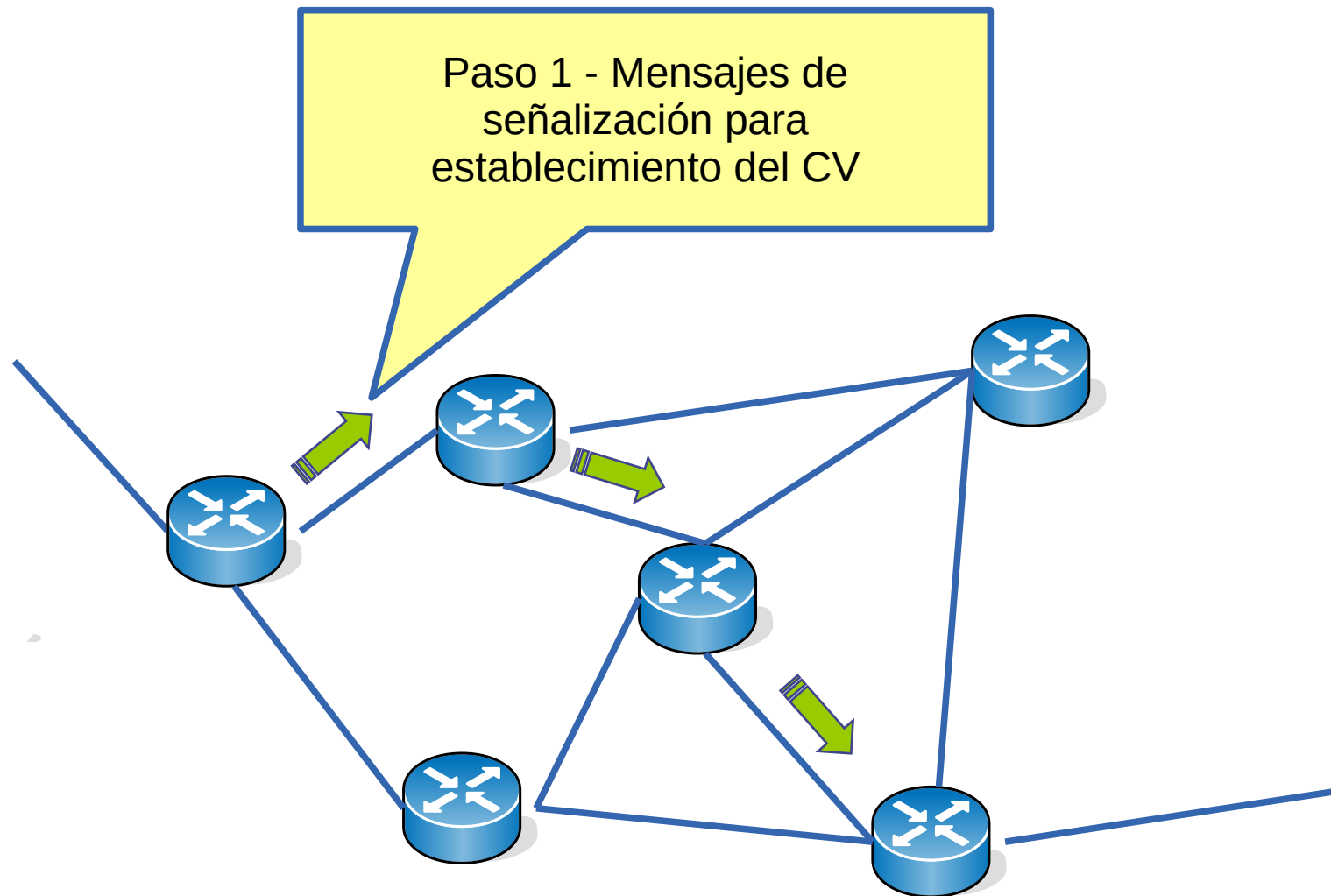
Redes de Circuitos Virtuales



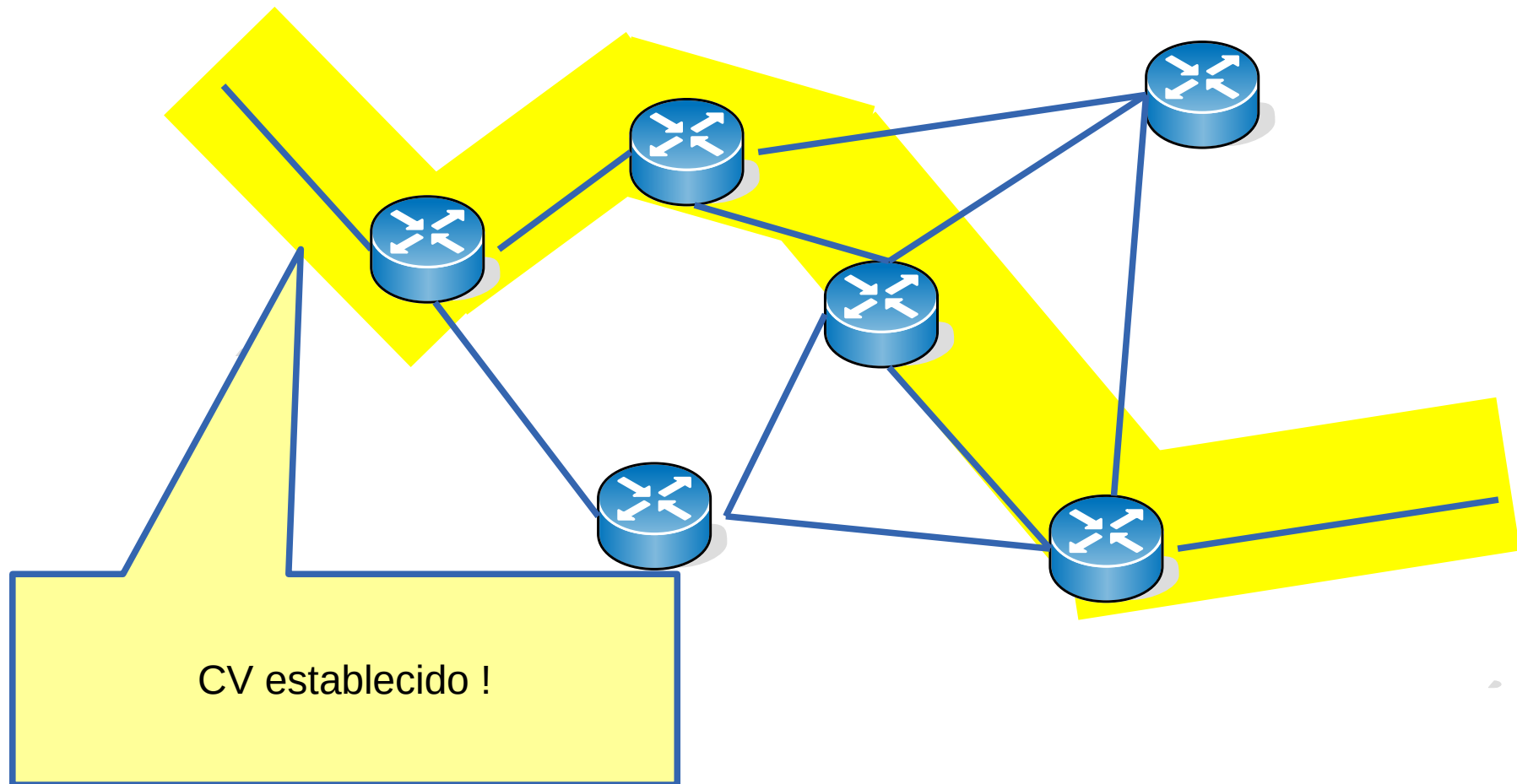
Redes de Circuitos Virtuales



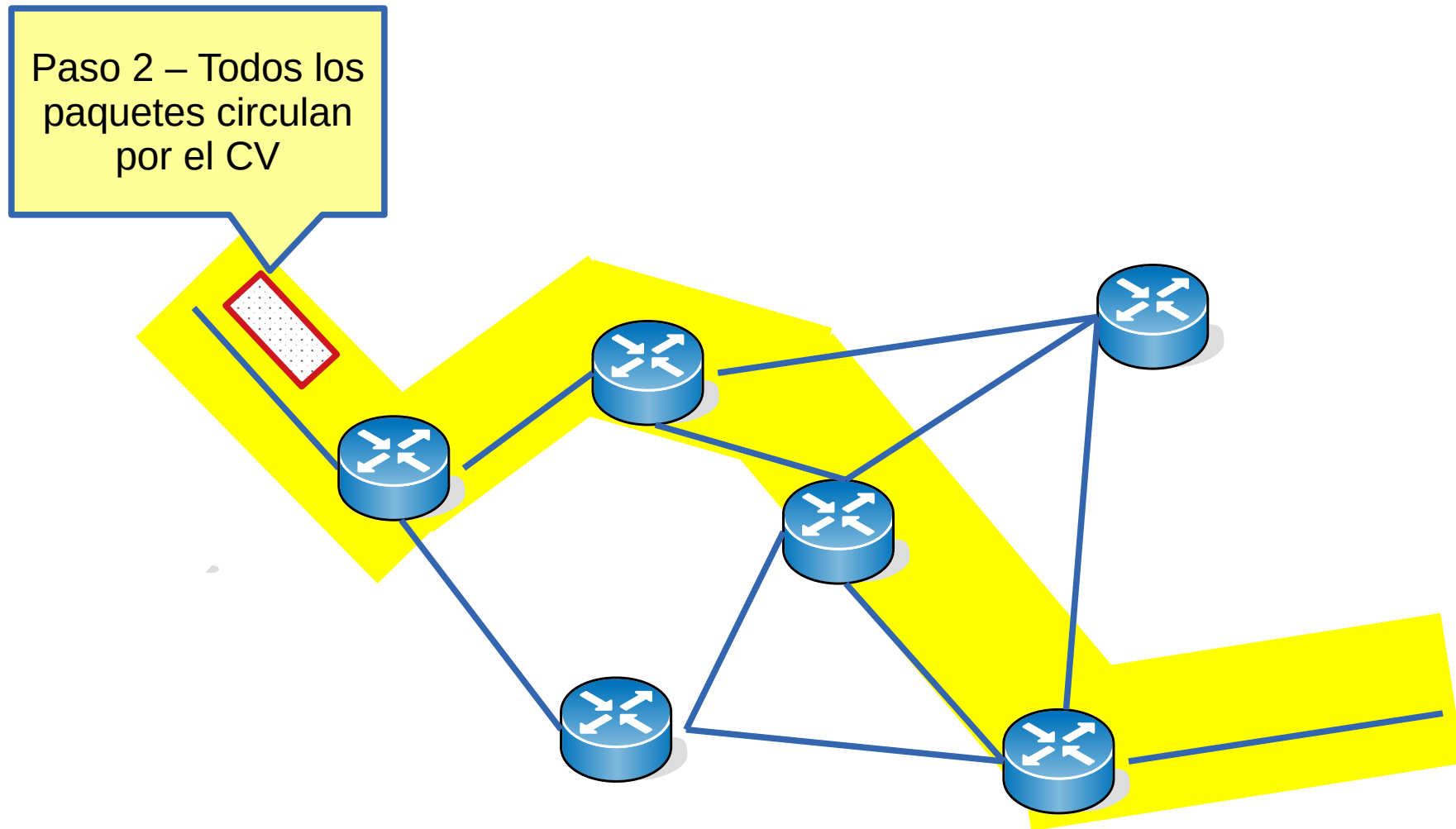
Redes de Circuitos Virtuales



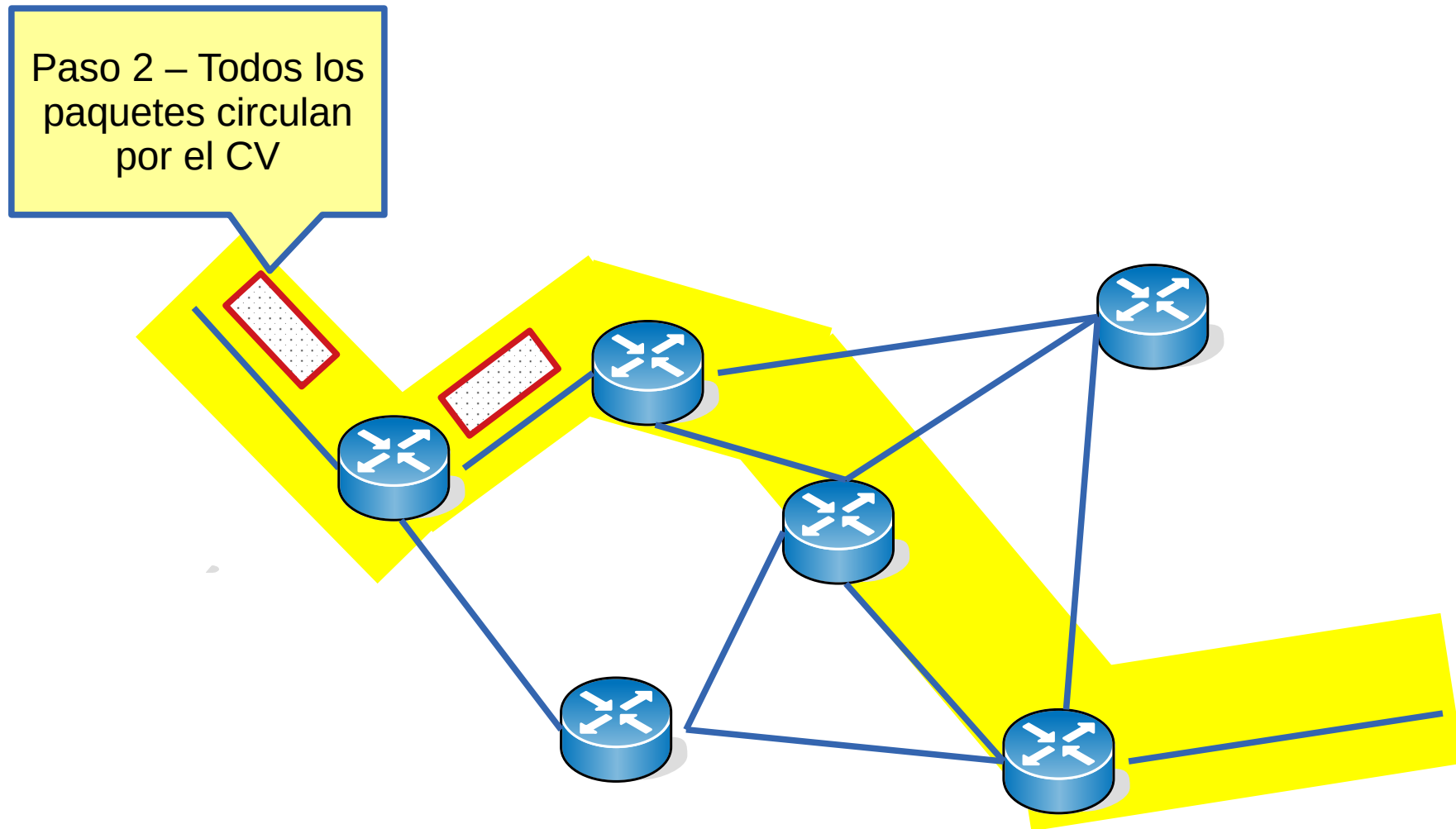
Redes de Circuitos Virtuales



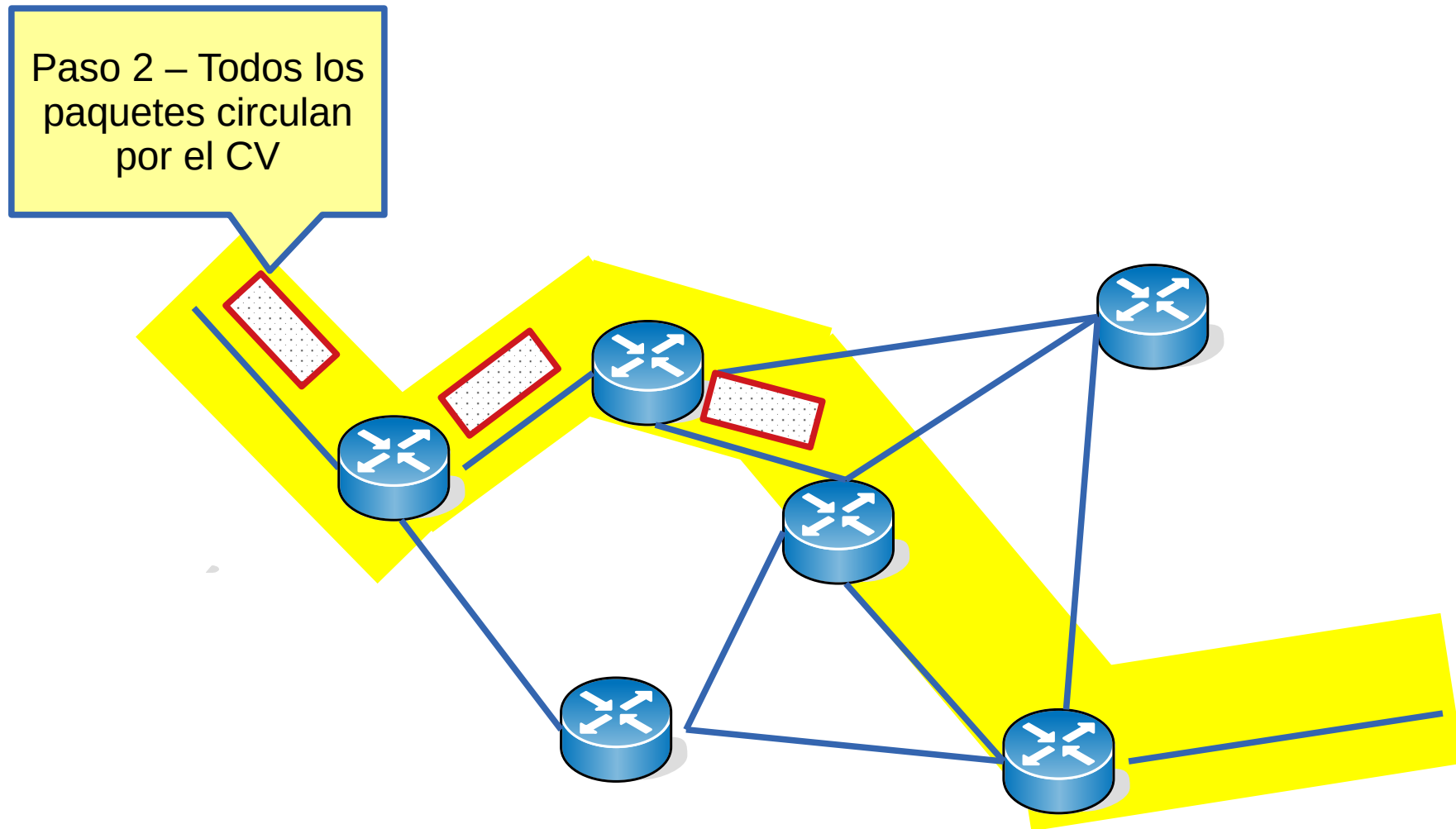
Redes de Circuitos Virtuales



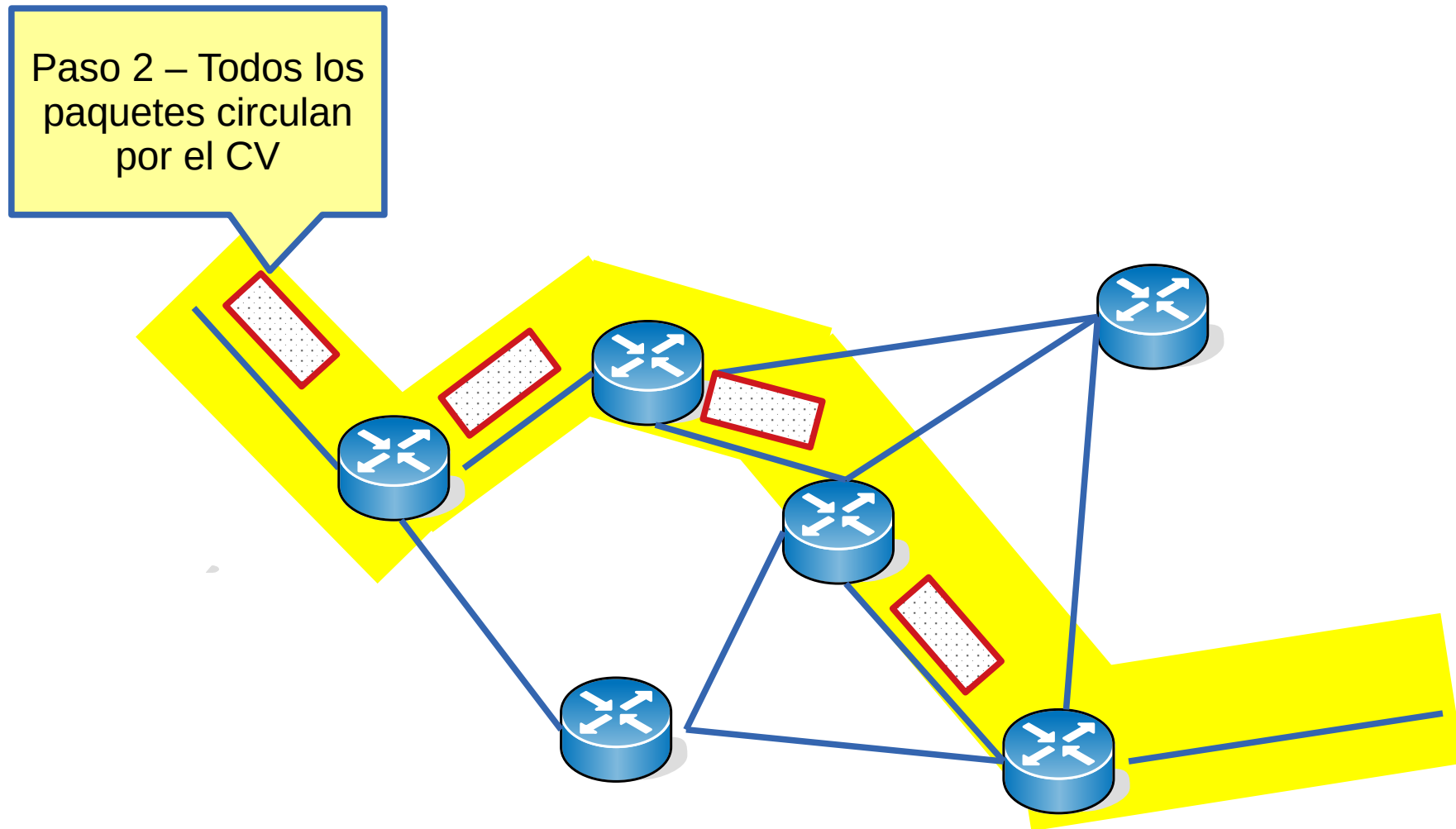
Redes de Circuitos Virtuales



Redes de Circuitos Virtuales

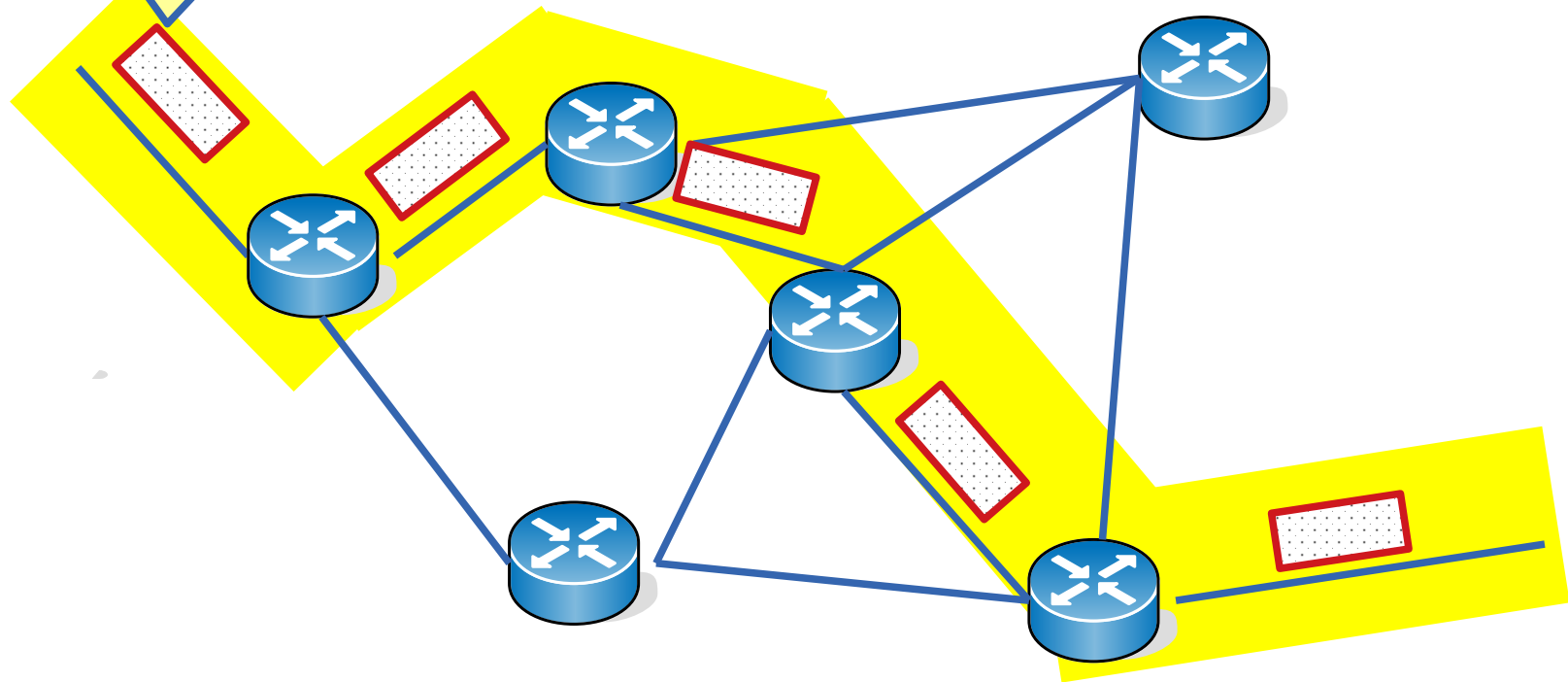


Redes de Circuitos Virtuales

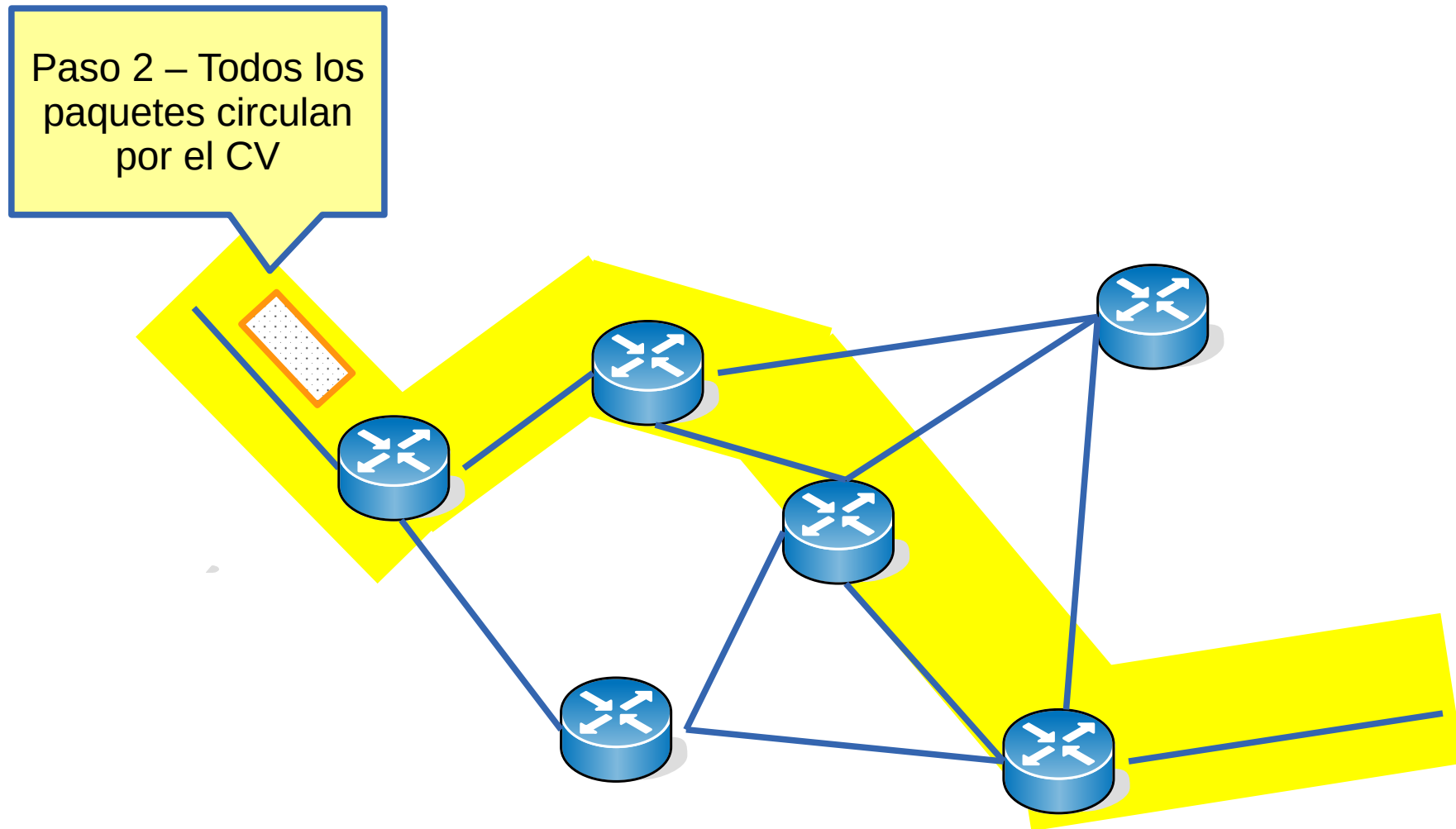


Redes de Circuitos Virtuales

Paso 2 – Todos los paquetes circulan por el CV

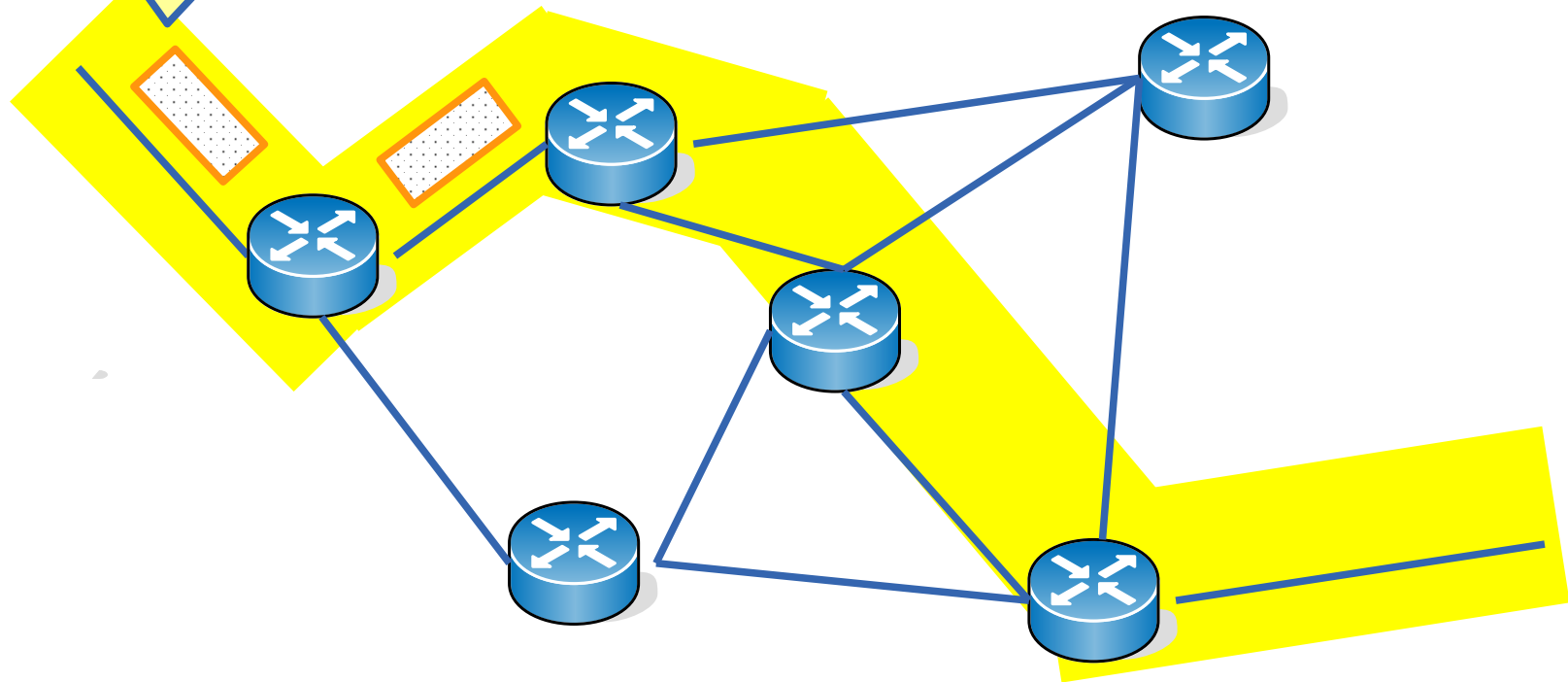


Redes de Circuitos Virtuales



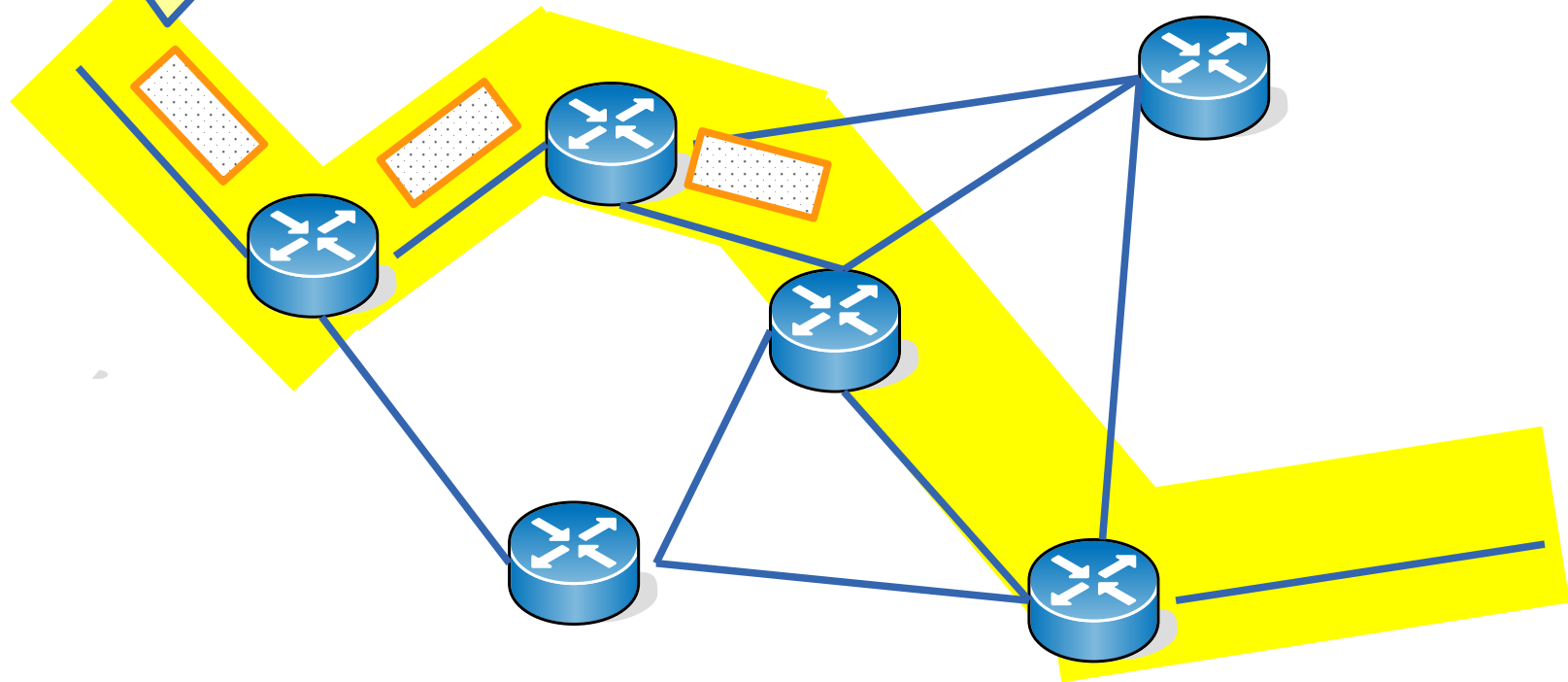
Redes de Circuitos Virtuales

Paso 2 – Todos los paquetes circulan por el CV



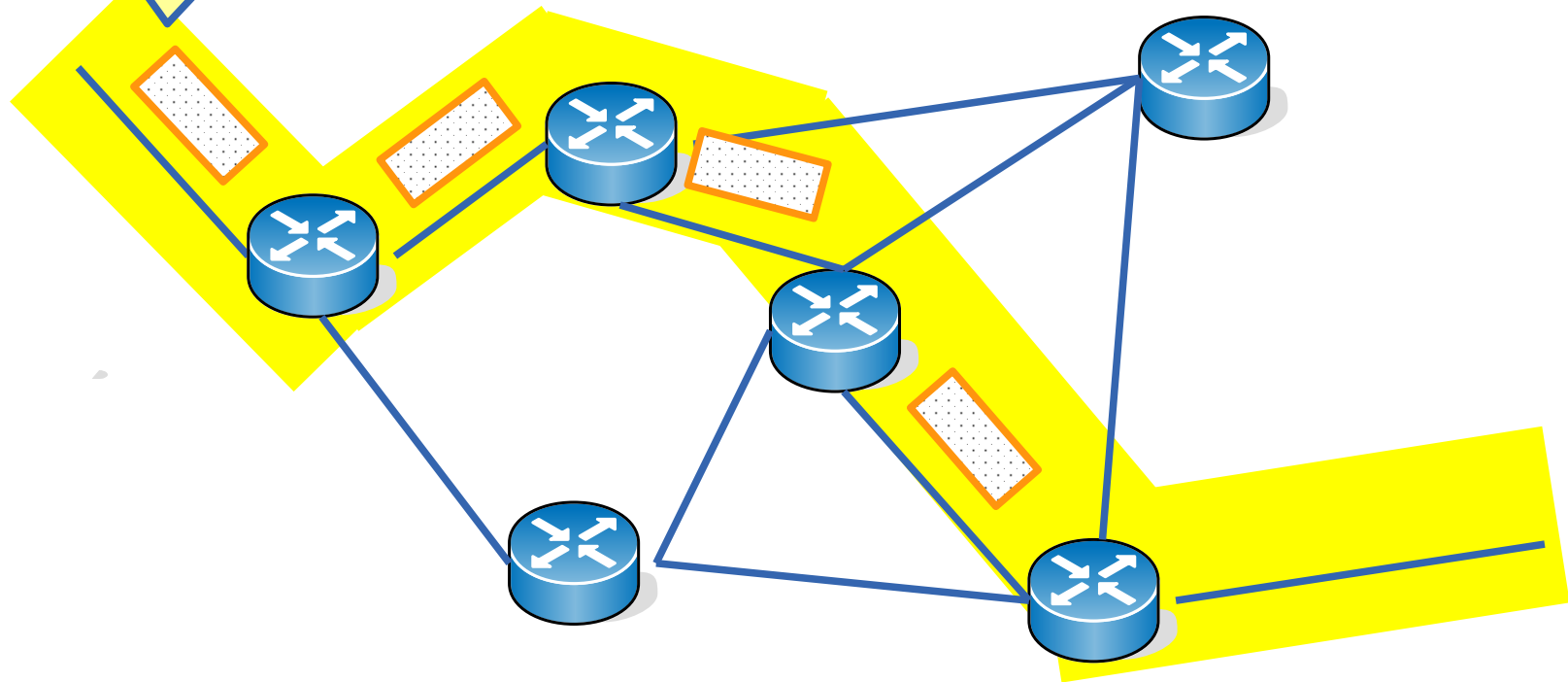
Redes de Circuitos Virtuales

Paso 2 – Todos los paquetes circulan por el CV



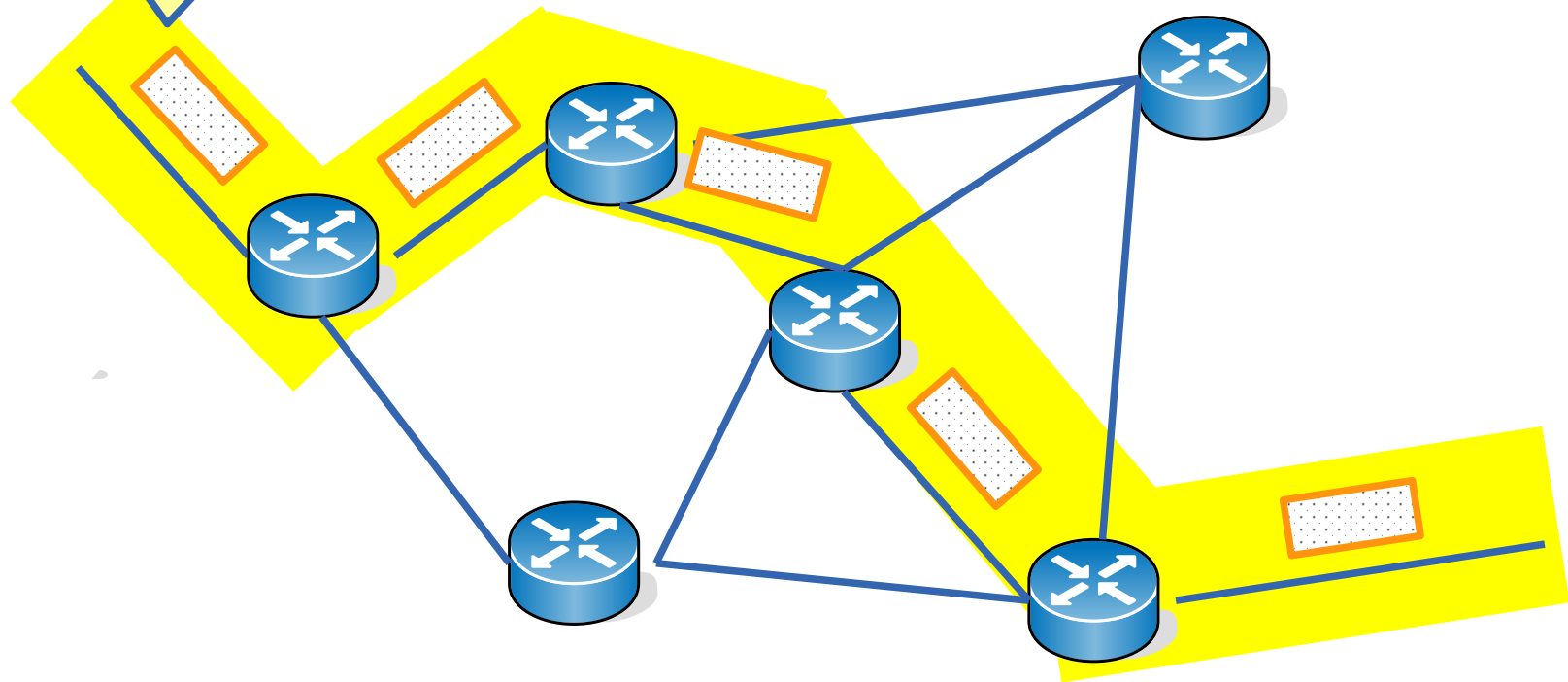
Redes de Circuitos Virtuales

Paso 2 – Todos los paquetes circulan por el CV

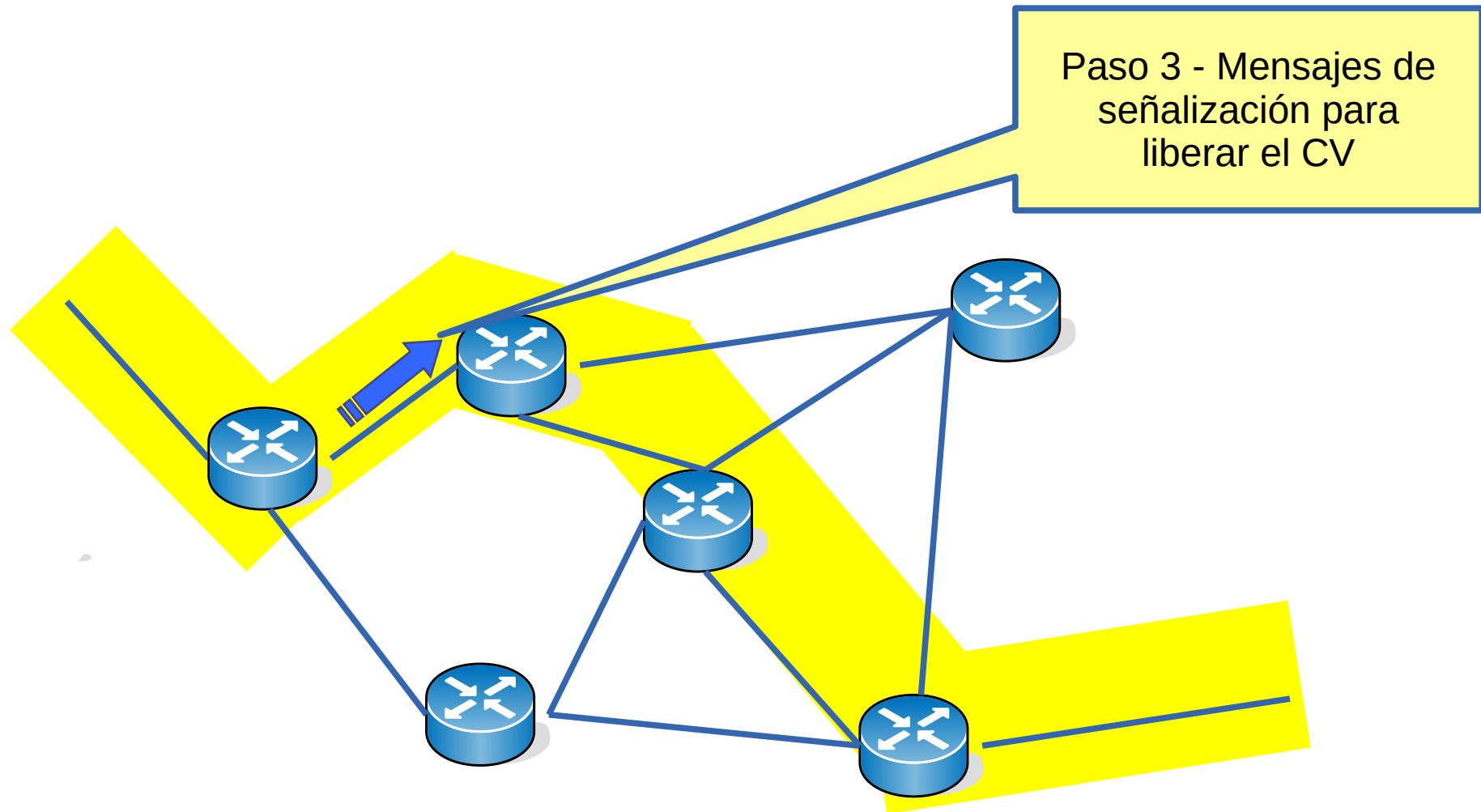


Redes de Circuitos Virtuales

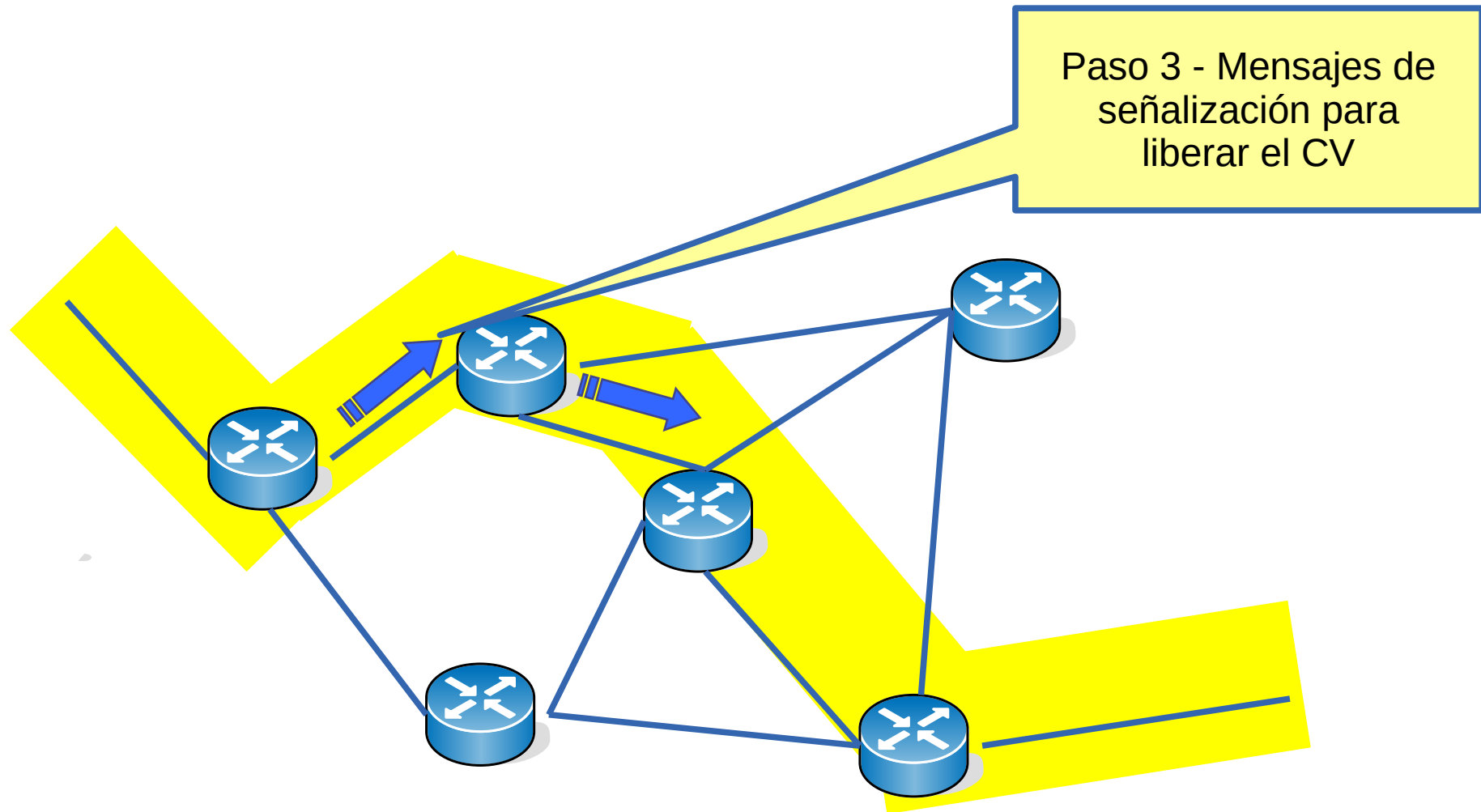
Paso 2 – Todos los paquetes circulan por el CV



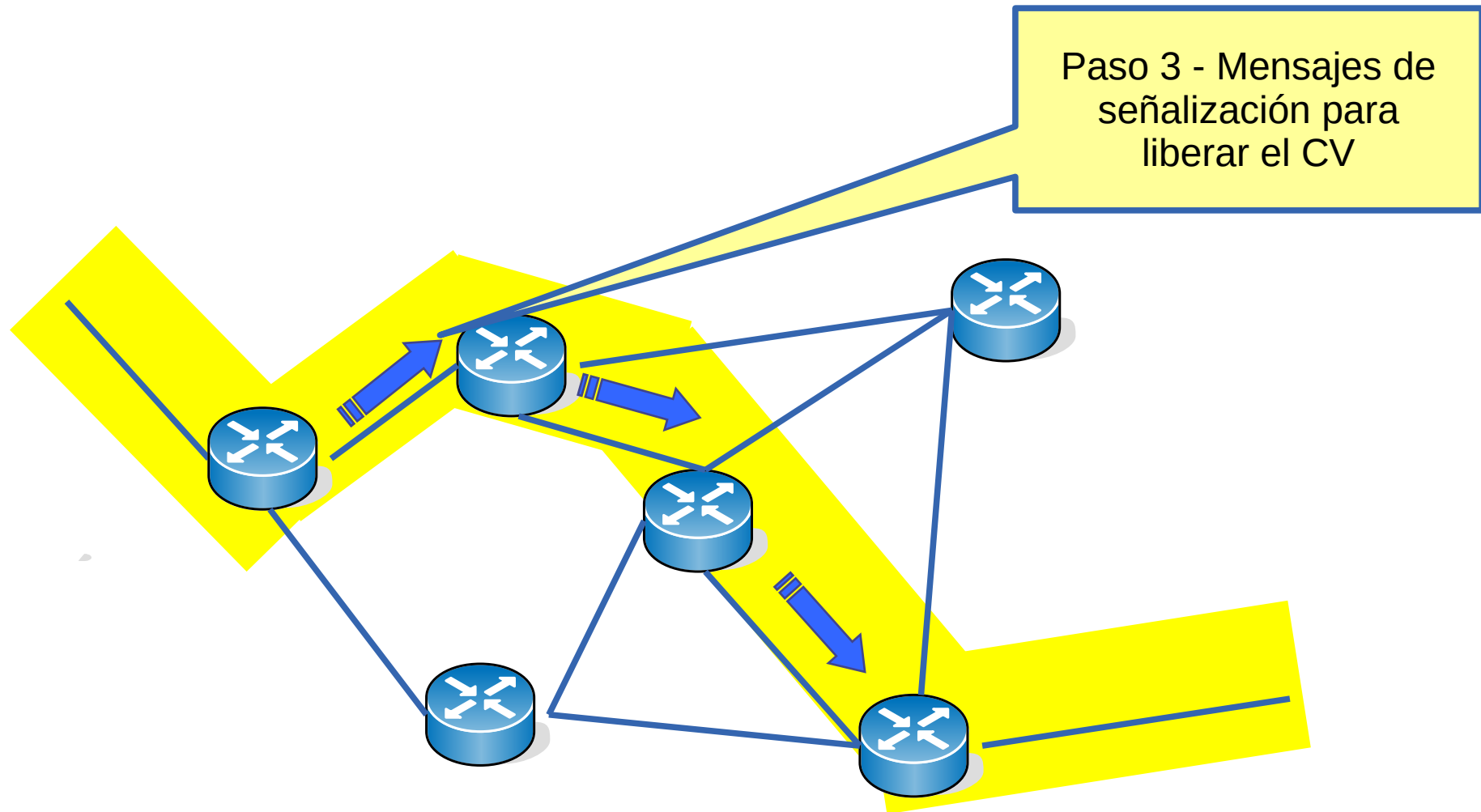
Redes de Circuitos Virtuales



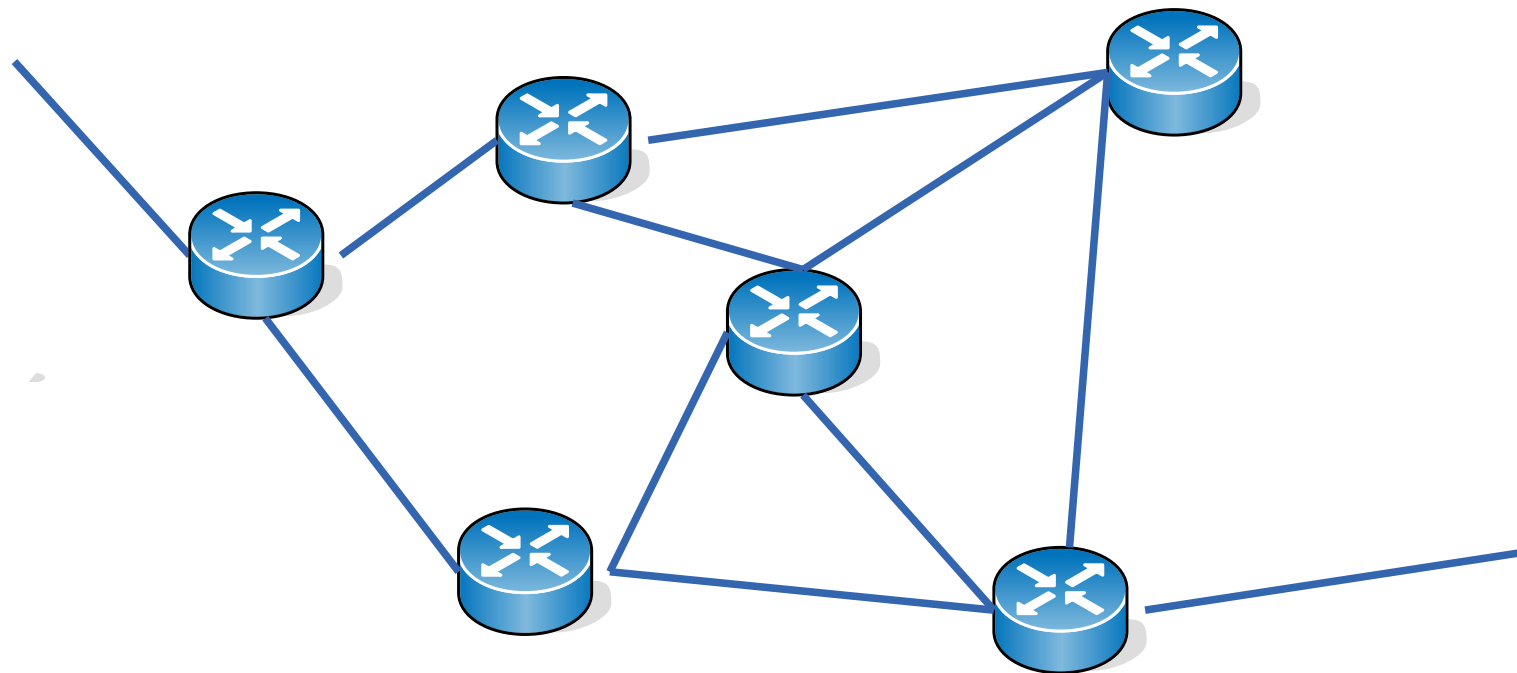
Redes de Circuitos Virtuales



Redes de Circuitos Virtuales



Redes de Circuitos Virtuales



Redes de Circuitos Virtuales

- El camino se identifica por un número, genéricamente llamado “**identificador de Circuito Virtual**”
 - Distintas tecnologías lo llaman de otras maneras (**etiquetas**, DLCI, etc.)
- No es práctico tener un identificador de CV para todo el circuito porque requeriría acuerdo y disponibilidad del mismo número en todos los nodos intermedios
 - El CV se identifica por el conjunto de **identificadores de cada tramo**
 - Cada par de nodos acuerda un **identificador de CV local** para ese tramo
- Cada nodo debe mantener **tablas con los CVs establecidos**
 - Si un nodo falla, se pierden los CV establecidos en los que participa
 - Para poder continuar enviando información, deberá establecerse un nuevo CV

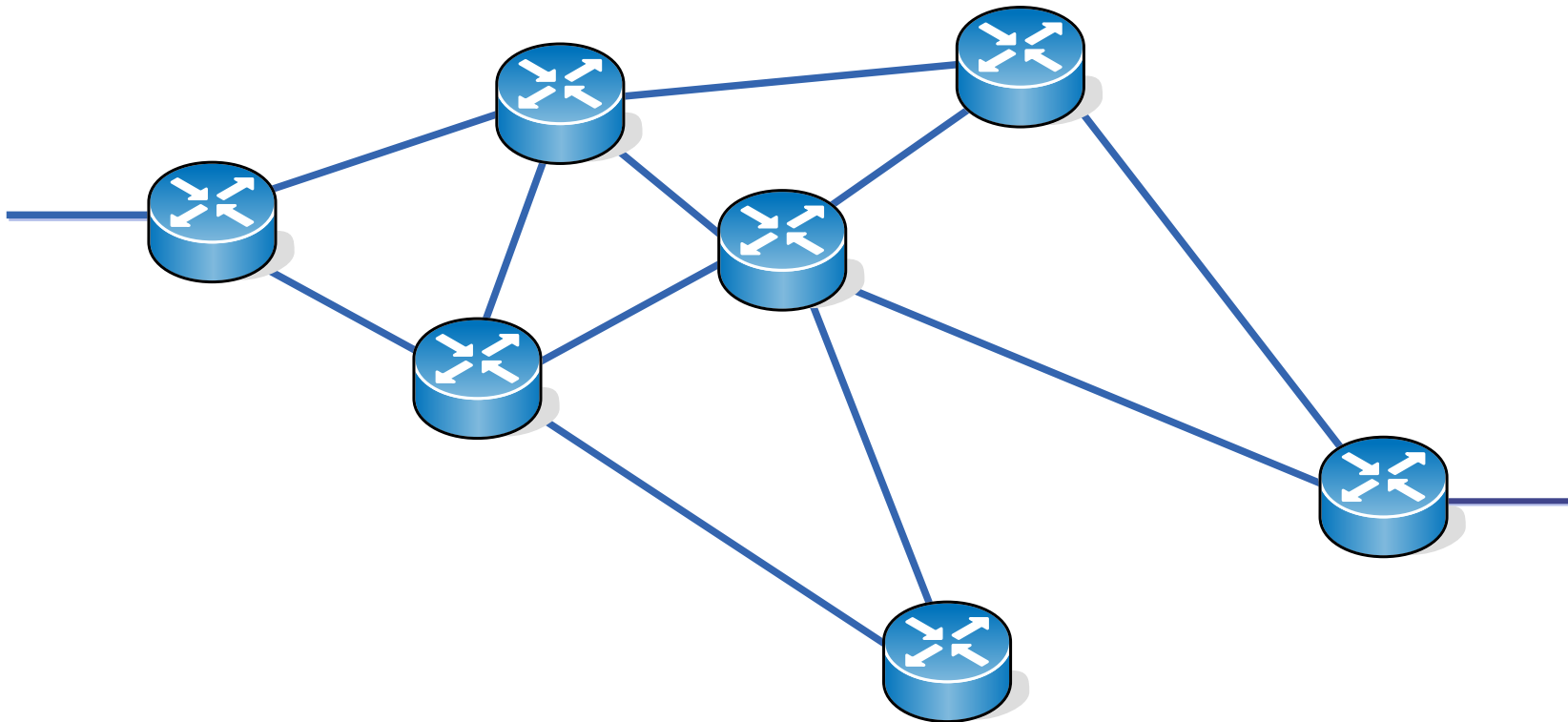
Tipos de Circuitos Virtuales

- Estáticos o **permanentes**
 - Se configuran administrativamente por el proveedor
 - Permanent Virtual Circuit, **PVC**
- Dinámicos o **conmutados**
 - Se establece el CV cuando se necesita
 - Switched Virtual Circuit, **SVC**
 - Se requiere señalización (mensajes de control) para establecerlo
 - Con esa señalización se asignan los identificadores de CV de cada tramo

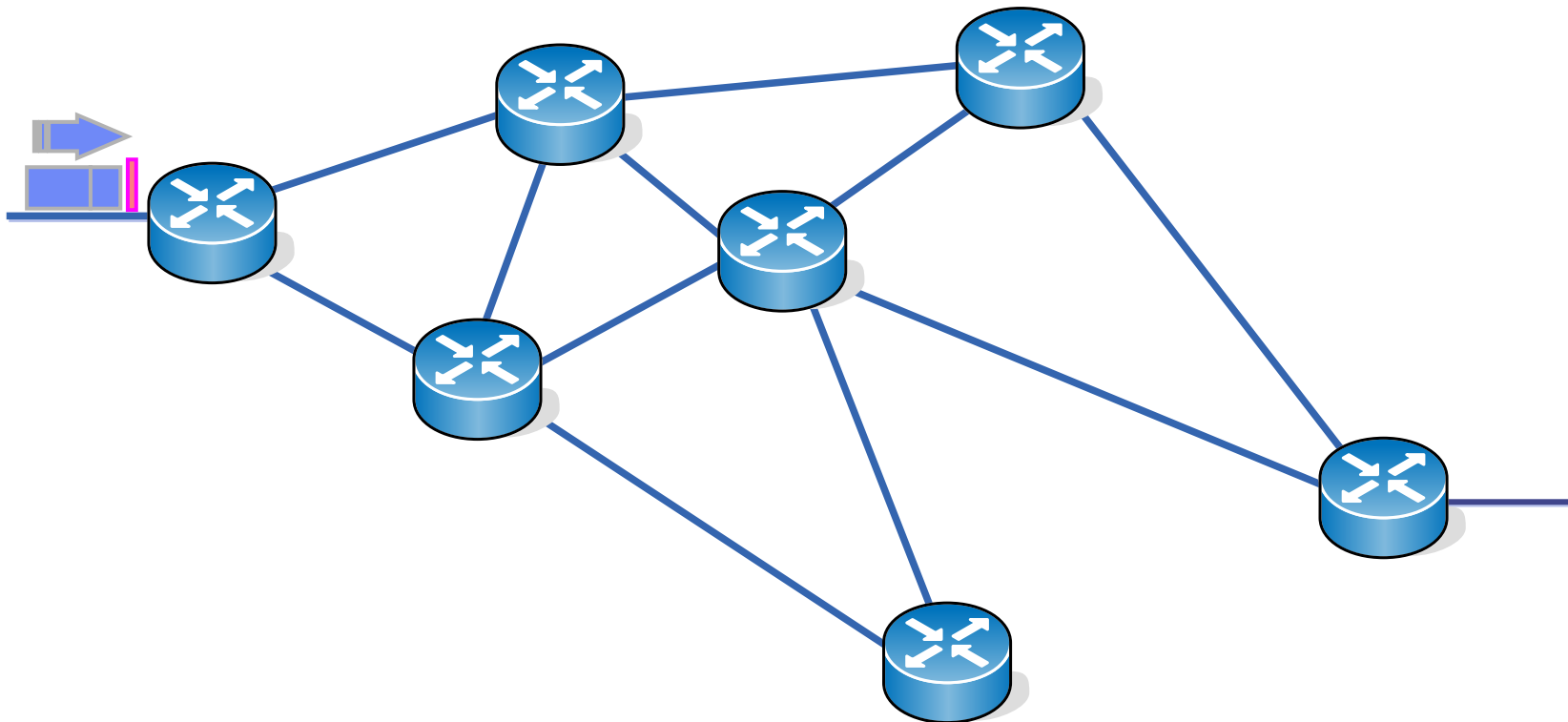
Encaminamiento o forwarding en redes de CV

- Los nodos cuando conmutan un paquete deberán cambiar el identificador de entrada (acordado con el nodo anterior) por el de salida (acordado con el nodo siguiente)
- El identificador de CV será parte del encabezado del paquete
- Los nodos tendrán una tabla de los CV que tienen establecidos con las correspondencias de identificador de entrada con identificador de salida
- Se dice que los nodos hacen cambios de identificadores o etiquetas (“[label swapping](#)”)
- El encaminamiento en nodos intermedios es mucho más sencillo que en datagramas
 - consiste en encontrar para cada pareja de identificador de CV de entrada e interfaz de entrada, cuál es el identificador de CV de salida y la interfaz de salida

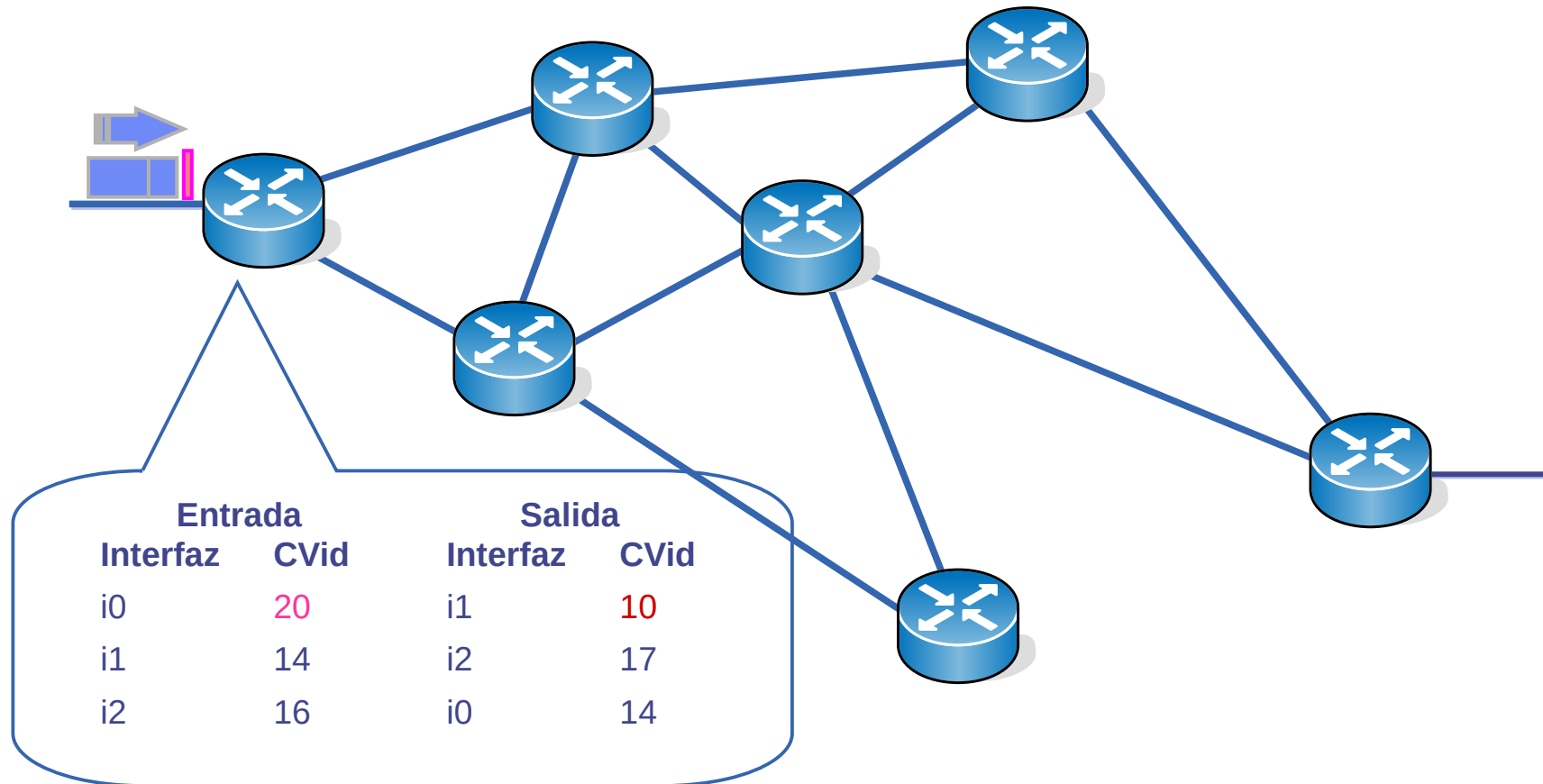
Tablas de forwarding en Circuitos Virtuales



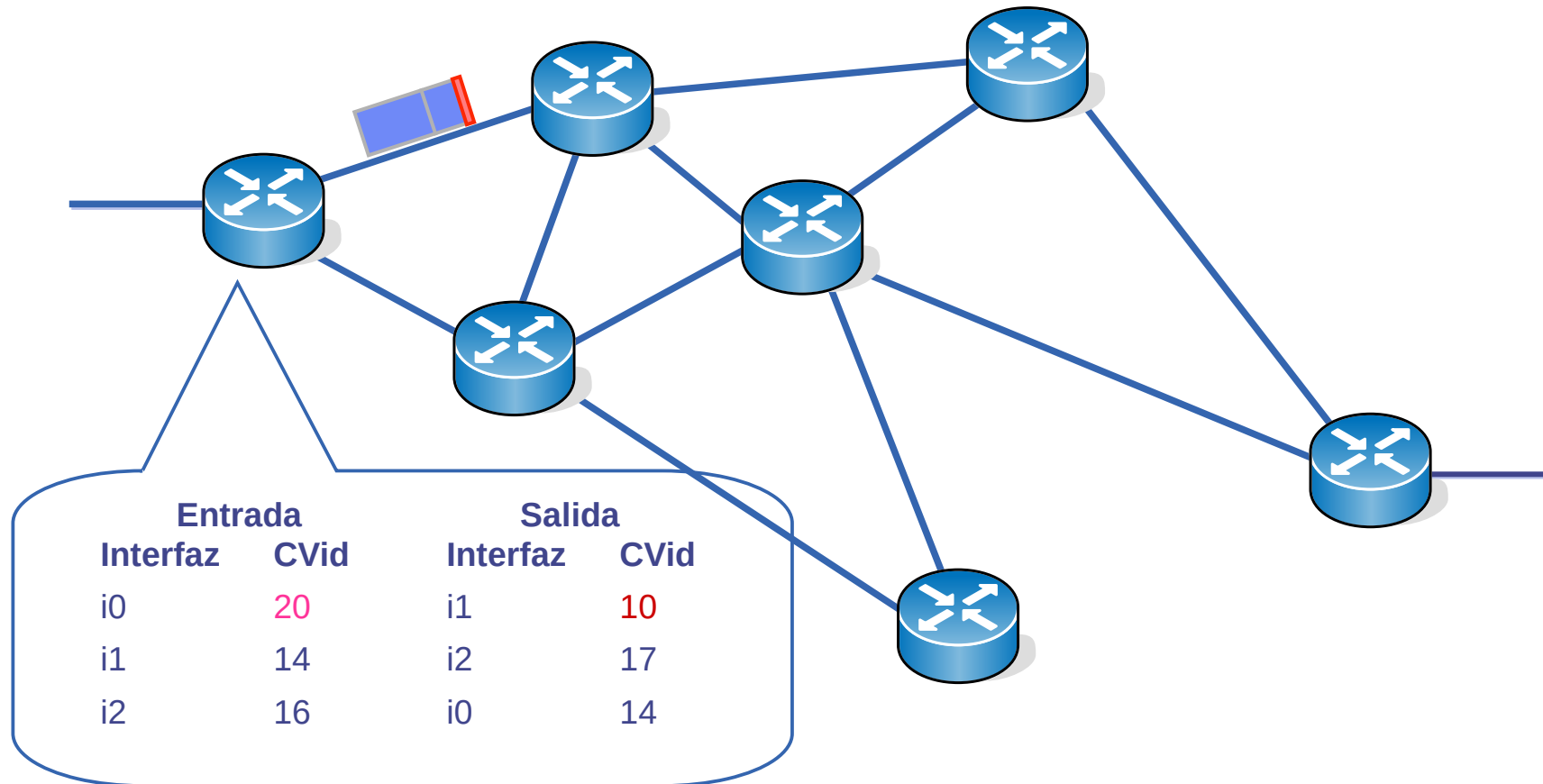
Tablas de forwarding en Circuitos Virtuales



Tablas de forwarding en Circuitos Virtuales

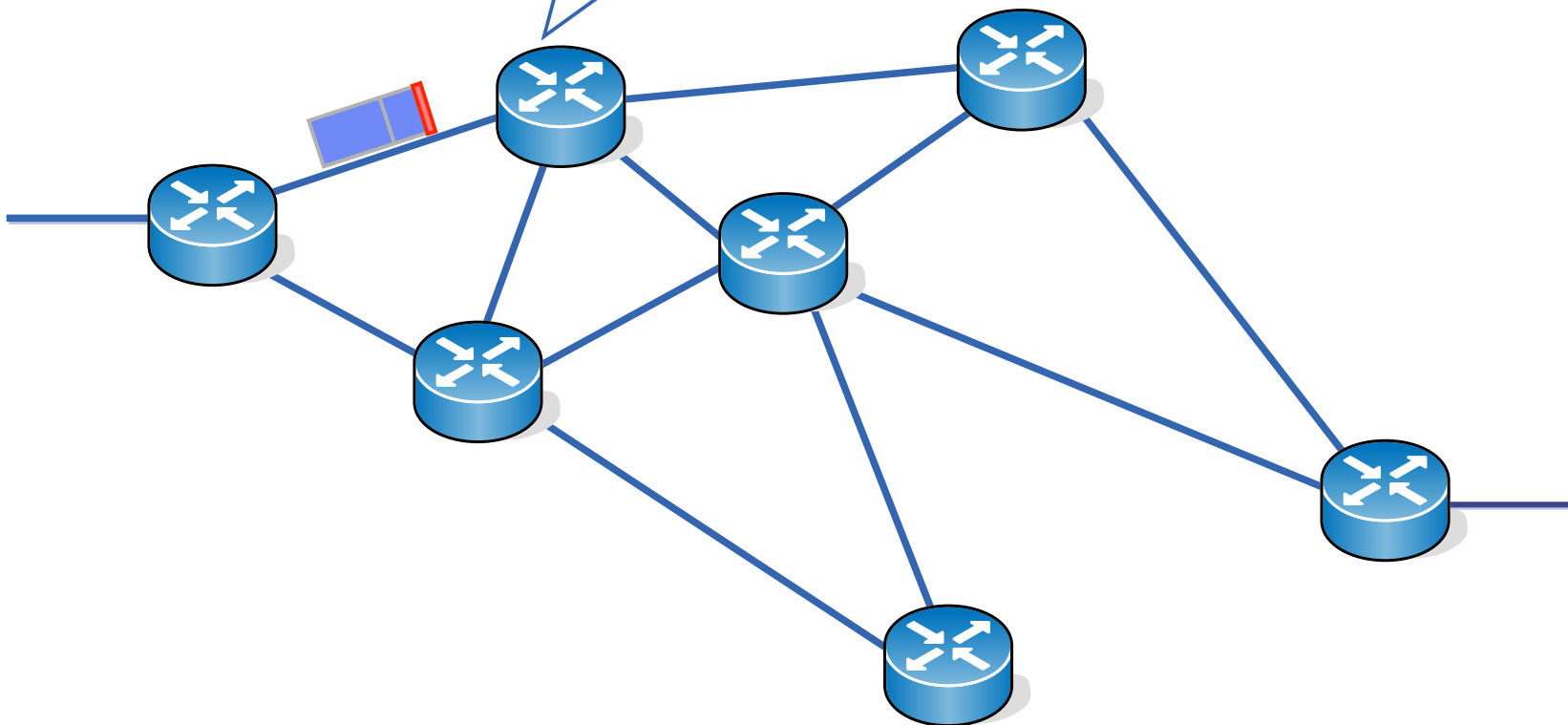


Tablas de forwarding en Circuitos Virtuales



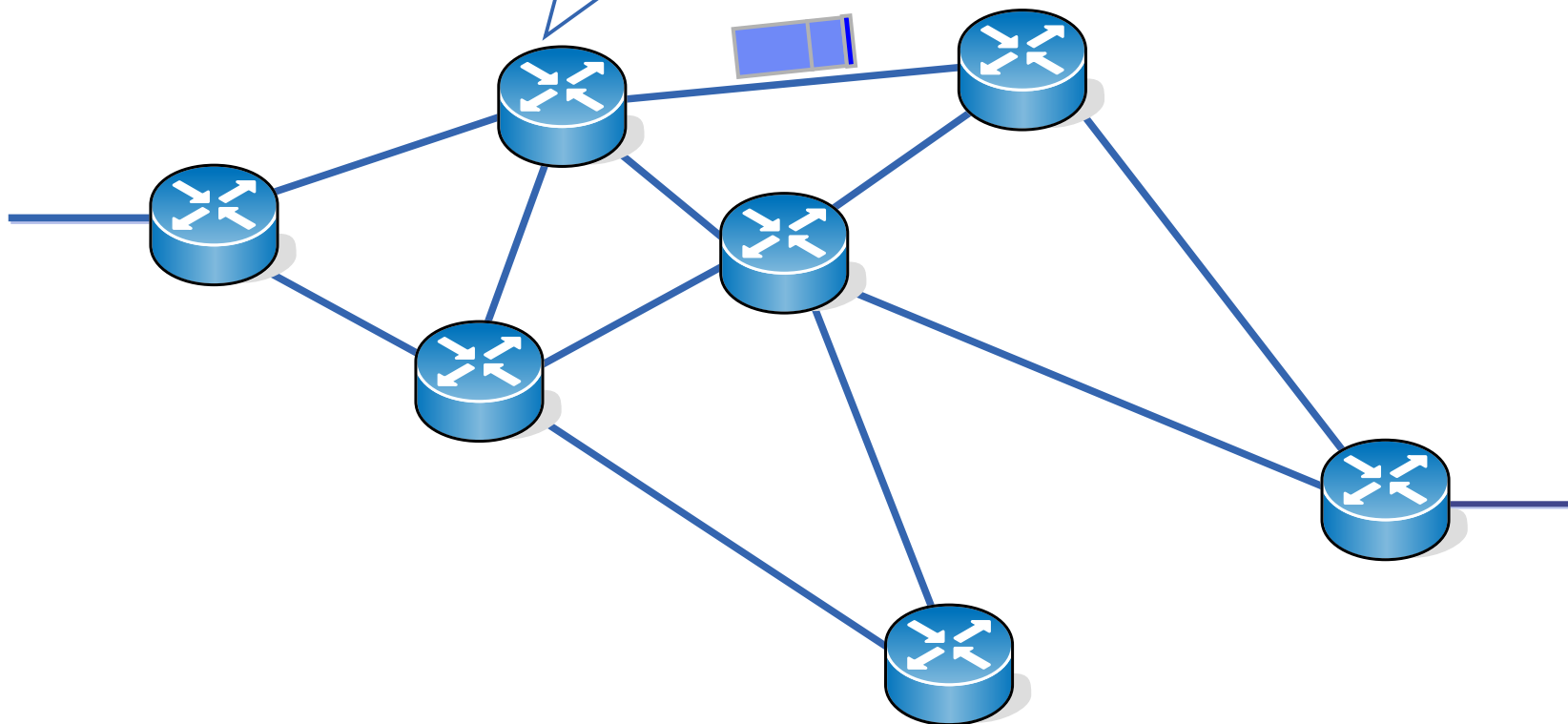
Tablas de forwarding en Circuitos Virtuales

Entrada		Salida	
Interfaz	CVid	Interfaz	Cvid
i0	10	i3	26
i1	14	i2	17
i2	16	i3	14

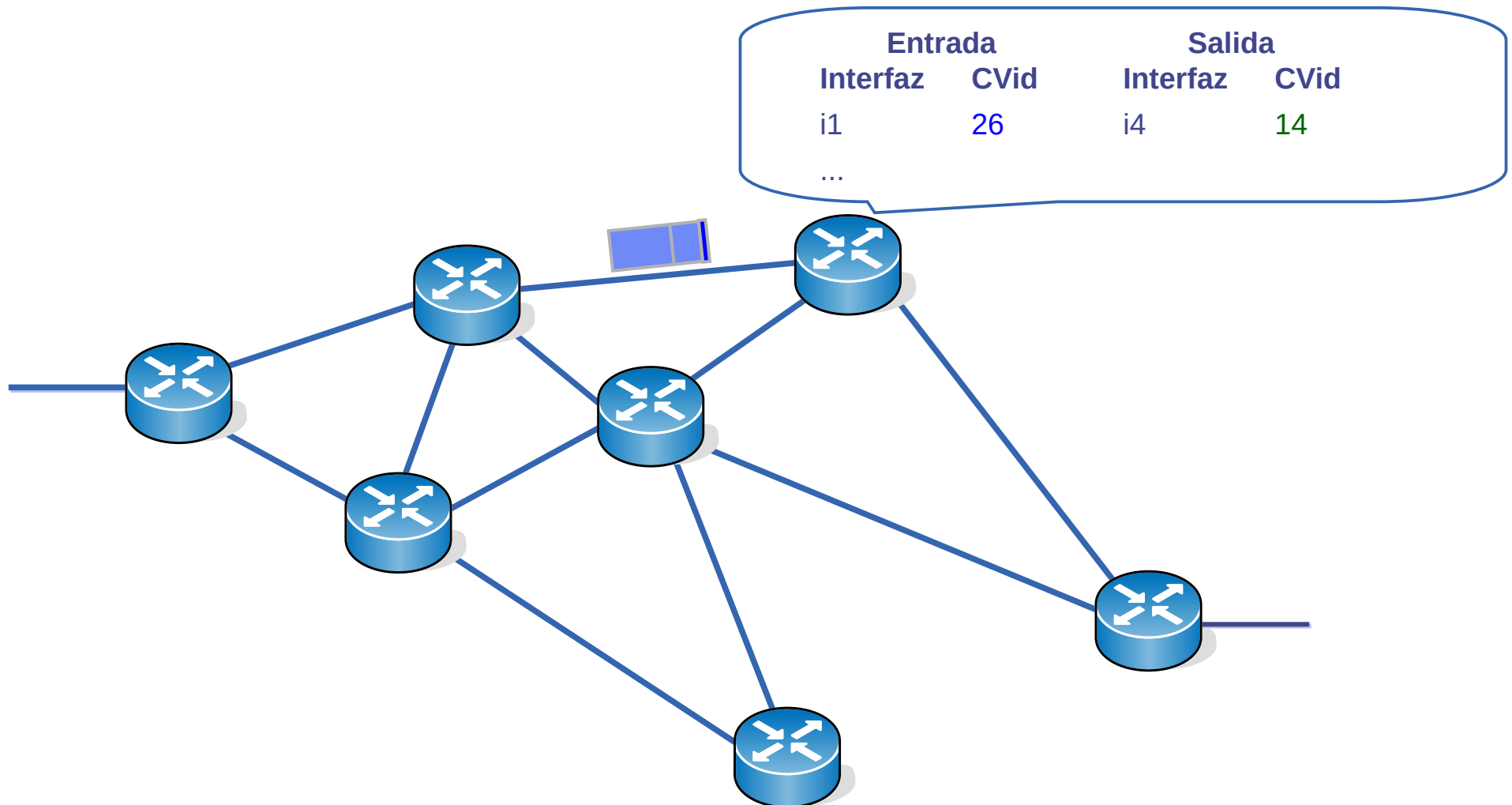


Tablas de forwarding en Circuitos Virtuales

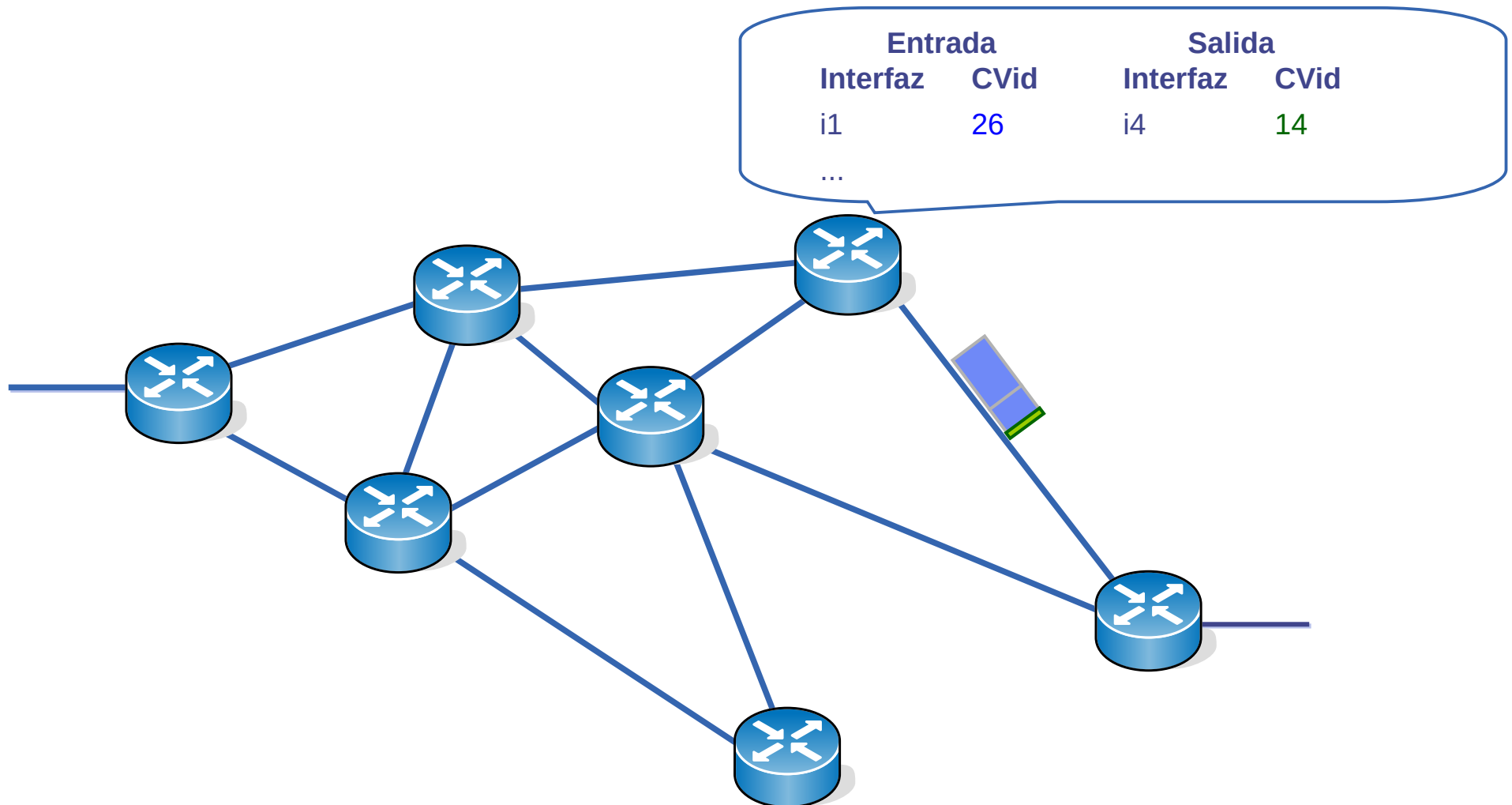
Entrada		Salida	
Interfaz	CVid	Interfaz	Cvid
i0	10	i3	26
i1	14	i2	17
i2	16	i3	14



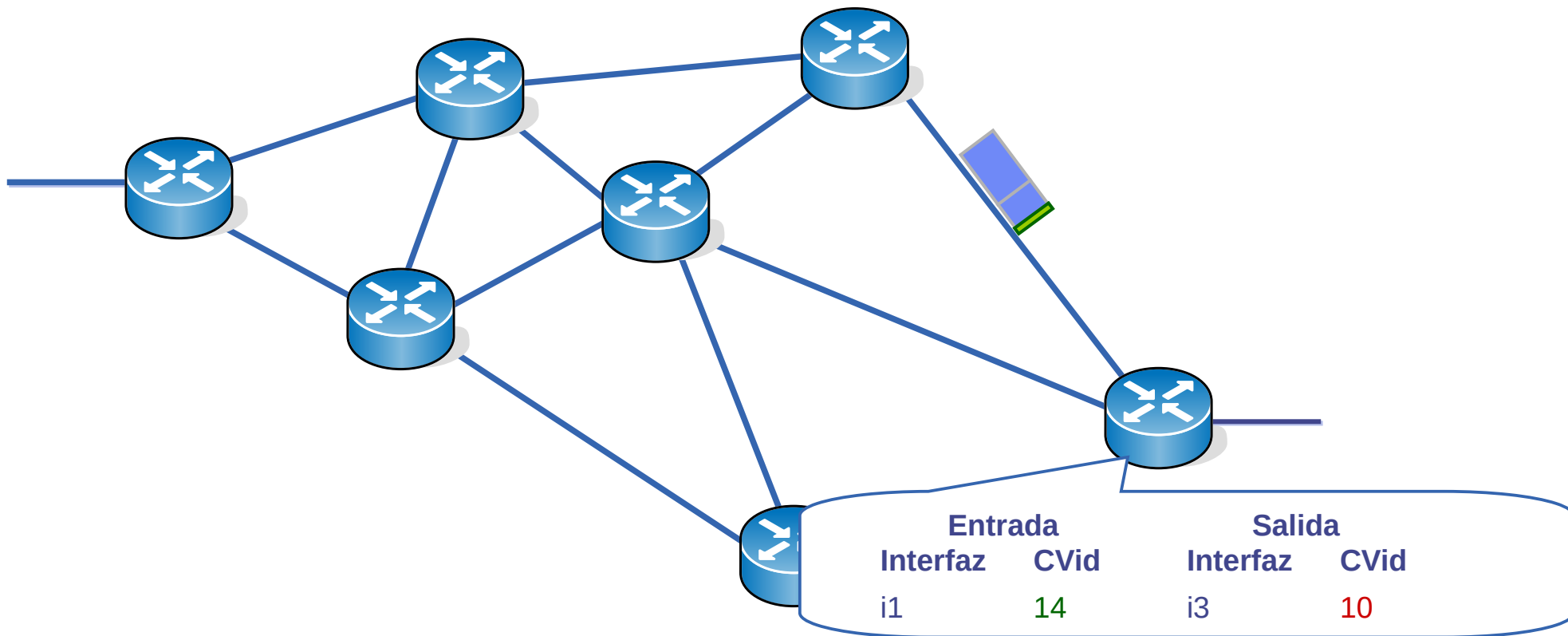
Tablas de forwarding en Circuitos Virtuales



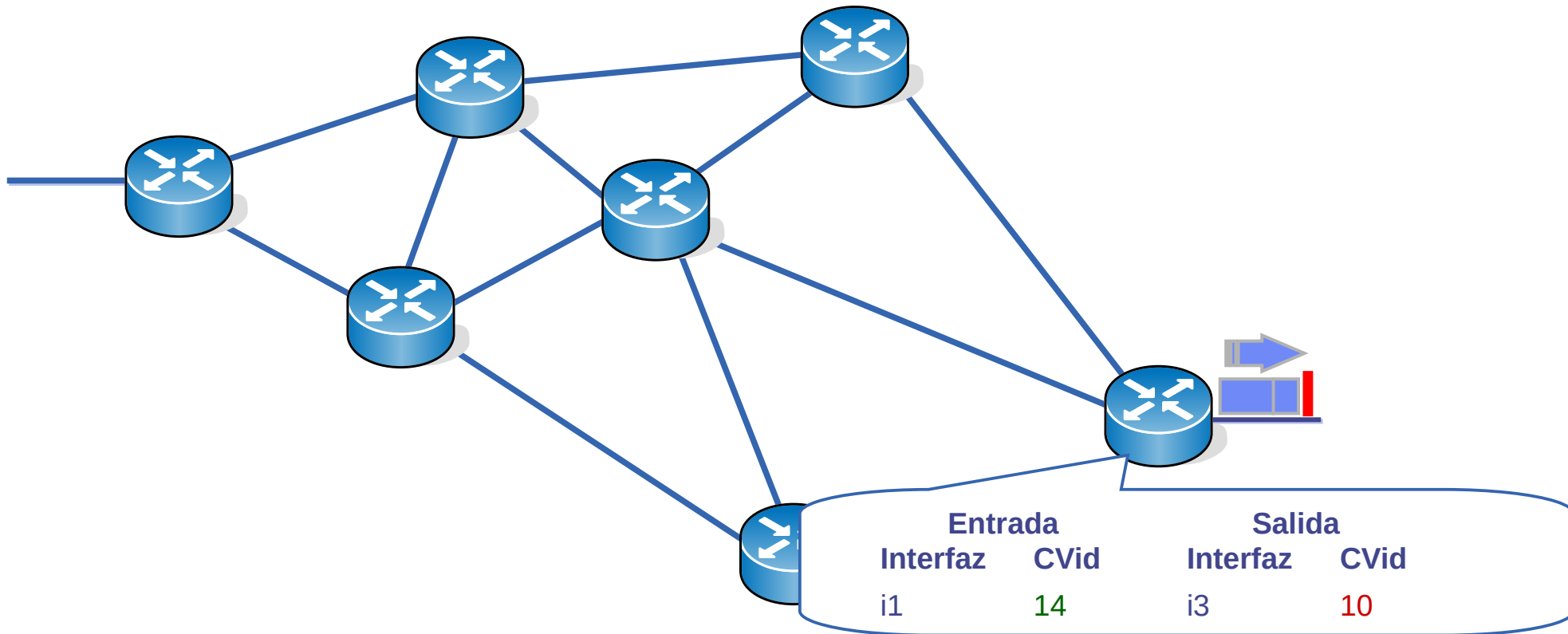
Tablas de forwarding en Circuitos Virtuales



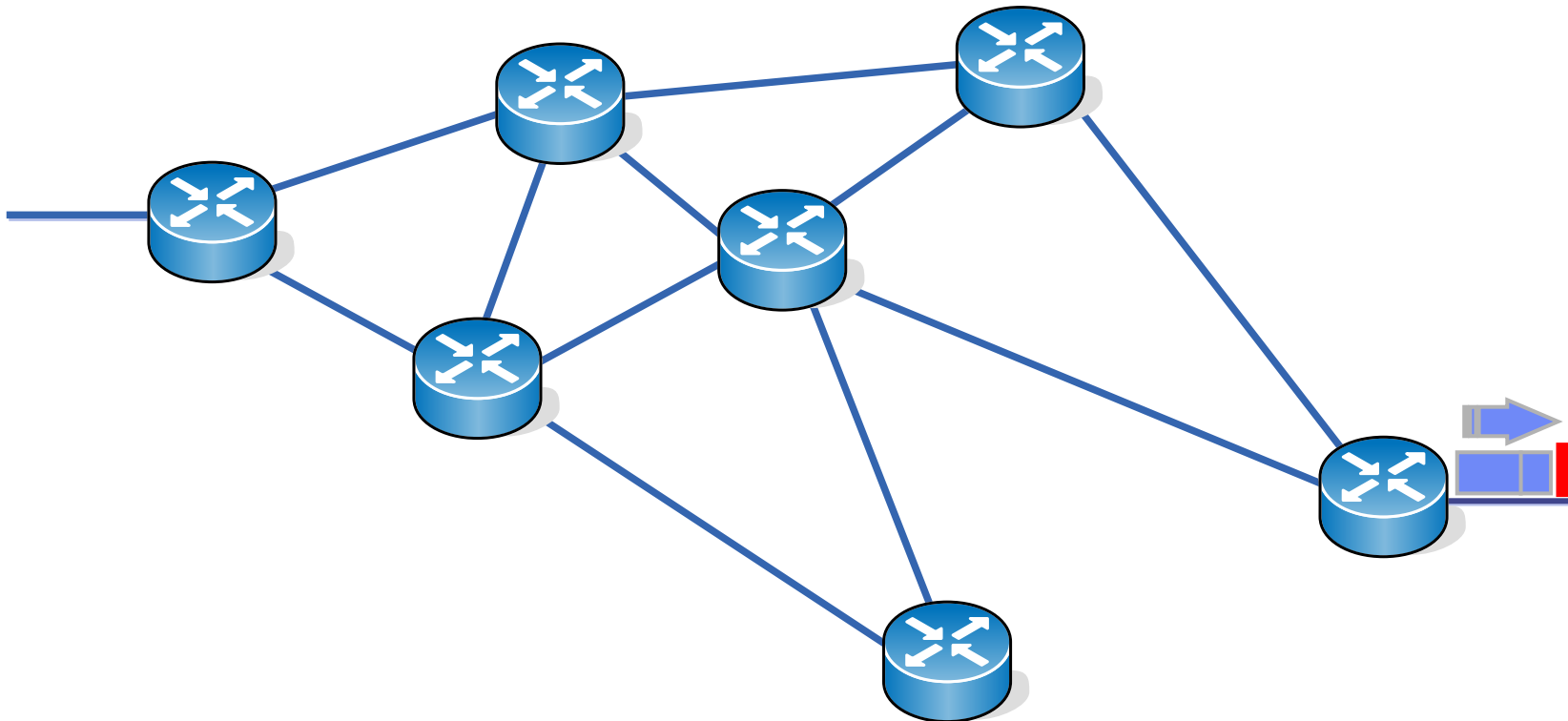
Tablas de forwarding en Circuitos Virtuales



Tablas de forwarding en Circuitos Virtuales



Tablas de forwarding en Circuitos Virtuales



Liberación de Circuitos Virtuales

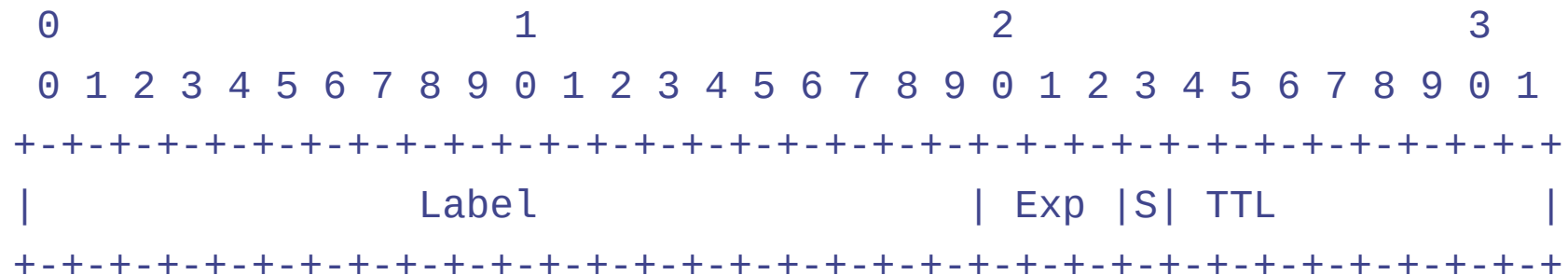
- Cuando no se precisa más, es necesario **liberar el circuito**
 - Para liberar los recursos en los equipos involucrados
- Si el circuito era conmutado (SVC), se requiere señalización (mensajes de control) para liberarlo
- Puede haber liberación por parte de la red (por ejemplo en caso de caída de un extremo)

Comparación datagramas vs Circuitos Virtuales

	Datagramas	Circuitos Virtuales
Establecimiento previo	No	Si
Dirección para forwarding	La del destino final	El identificador de CV (podría tener menos bits que una dirección global porque es de alcance local)
La red almacena el estado de los flujos	No	Si, en la tabla de CVs establecidos
Encaminamiento de paquetes	Se consulta la tabla de forwarding	Se consulta la tabla de CV
Fallas en los enrutadores	Afectan solamente a los paquetes en tránsito	Cortan los Cvs establecidos
Control de Congestión	Difícil	Posible
Calidad de Servicio	Difícil	Posible

Ejemplo: MPLS

- **MPLS** = Multi Protocol Label Switching
- Se verá con más detalle en Redes de Datos 2
- El objetivo es agregar a una red de datagramas como IP las bondades de las redes de circuitos virtuales (Ingeniería de tráfico, Calidad de servicio)
- También brindar más simplicidad y mayor velocidad en el forwarding
- Estandarizado en RFC 3031 y RFC 3032 (y otras RFC con actualizaciones)
- Agrega un pequeño encabezado a los paquetes IP para especificar las etiquetas (identificadores de CV)



Label: Valor de la etiqueta, 20 bits

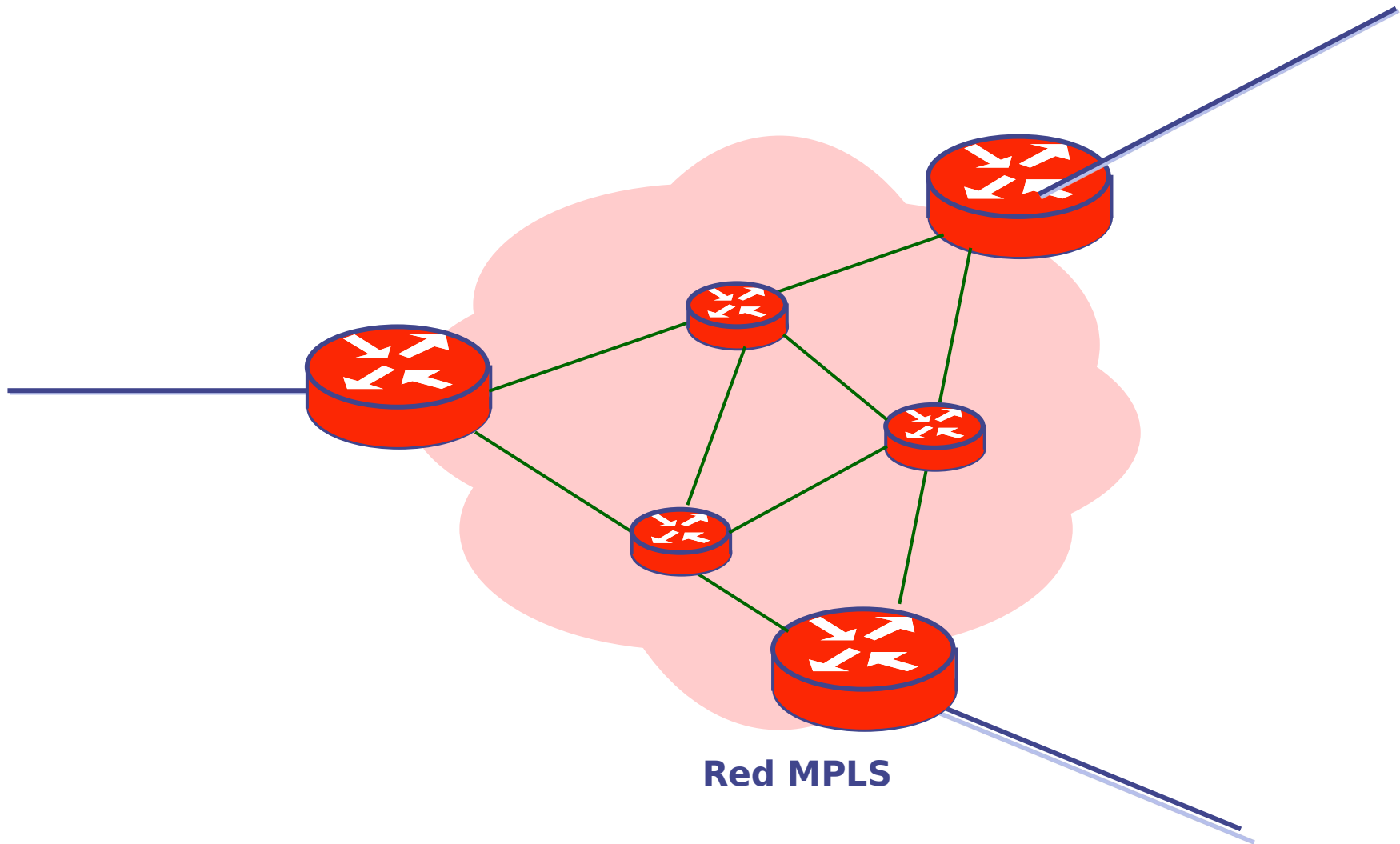
Exp: Calidad de servicio, 3 bits

S: Bottom of Stack, 1 bit

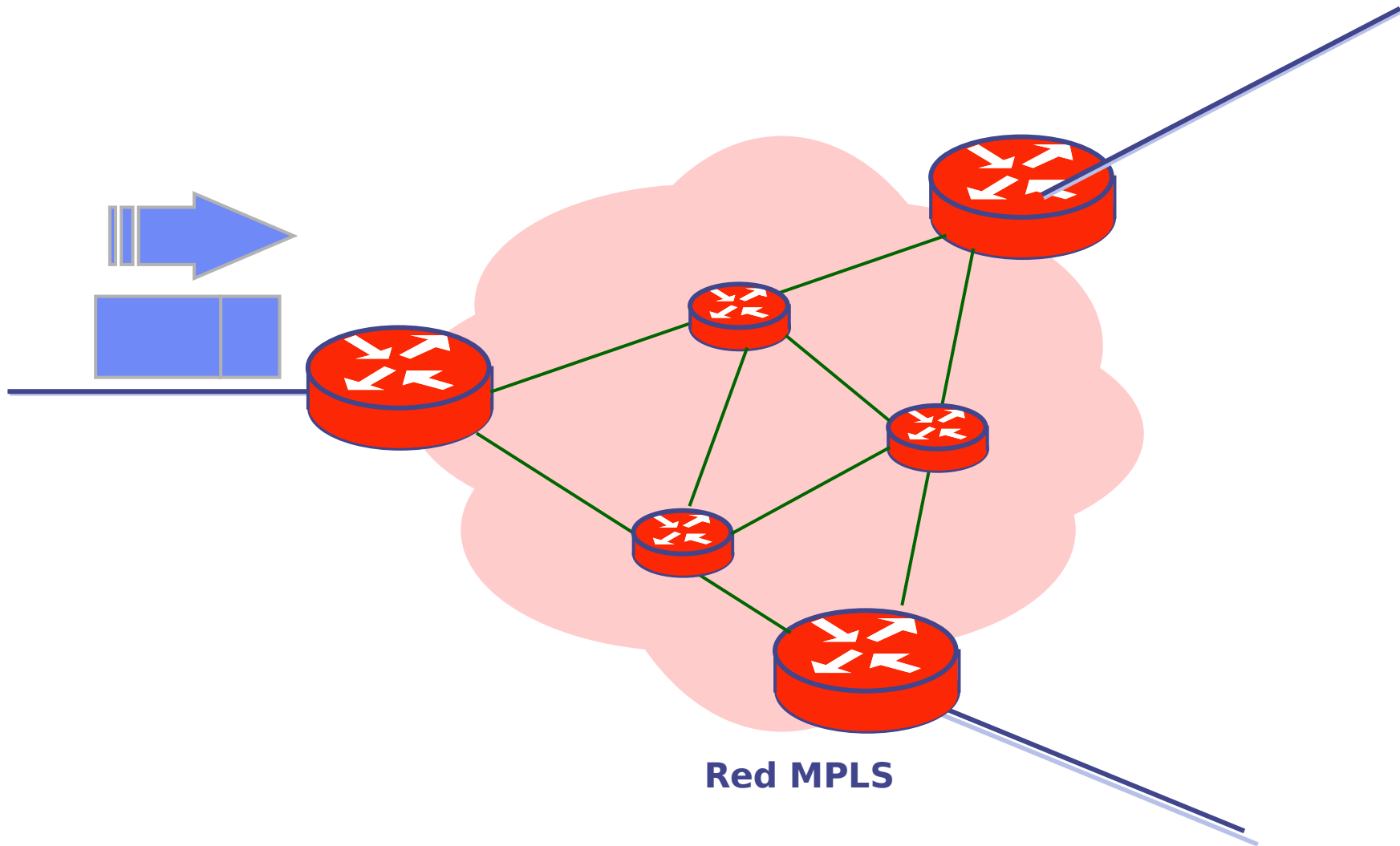
TTL: Time to Live, 8 bits

- Se hace una partición del conjunto de todos los posibles paquetes en clases de equivalencia "**Forwarding Equivalence Classes (FECs)**"
 - Una FEC es un grupo de paquetes que se reenvían de la misma manera
 - En IP sería un rango en la tabla de forwarding
 - Cuando ingresa un paquete a la red MPLS se lo asigna a una FEC
 - Una vez asignados a una FEC los paquetes son indistinguibles desde el punto de vista de su encaminamiento
 - En los equipos intermedios se encamina por las etiquetas MPLS
- El concepto de FEC provee gran **flexibilidad** y escalabilidad
 - Paquetes con distinto destino pueden agruparse en la misma FEC
 - Posibles criterios de FEC
 - IP Destino
 - IP Origen – IP Destino
 - IP Origen - IP Destino - puertos origen y destino
 - Calidad de servicio requerida
 - Etc.

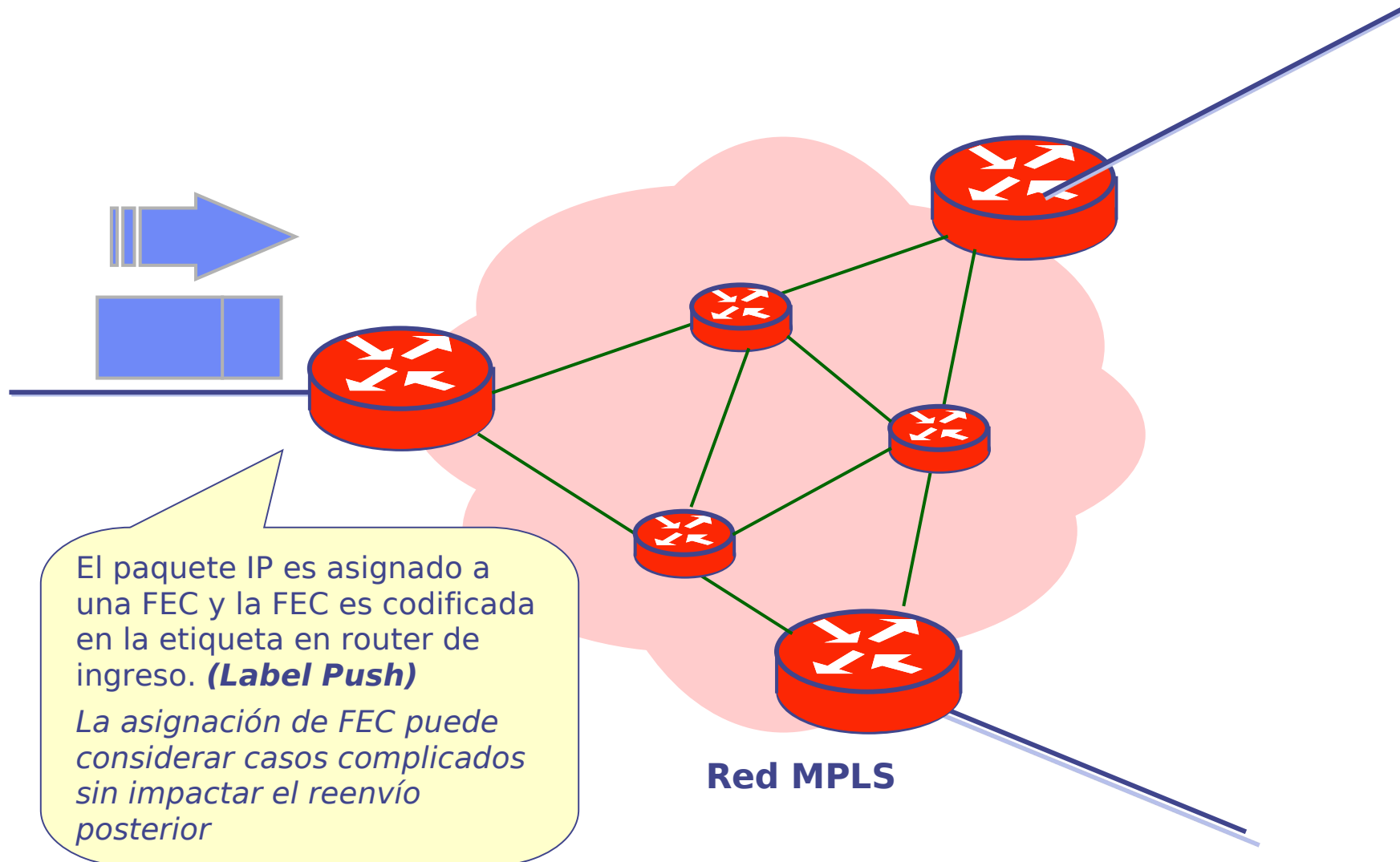
Asignación de FECs y encaminamiento de paquetes



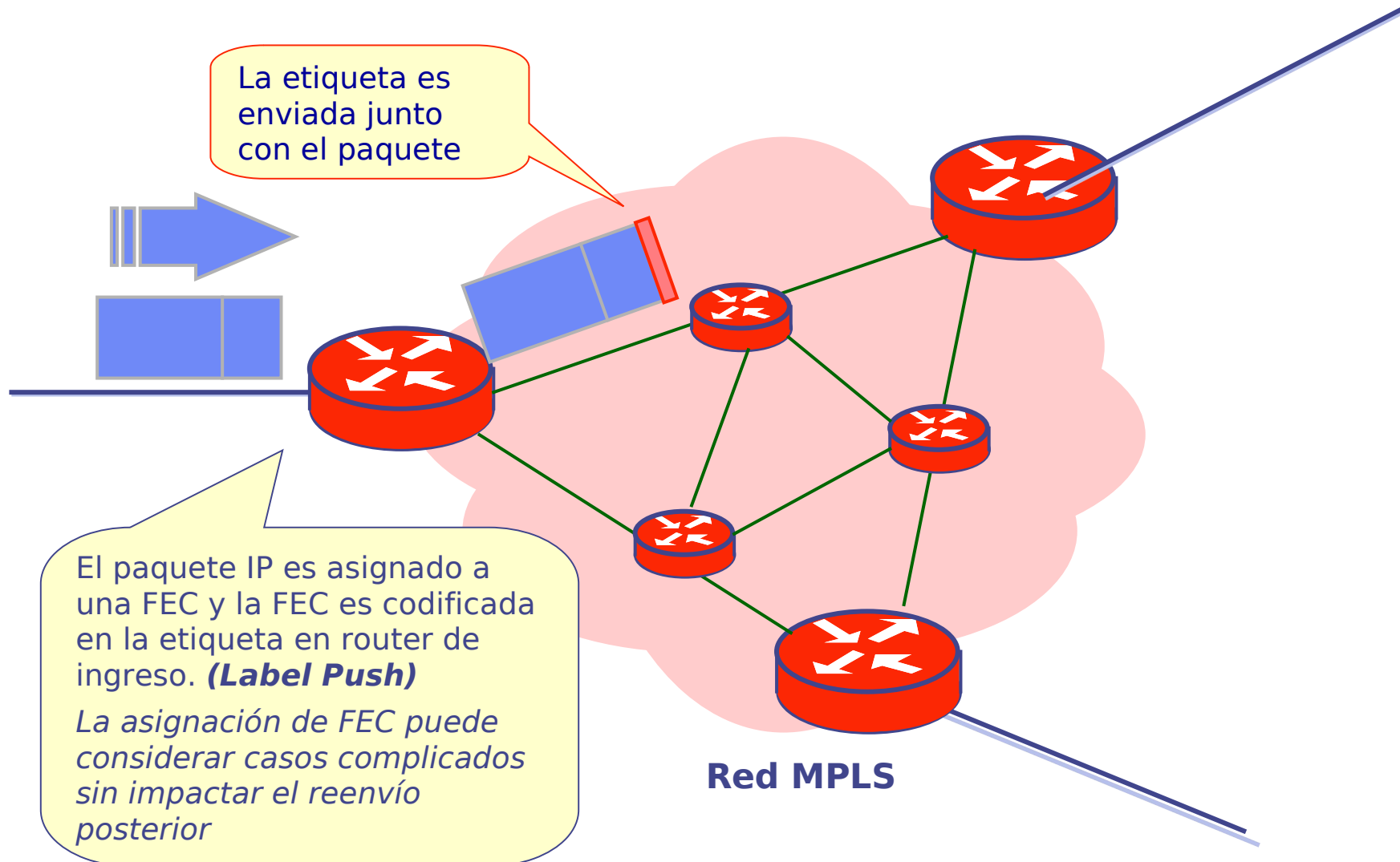
Asignación de FECs y encaminamiento de paquetes



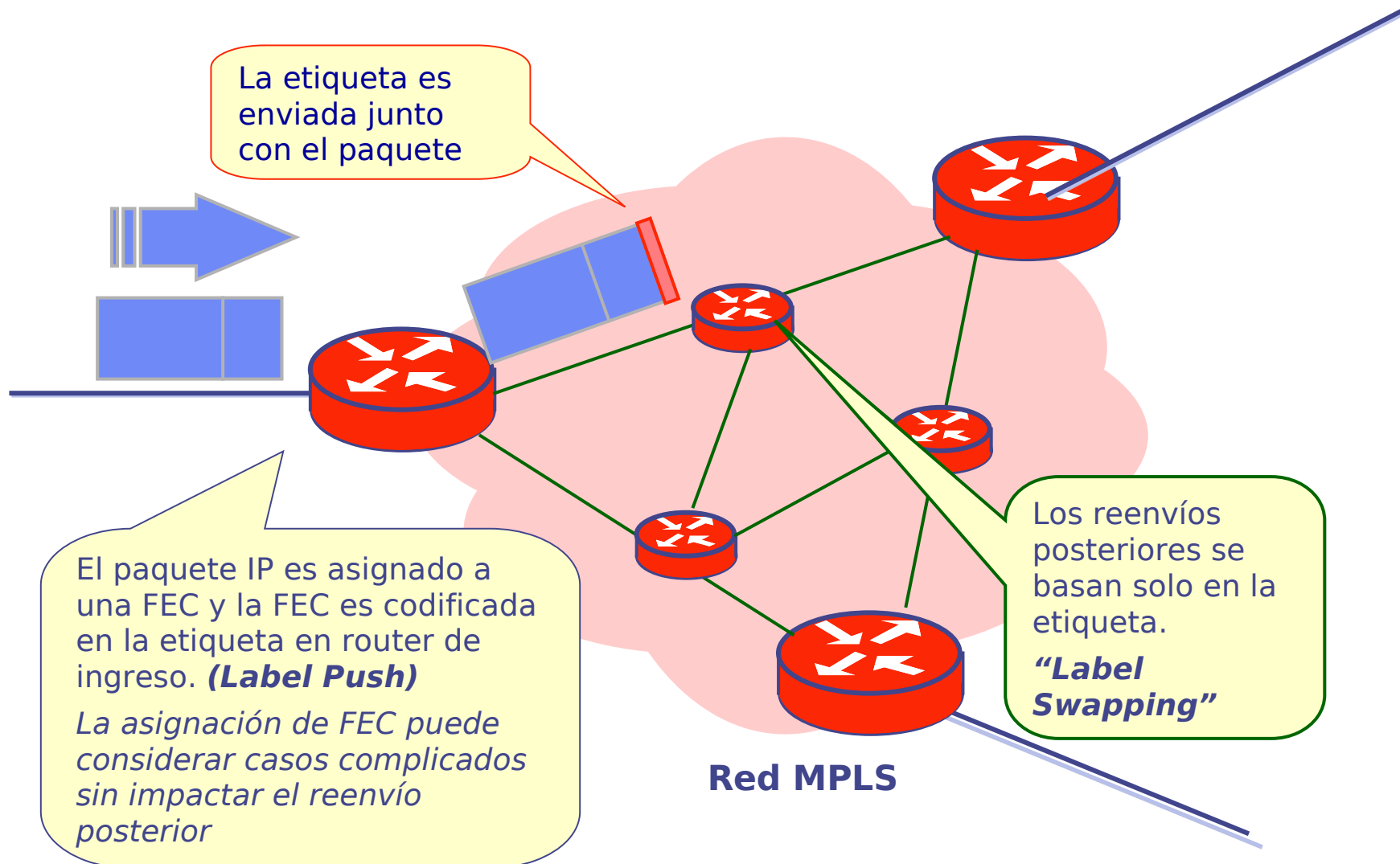
Asignación de FECs y encaminamiento de paquetes



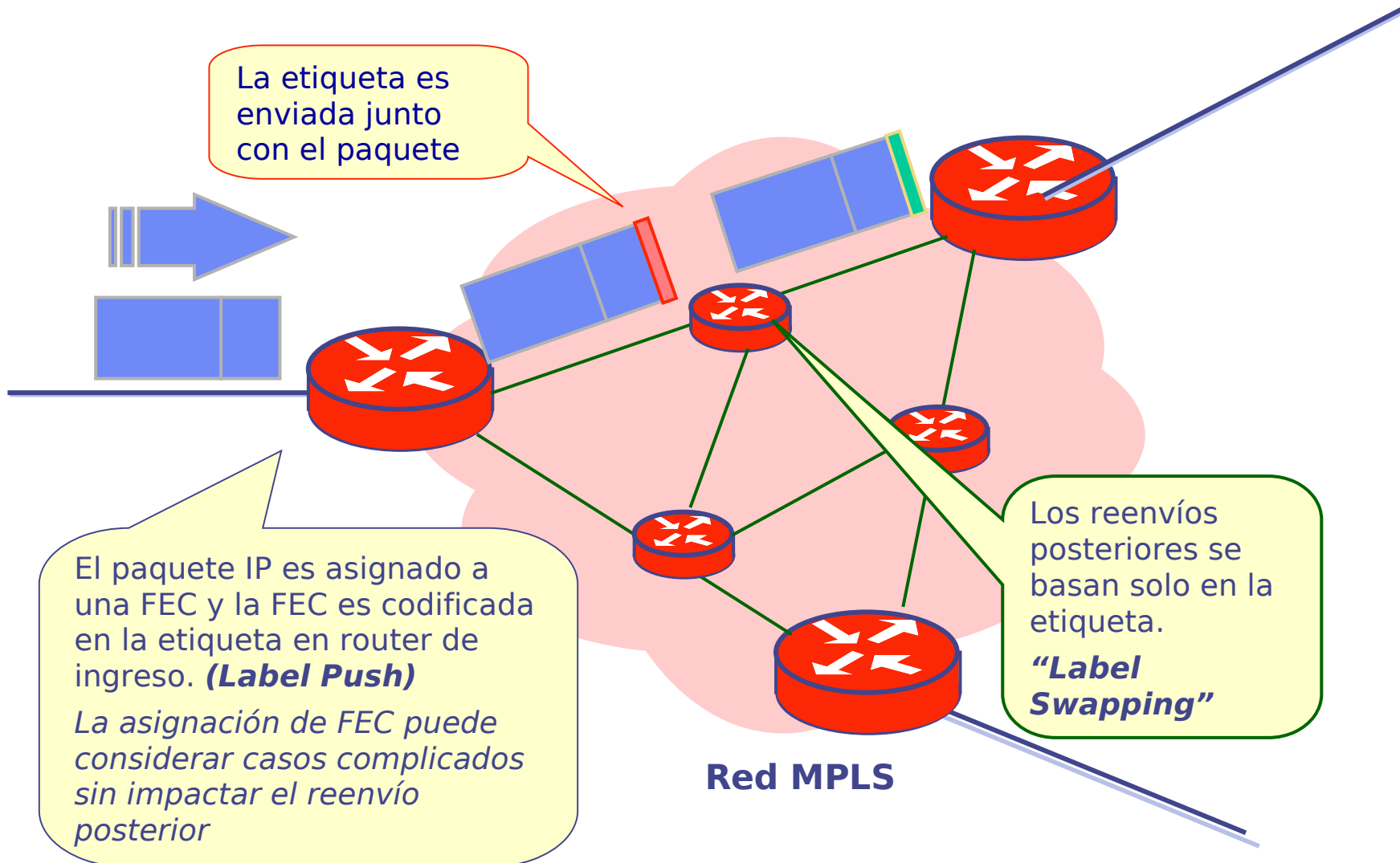
Asignación de FECs y encaminamiento de paquetes



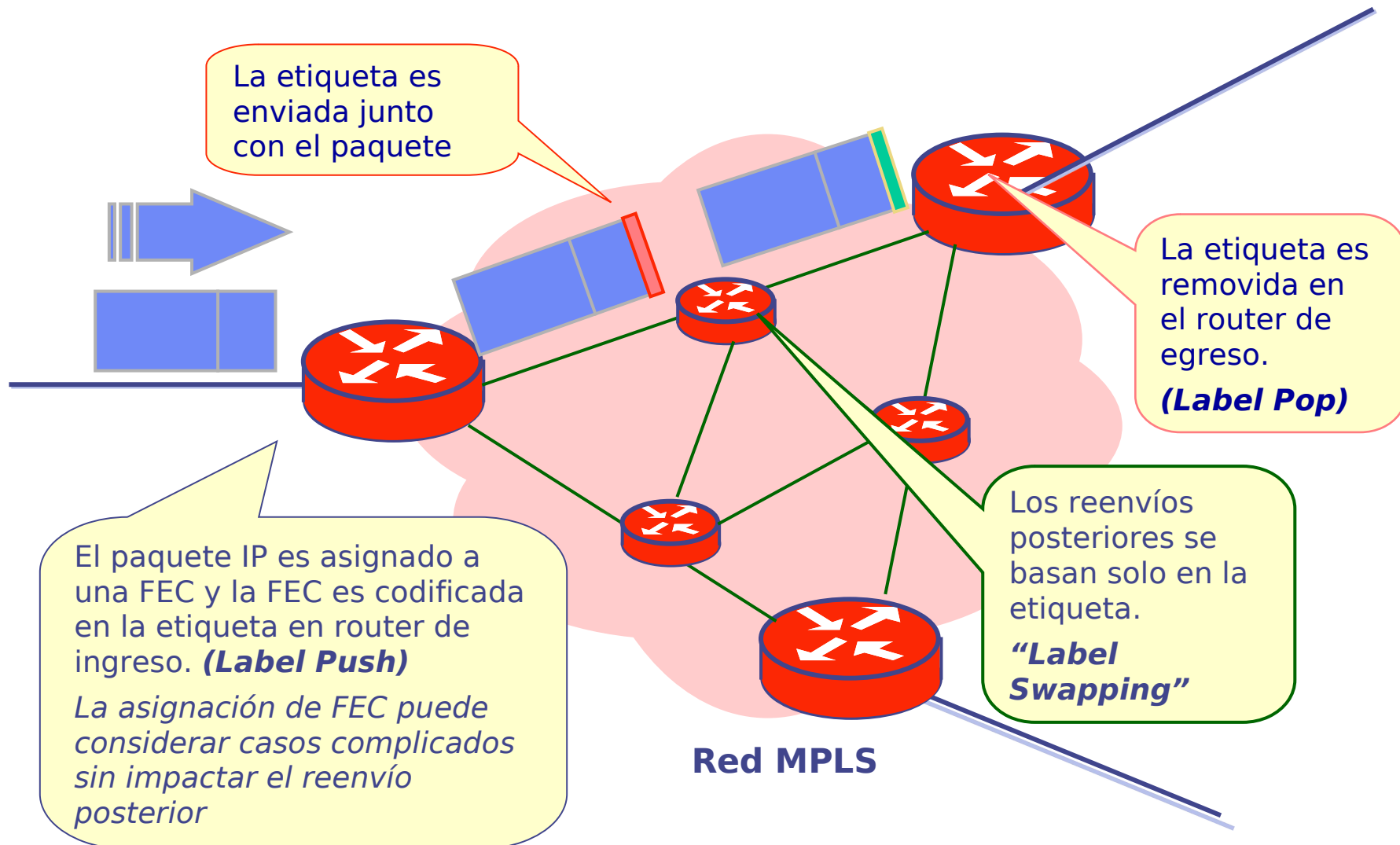
Asignación de FECs y encaminamiento de paquetes



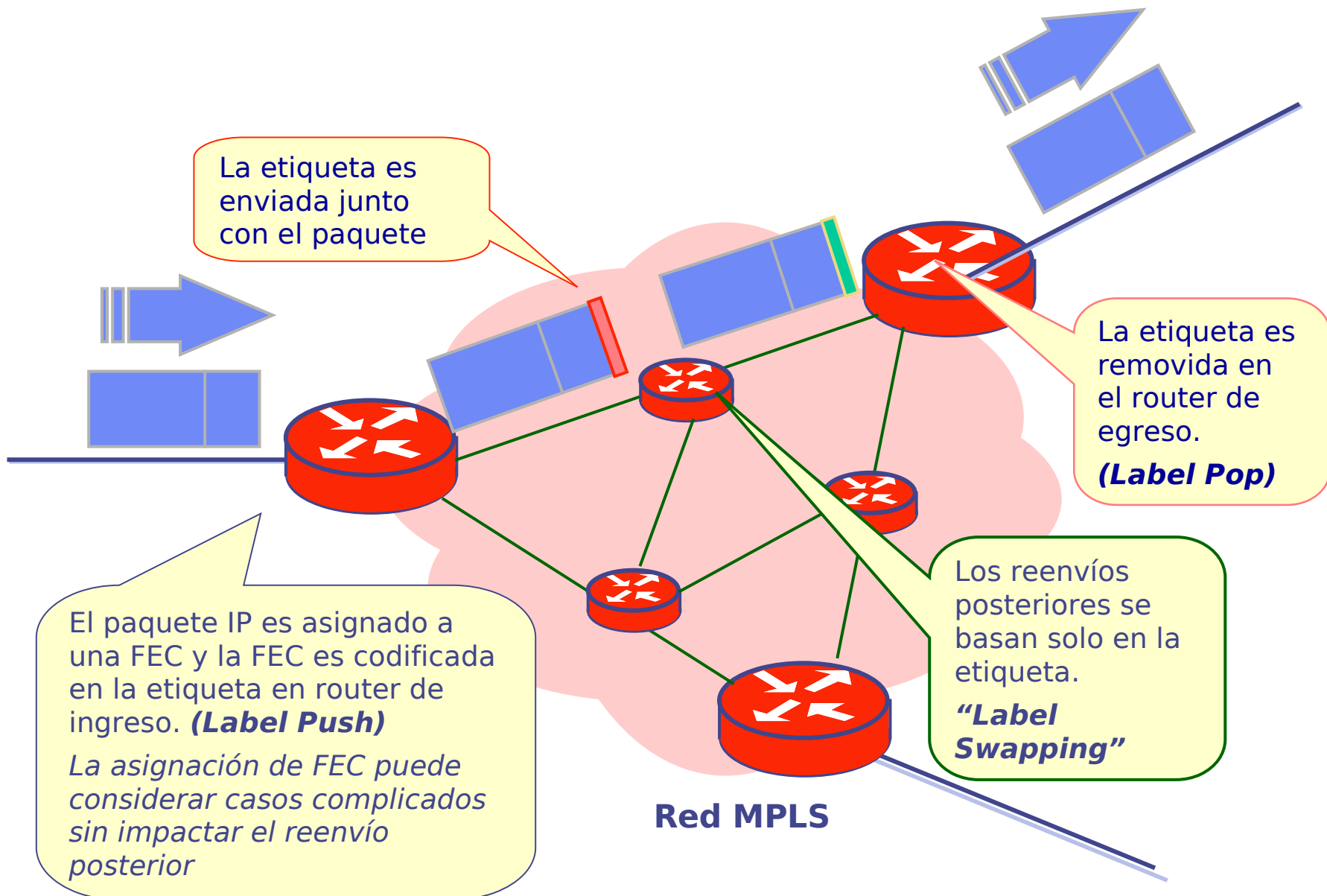
Asignación de FECs y encaminamiento de paquetes



Asignación de FECs y encaminamiento de paquetes



Asignación de FECs y encaminamiento de paquetes

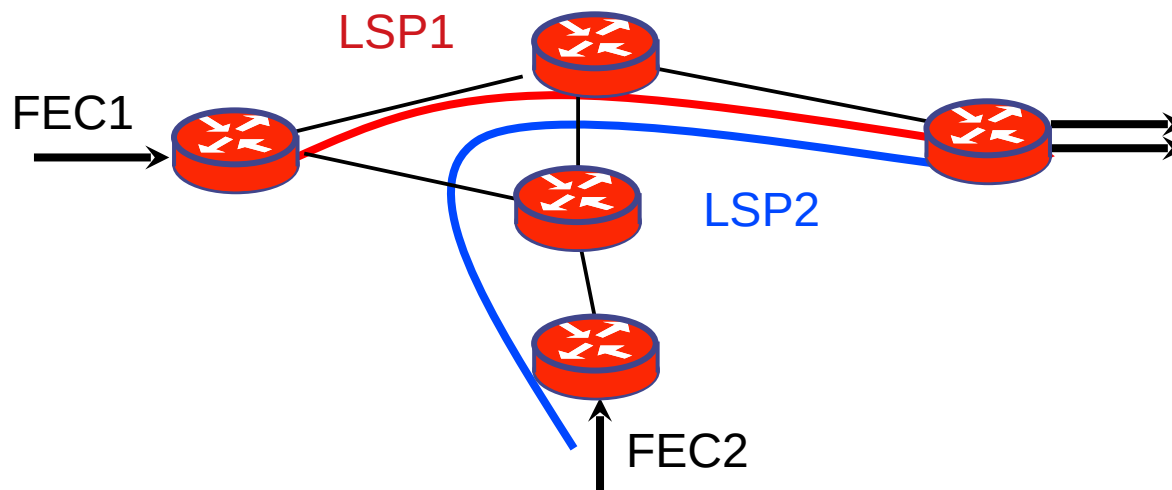


LSP: Label switched path

- En cada router se realiza la siguiente asociación:

Tabla de CVs			
Interfaz de entrada	Etiqueta de entrada	Interfaz de salida	Etiqueta de salida
*	2000	o4	18
.....			

- De esta forma se construye para cada FEC un túnel, llamado Label Switched Path (**LSP**)



Protocolos de distribución de etiquetas

- Un protocolo de distribución de etiquetas es un conjunto de procedimientos por los cuales un router MPLS informa a otro de las asociaciones FEC - Etiqueta que ha realizado
- Algunos Protocolos propuestos:
 - LDP Specification (RFC 3036)
 - Carrying Label Information in BGP-4 (RFC 3107)
 - RSVP-TE: Extensions to RSVP for LSP Tunnels (RFC 3209)
 - Constraint-Based LSP Setup using LDP (RFC 3212)
- La arquitectura no asume que exista un único protocolo funcionando

Usos típicos de MPLS

- VPNs (Virtual Private Networks)
 - Por ejemplo un cliente tiene una red con puntos de presencia en varios departamentos (sucursales) y lo ven como una red privada
 - Varios clientes pueden ver “su” red de forma independiente de la red de otros clientes
- Ingeniería de tráfico
 - Elegir caminos para los paquetes con criterios diferentes a la dirección de destino que es lo que hace el “Longest Prefix Match”