

# **BLOQUE IV.**

## **Nivel de enlace de datos**

---

**PUNTES Y CONMUTADORES.**



# Contenidos

---

1. Introducción
  1. Funciones de un protocolo de enlace de datos
2. Entramado
  1. Protocolos orientados a carácter
  2. Protocolos orientados a bit
3. Corrección de errores
  1. Códigos de control de errores
  2. Códigos polinómicos
4. Técnicas de control de flujo y protocolos de control de errores

# Contenidos

---

5. Protocolos de Control de Acceso al Medio
6. Direccionamiento
7. Hubs, Puentes y Conmutadores
8. Protocolo STP
9. Ejemplo de protocolos de nivel de enlace de datos
  1. HDLC
  2. PPP

# Contenidos

---

5. Protocolos de Control de Acceso al Medio

6. Direccionamiento

7. Hubs, Puentes y Conmutadores

8. Protocolo STP

9. Ejemplo de protocolos de nivel de enlace de datos

1. HDLC

2. PPP

# Contenidos

---

5. Protocolos de Control de Acceso al Medio

6. Direccionamiento

7. Hubs, Puentes y Conmutadores

1. Hubs

2. Puentes y conmutadores

1. Tabla de direcciones

2. Filtrado, reenvío e inundación

3. Procesos a realizar por un conmutador y sus prioridades

4. Cut-Through vs. Store-and-Forward

8. Protocolo STP

9. Ejemplo de protocolos de nivel de enlace de datos

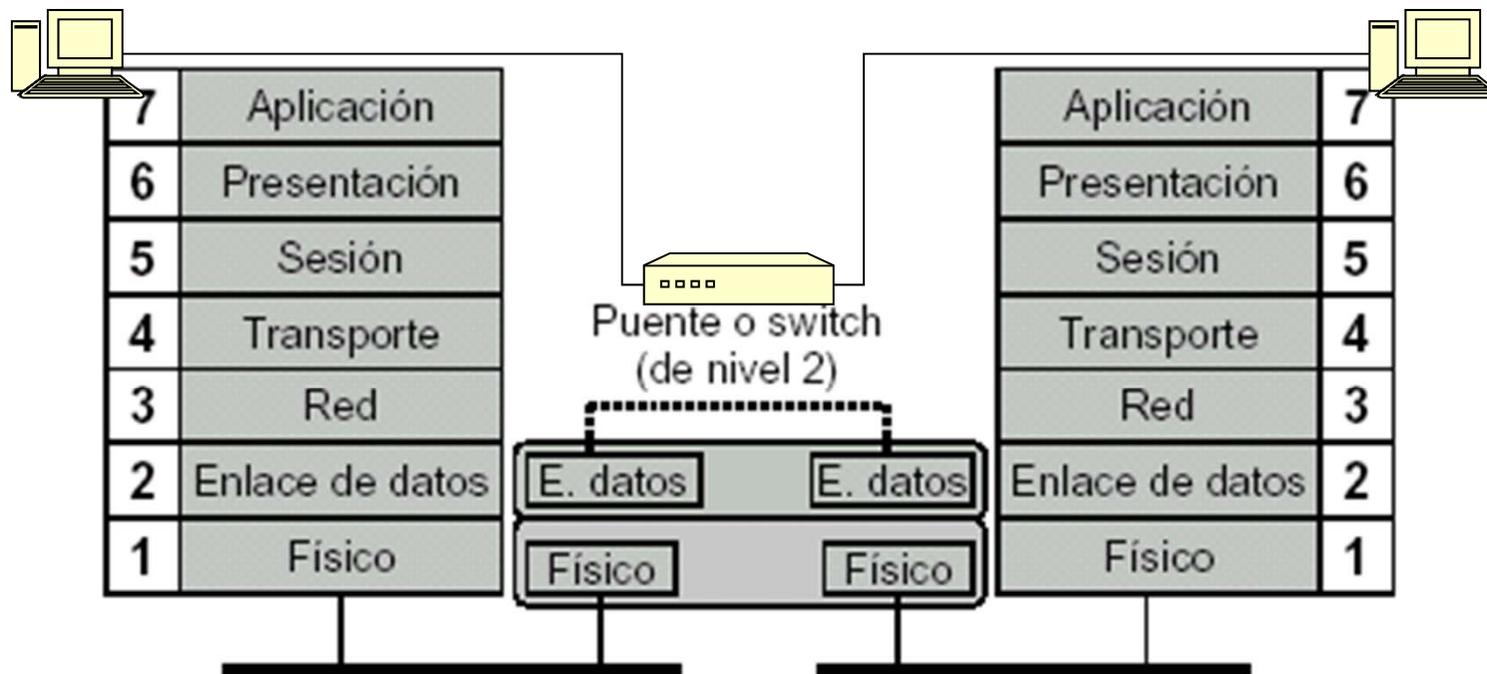
## 7.2 Puentes y conmutadores

---

- La diferencia entre puentes y conmutadores es más una cuestión de *marketing* que una cuestión técnica
  - Las funciones de un puente y un conmutador son exactamente las mismas
- Antes de los 90
  - La mayor parte de los puentes raras veces tenían más de dos puertos
  - Sus prestaciones estaban limitadas tanto por el *software* como por el *hardware* disponible en aquel momento
- A partir de entonces
  - Aparición de nuevos procesadores, memorias y ASIC, permitió la fabricación de dispositivos con mucha más capacidad y un número elevado de puertos
  - Estos nuevos puentes con mejores prestaciones se denominaron conmutadores pero sus funciones seguían siendo exactamente las mismas

## 7.2 Puentes y conmutadores

- Al contrario que los *hubs* ⇒ los puentes (*bridges*) y conmutadores (*switch*) son **dispositivos de capa 2** (operan sobre las tramas)

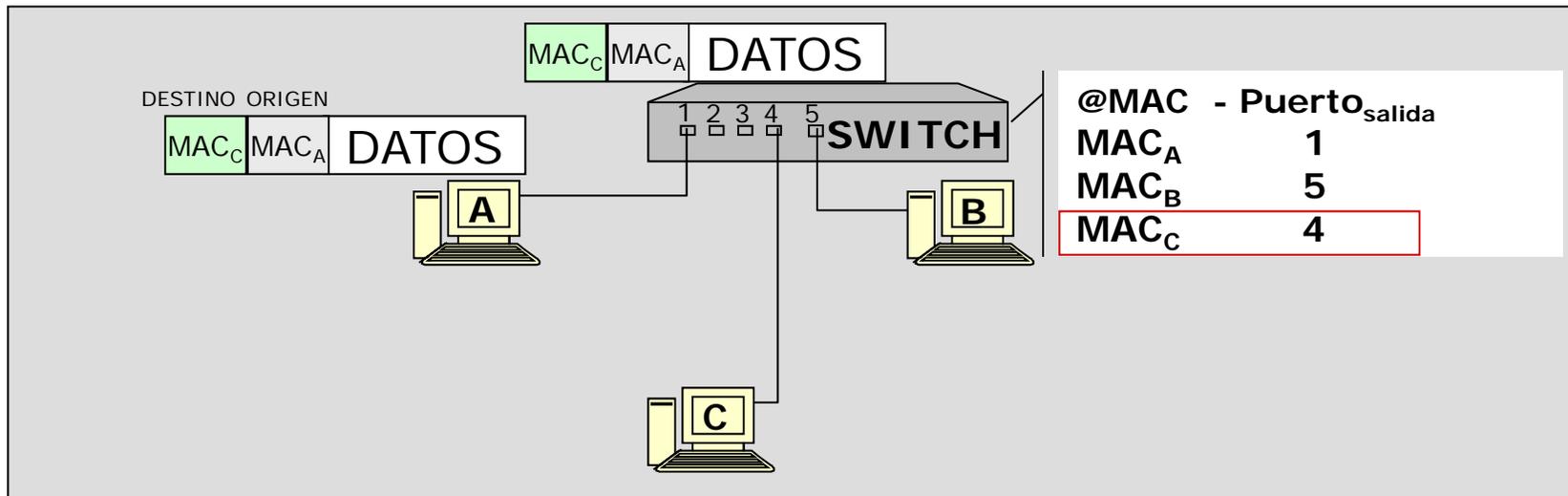
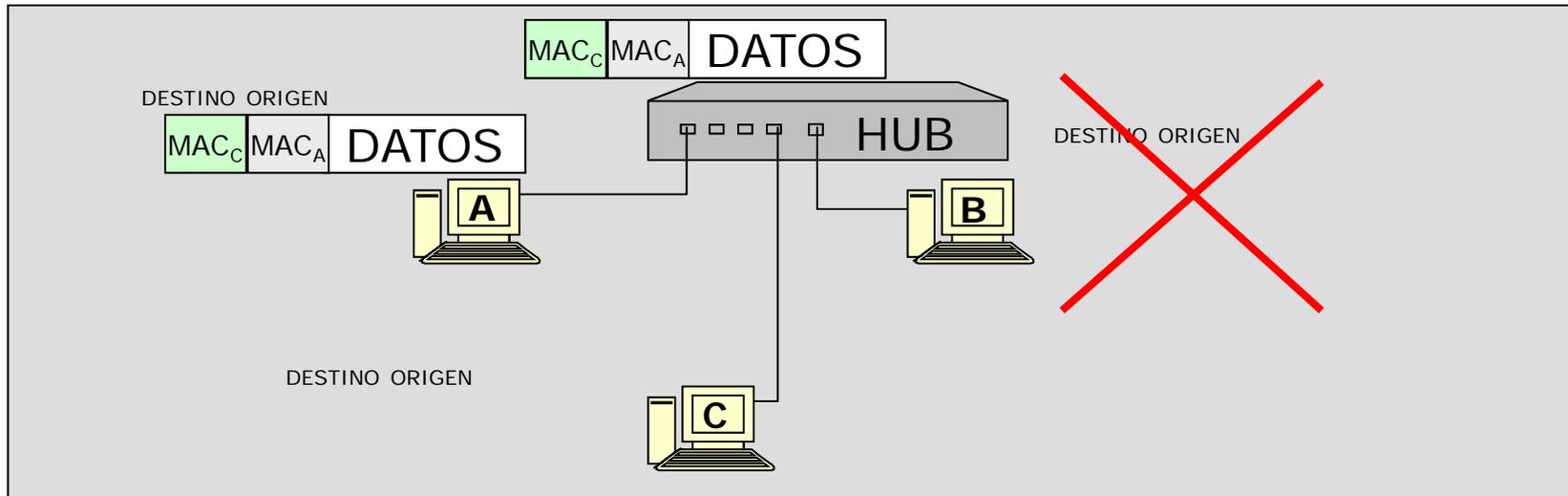


## 7.2 Puentes y conmutadores

---

- Cuando una trama llega a un puerto de un conmutador
  - El conmutador no se limita a copiar la trama en todas las demás interfaces como haría un *hub*
  - Al contrario, **examina la dirección física destino de la trama** e intenta reenviarla sólo hacia el puerto de salida más adecuado
- Para tomar esta decisión es necesario que el puente mantenga una tabla en la que se relacionen las direcciones de las interfaces de red y los puertos del puente

# 7.2 Puentes y conmutadores



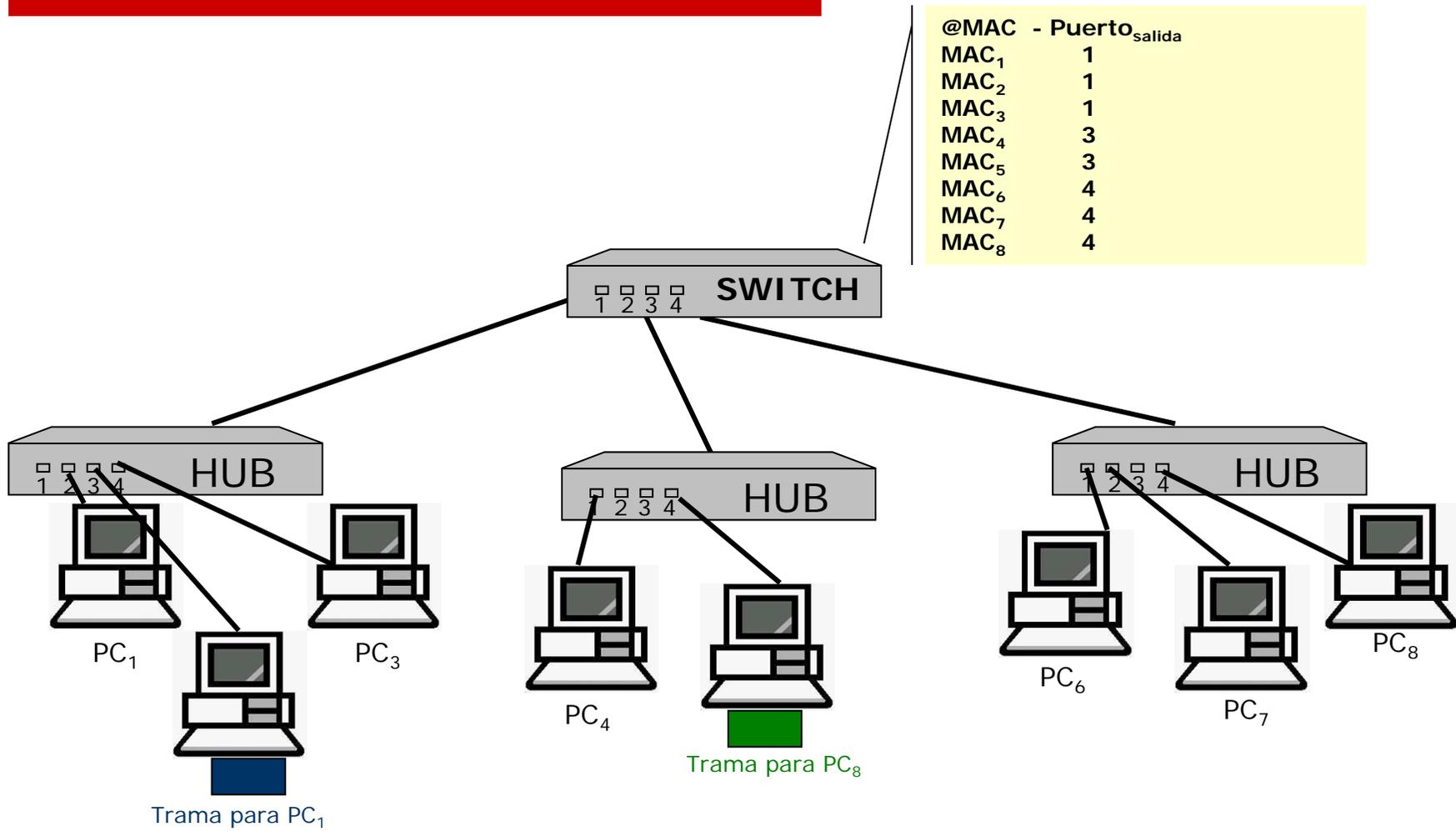
## 7.2 Puentes y conmutadores

---

### □ Dominio de colisión

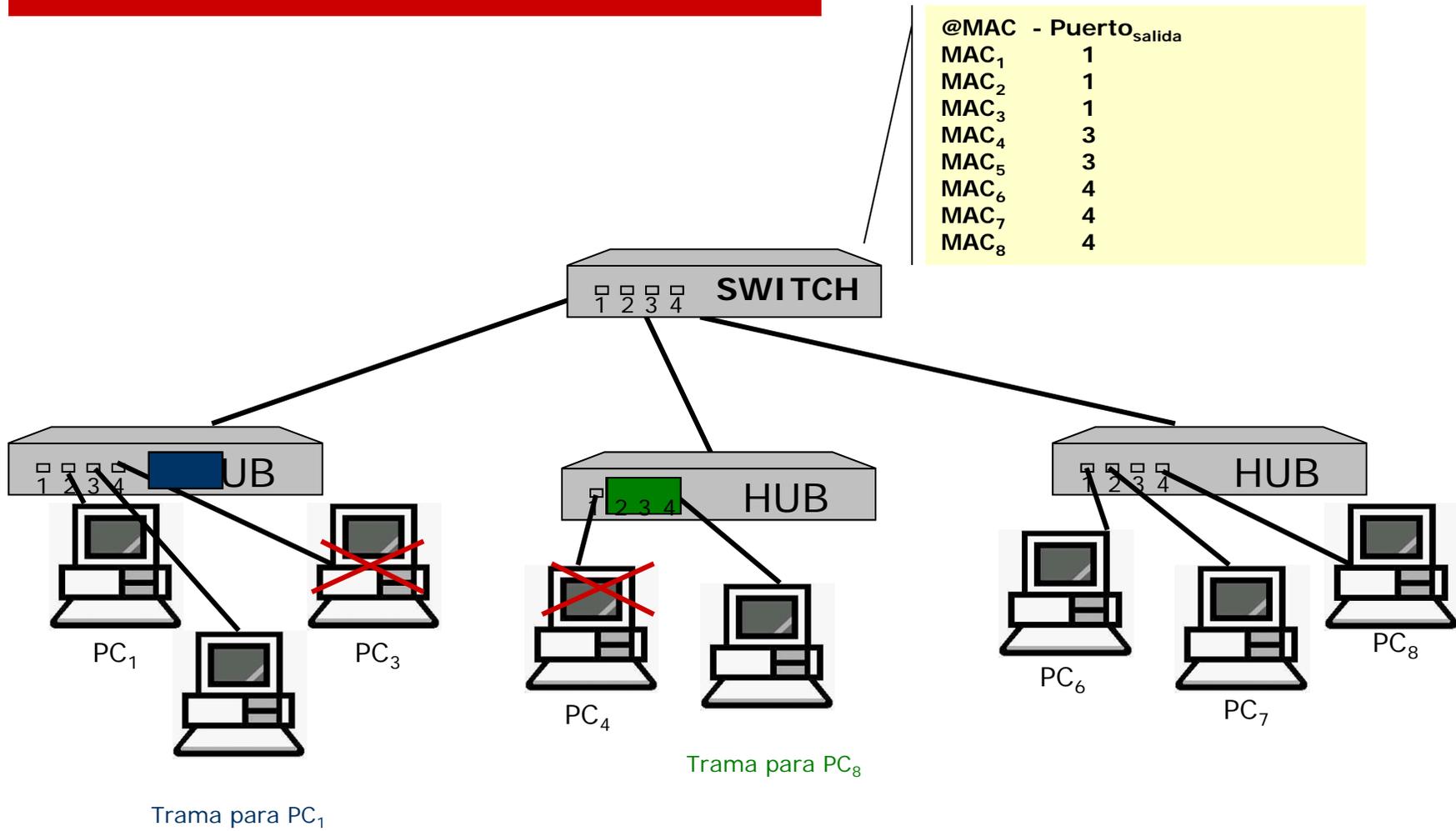
- Los dominios de colisión son los segmentos de red física conectados donde pueden ocurrir colisiones ⇒ causan que la red sea ineficiente
- Cada vez que ocurre una colisión en la red se detienen todas las transmisiones por un período de tiempo
  - La duración de este período sin transmisión varía y depende de un algoritmo de postergación para cada dispositivo de la red
- Si ↑ colisiones ⇒ prestaciones ↓
- Los tipos de dispositivos que interconectan los segmentos de red definen los dominios de colisión
  - **Los dispositivos de nivel de enlace de datos dividen los dominios de colisión**
  - **Permiten que haya más de una transmisión a la vez por la red**

# 7.2 Puentes y conmutadores



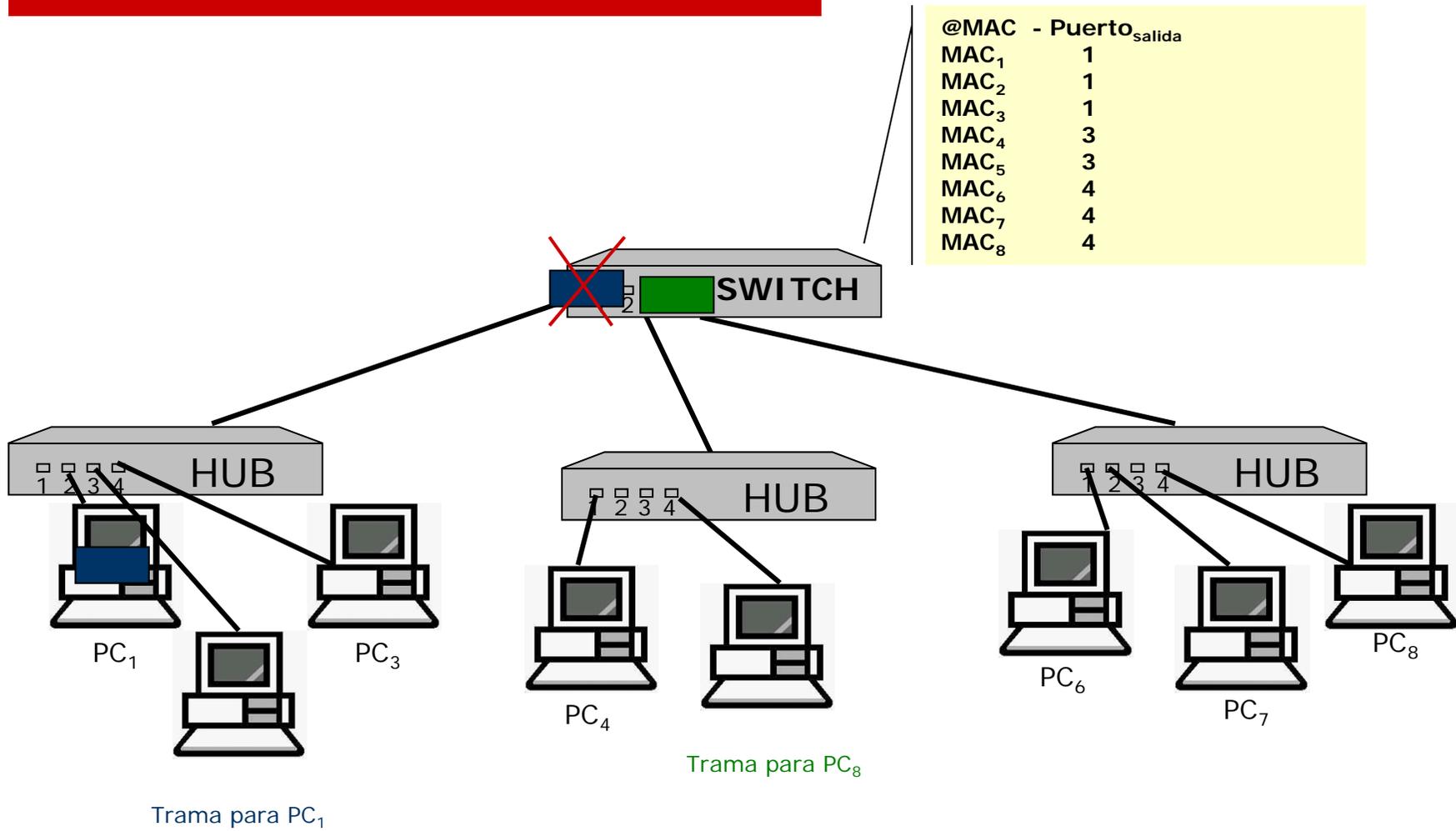
Al haber más dominios de colisión (más pequeños), el número de colisiones se reduce

# 7.2 Puentes y conmutadores



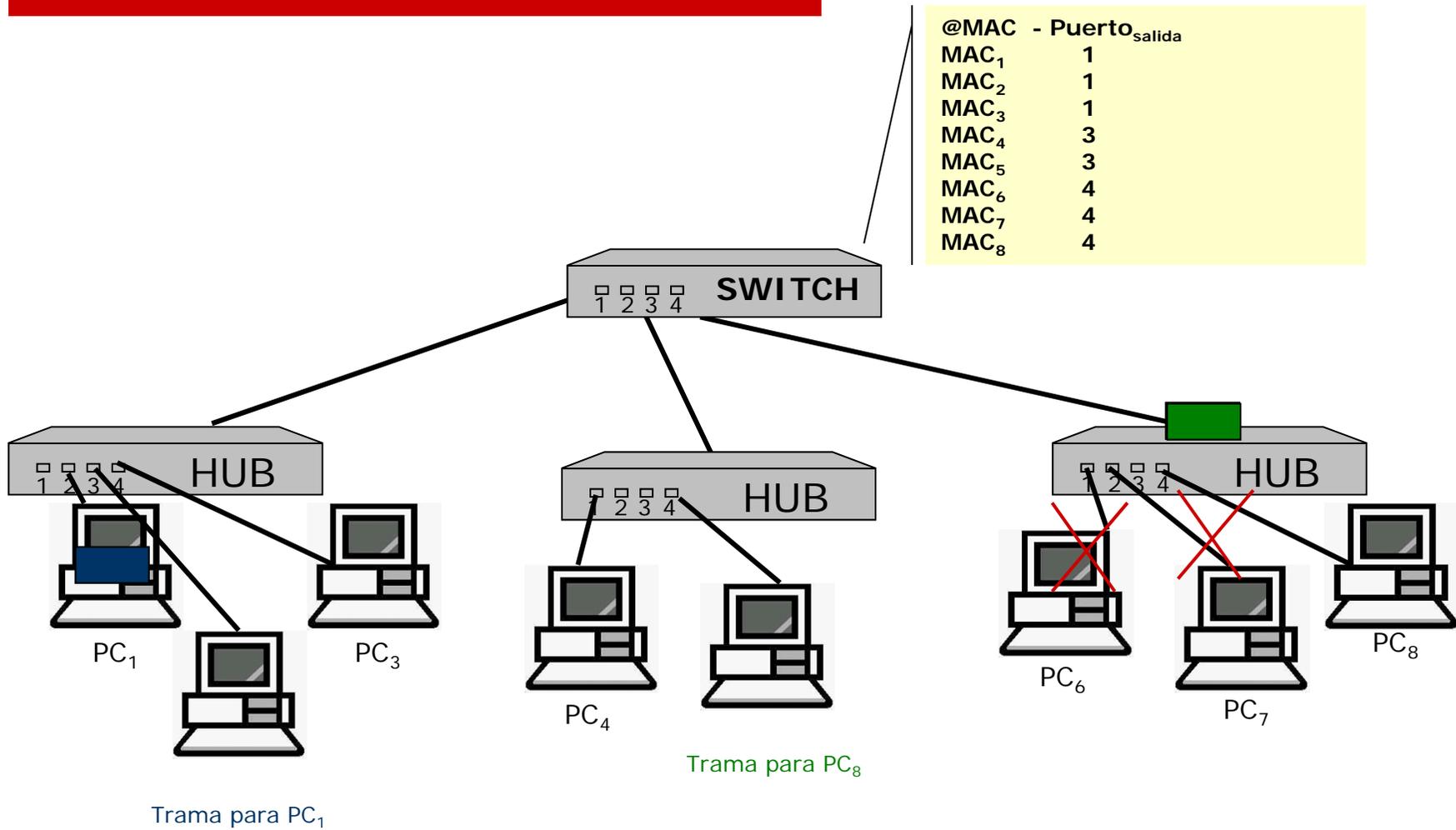
Al haber más dominios de colisión (más pequeños), el número de colisiones se reduce

# 7.2 Puentes y conmutadores



Al haber más dominios de colisión (más pequeños), el número de colisiones se reduce

# 7.2 Puentes y conmutadores

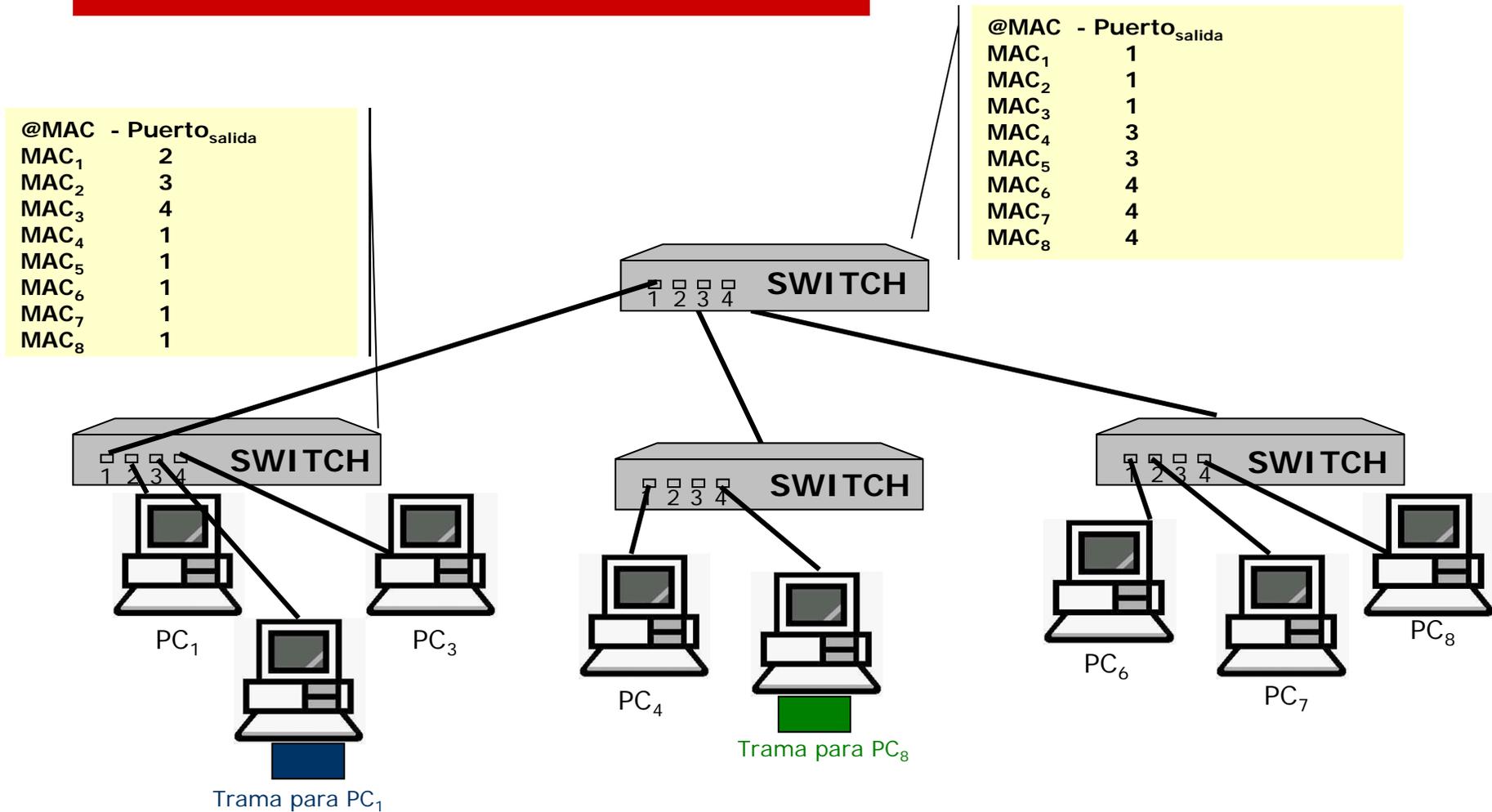


## 7.2 Puentes y conmutadores

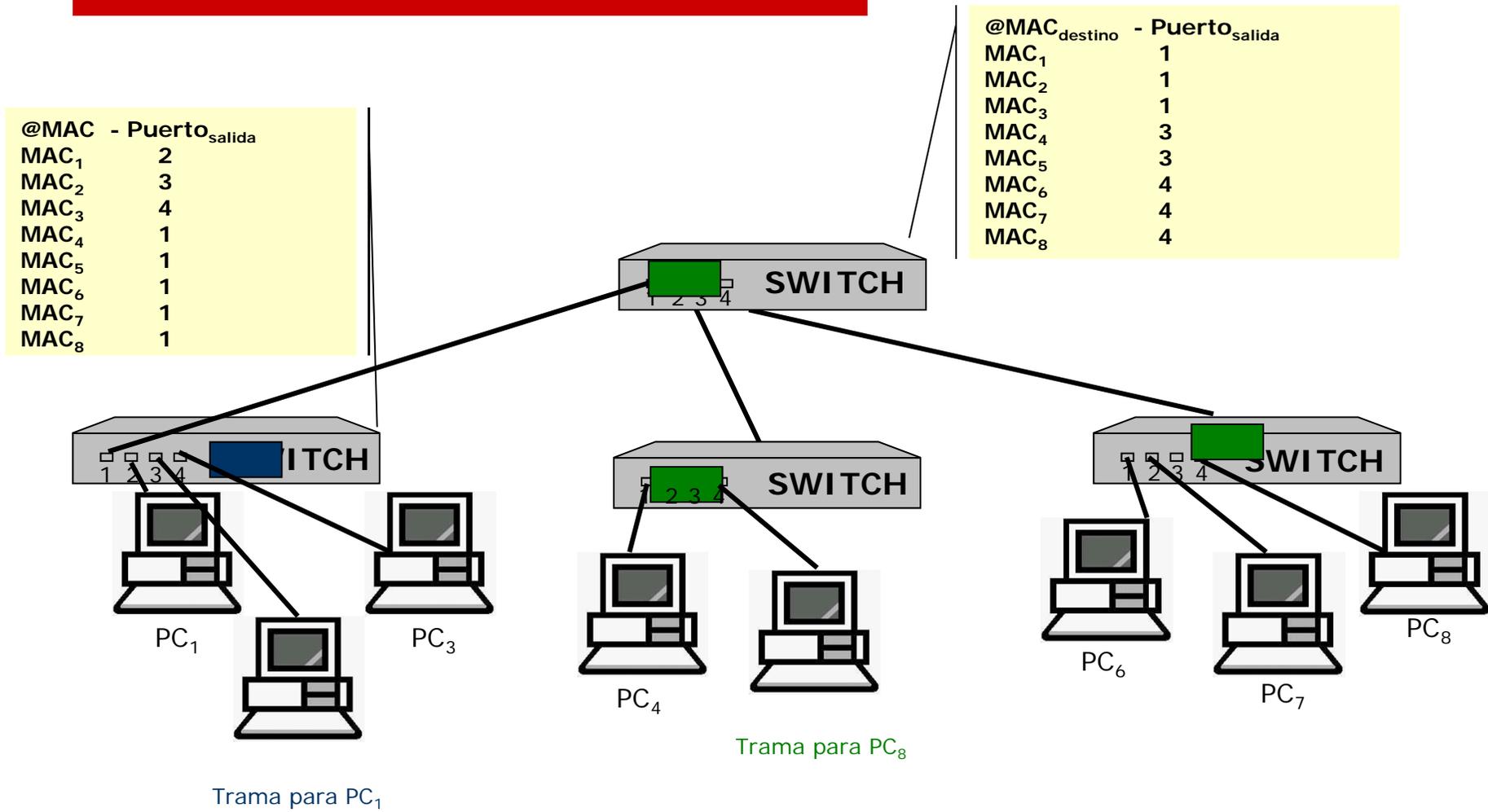
---

- Cuantos más conmutadores, más dominios de colisión (más pequeños)
  - Tener muchos dominios de colisión es mejor que tener pocos

# 7.2 Puentes y conmutadores



# 7.2 Puentes y conmutadores



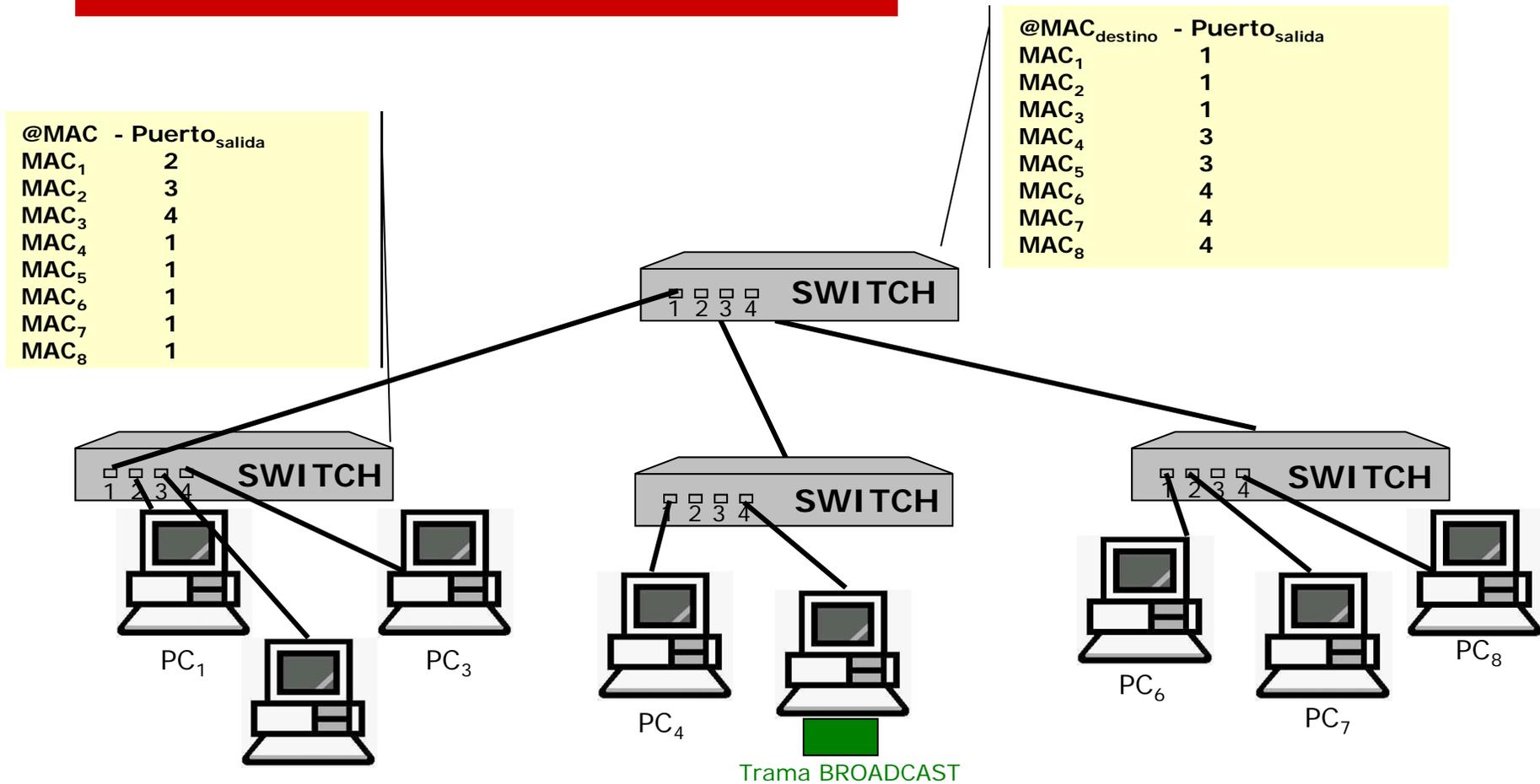
## 7.2 Puentes y conmutadores

---

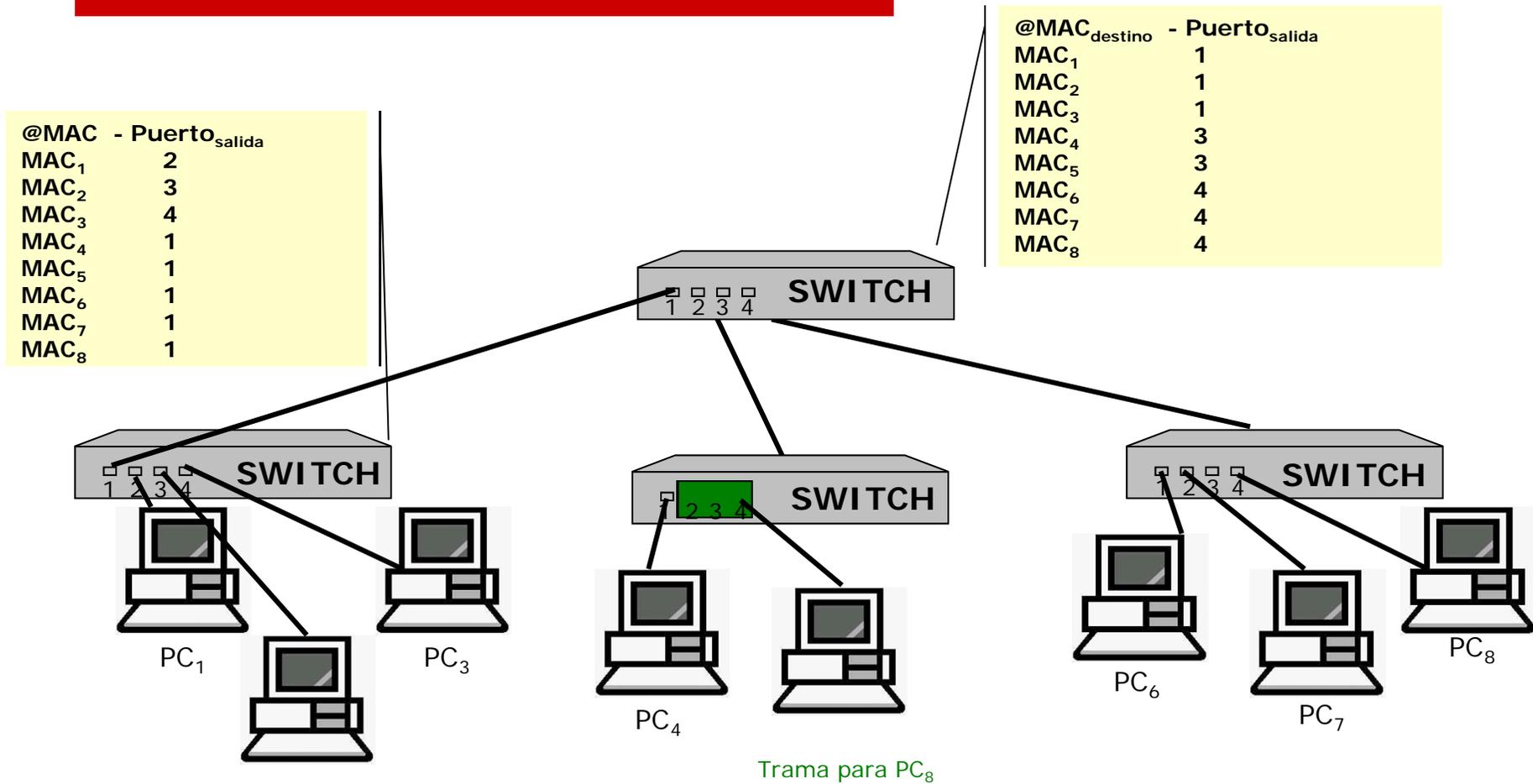
### □ Dominio de *broadcast*

- Los dominios de broadcast son los segmentos de red física conectados donde se recibirá un mismo mensaje de tipo *broadcast*
- Cuando un nodo necesita comunicarse con todos los equipos de la misma red, envía una trama de *broadcast* con una dirección MAC destino FF-FF-FF-FF-FF-FF

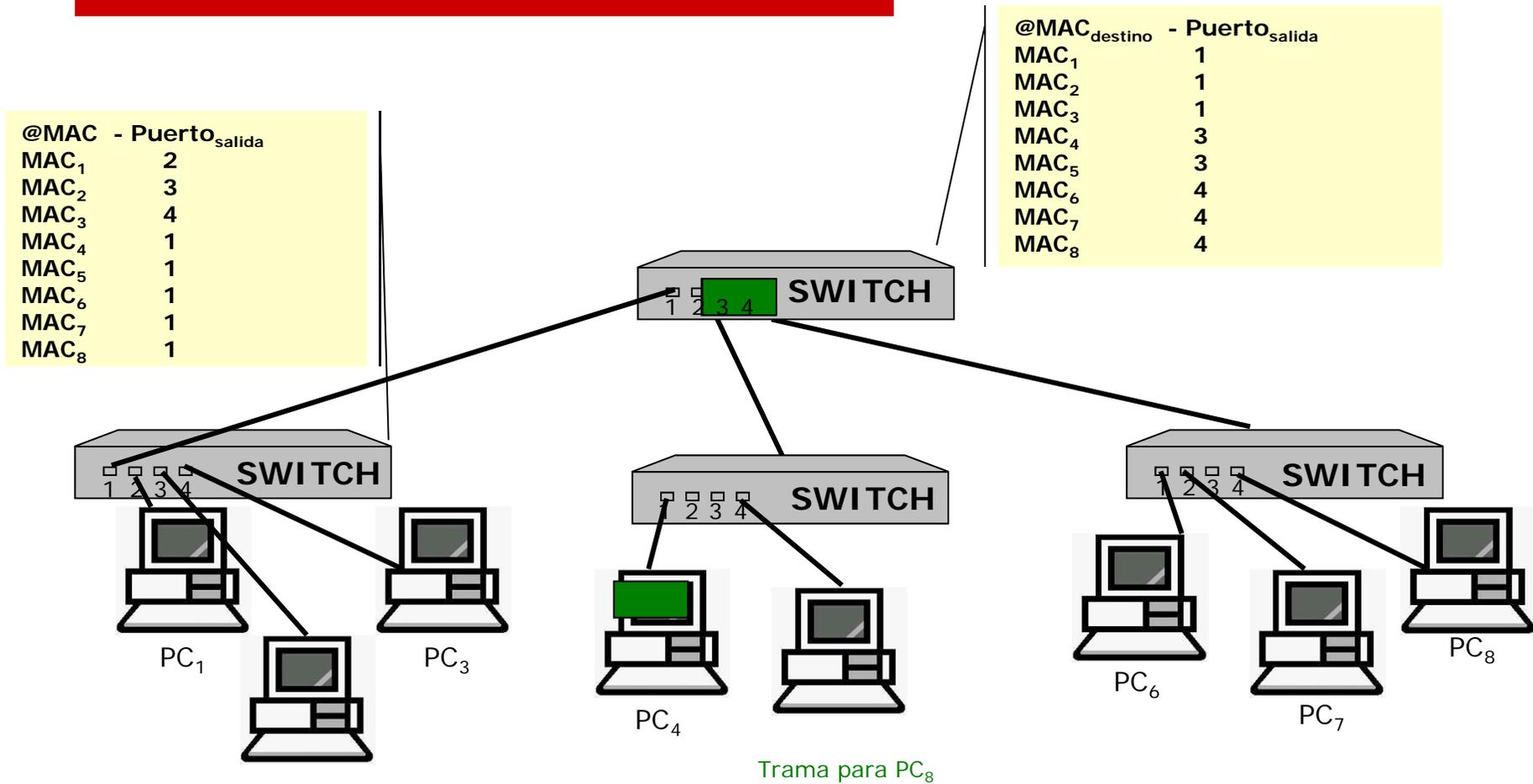
# 7.2 Puentes y conmutadores



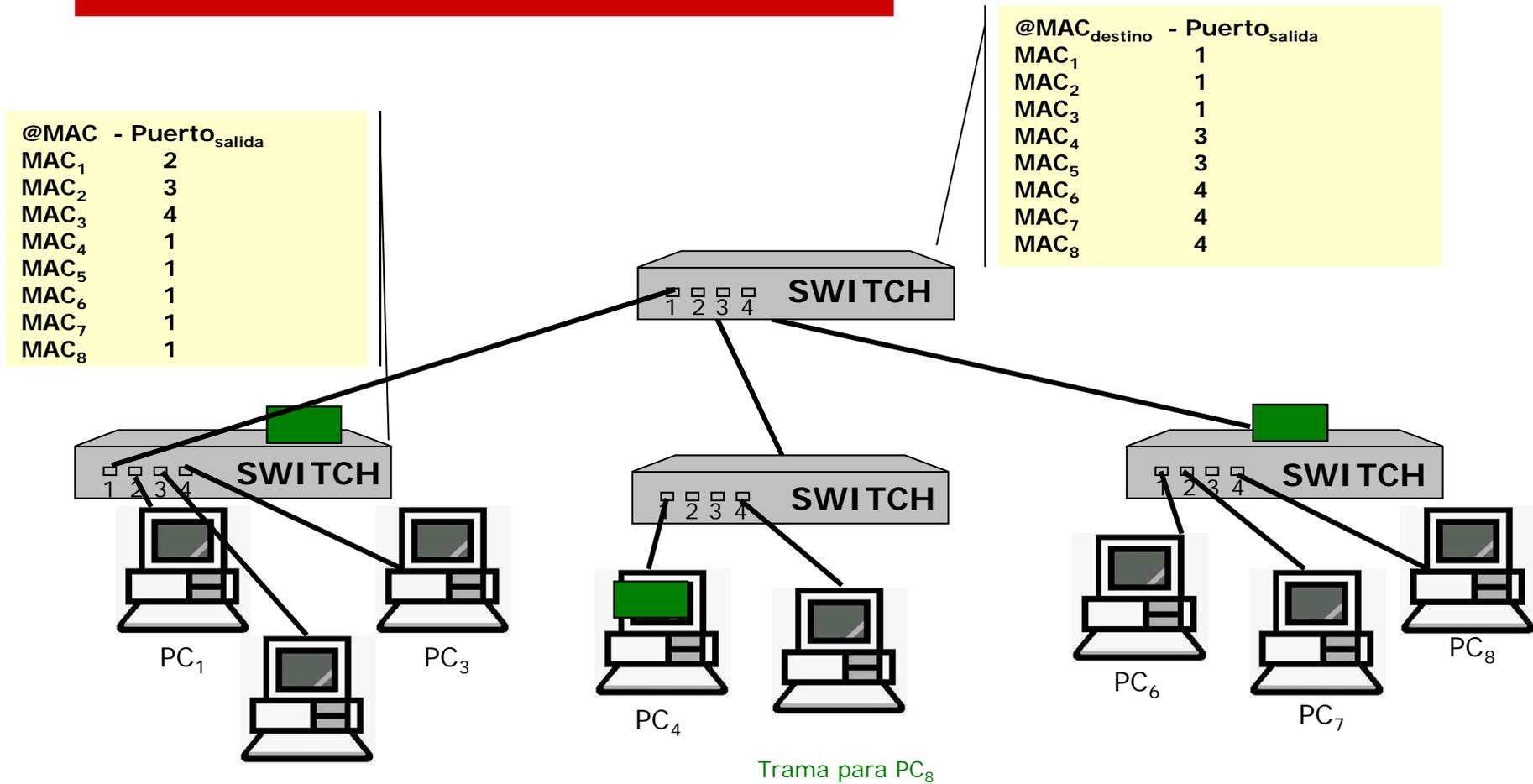
# 7.2 Puentes y conmutadores



# 7.2 Puentes y conmutadores



# 7.2 Puentes y conmutadores



...un único dominio de *broadcast*...

# Contenidos

---

5. Protocolos de Control de Acceso al Medio

6. Direccionamiento

7. Hubs, Puentes y Conmutadores

1. Hubs

2. Puentes y conmutadores

1. Tabla de direcciones

2. Filtrado, reenvío e inundación

3. Procesos a realizar por un conmutador y sus prioridades

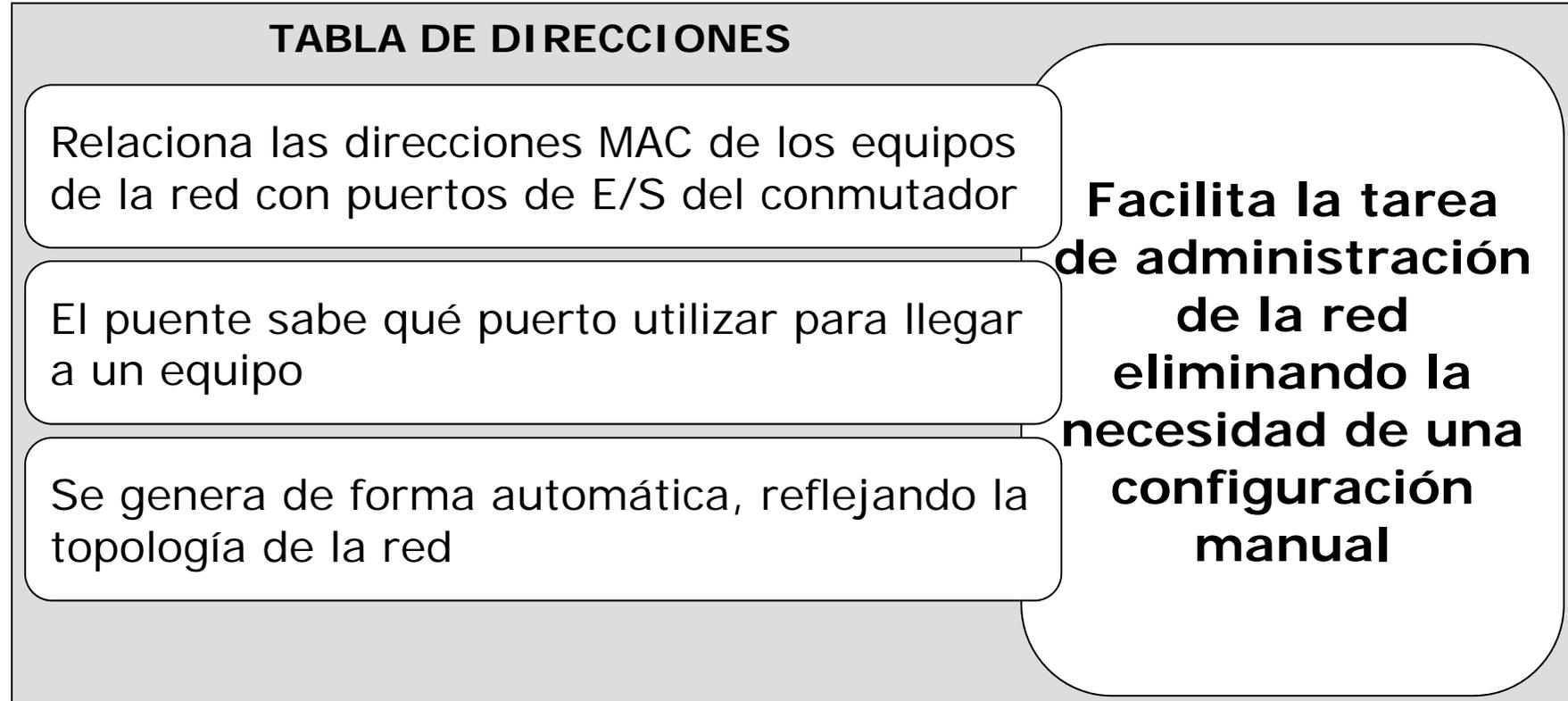
4. Cut-Through vs. Store-and-Forward

8. Protocolo STP

9. Ejemplo de protocolos de nivel de enlace de datos: HDLC

## 7.2.1 Tabla de direcciones

---



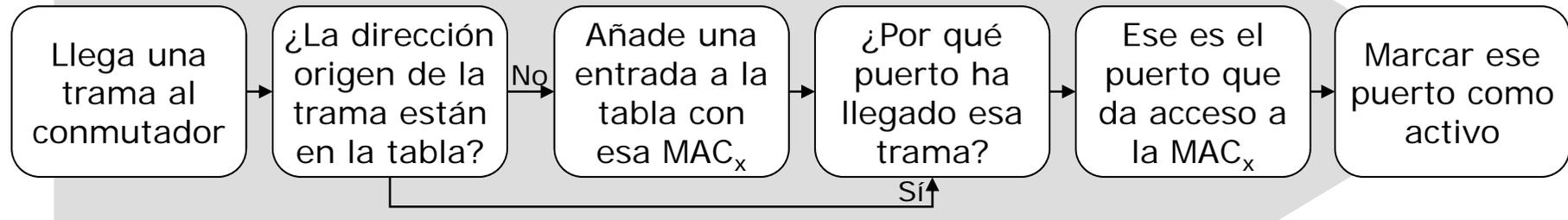
## 7.2.1 Tabla de direcciones

---

- Un puente crea las entradas de la tabla de direcciones en función de la dirección origen de las tramas recibidas
  - Cuando se recibe una trama de A por el puerto 1  $\Rightarrow$  el host A se añade a la tabla de direcciones indicando que está accesible a través de dicho puerto
  - A este proceso se le denomina **aprendizaje**
  - Tanto las entradas en la tabla como la marca de actividad se construyen en función del emisor NUNCA del receptor de las tramas
  - Es un proceso continuo

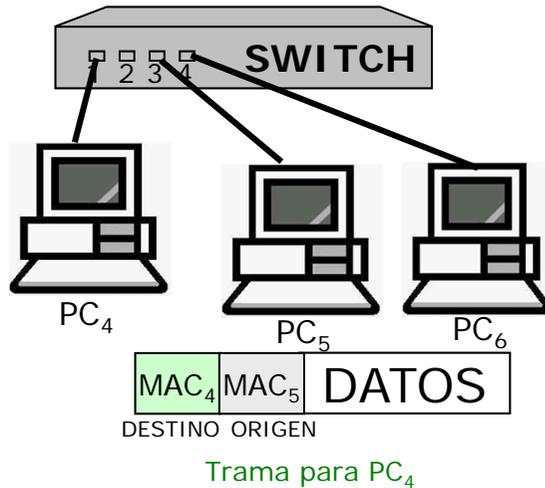
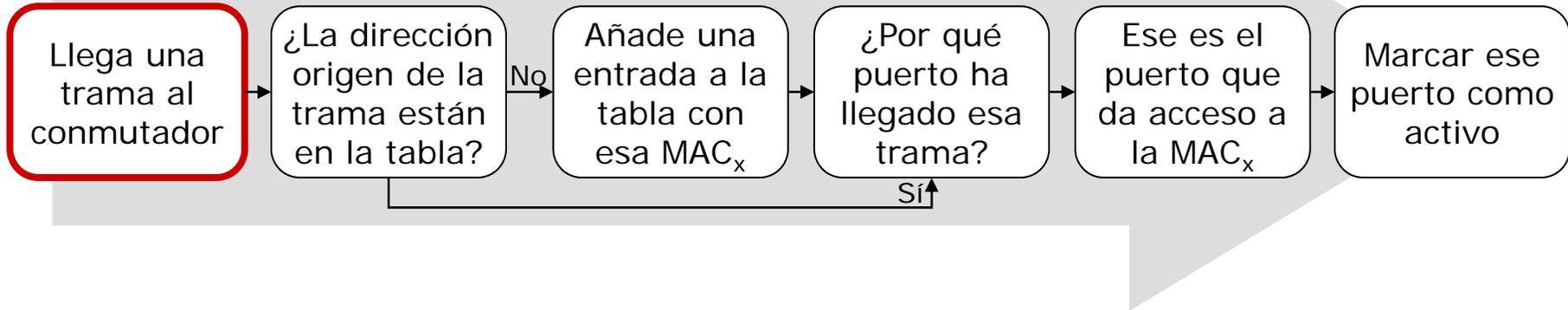
## 7.2.1 Tabla de direcciones

### aprendizaje



# 7.2.1 Tabla de direcciones

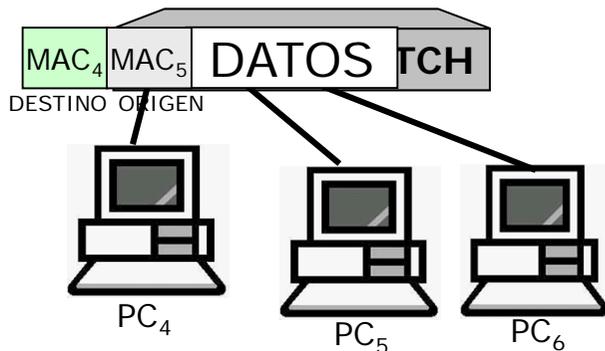
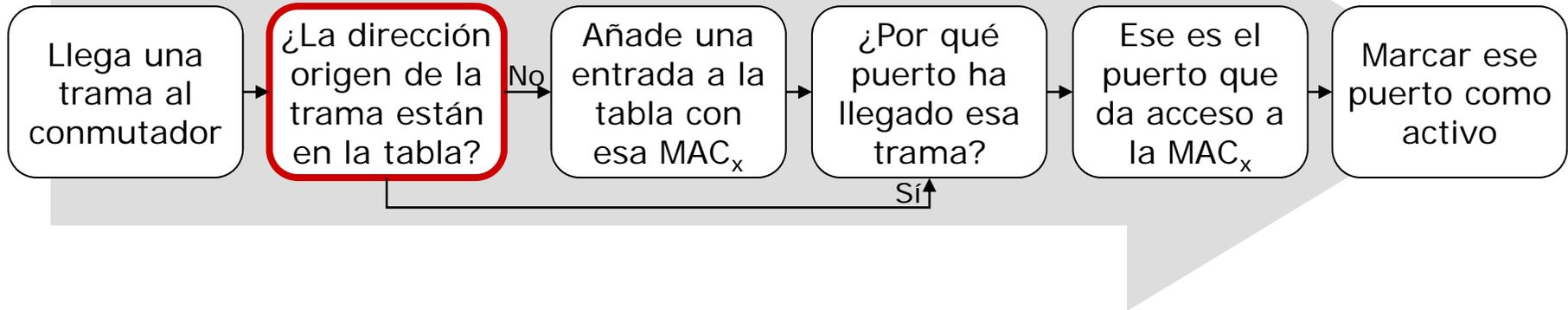
## aprendizaje



MAC	Puerto <sub>salida</sub>	Actividad
MAC <sub>6</sub>	4	0
MAC <sub>4</sub>	1	1

# 7.2.1 Tabla de direcciones

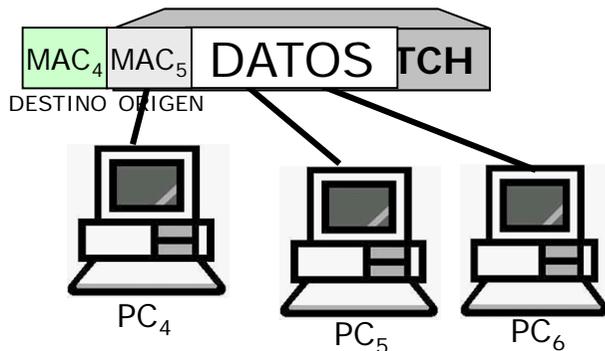
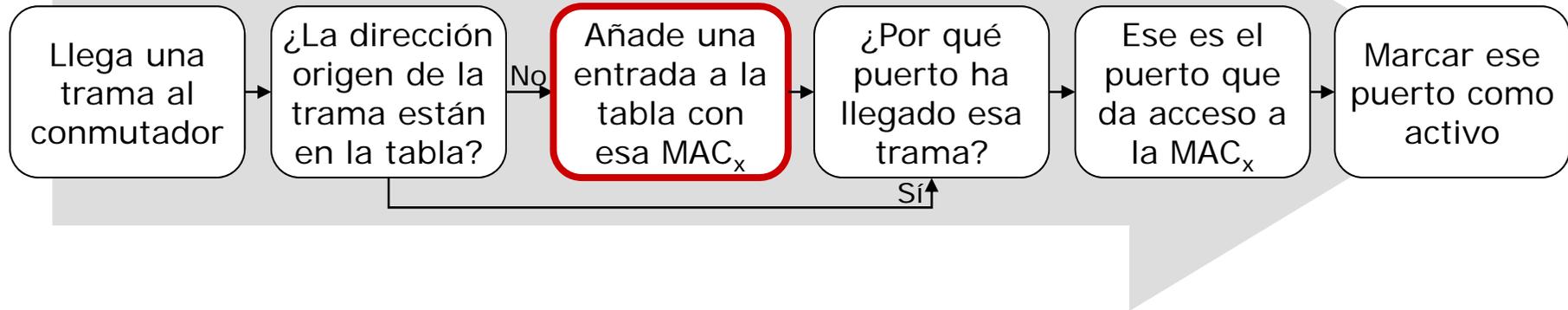
## aprendizaje



MAC	Puerto <sub>salida</sub>	Actividad
MAC <sub>6</sub>	4	0
MAC <sub>4</sub>	1	1

# 7.2.1 Tabla de direcciones

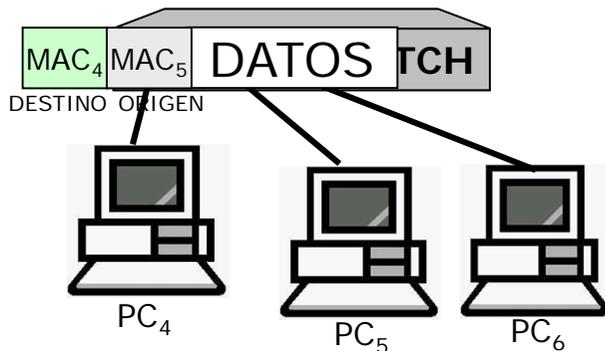
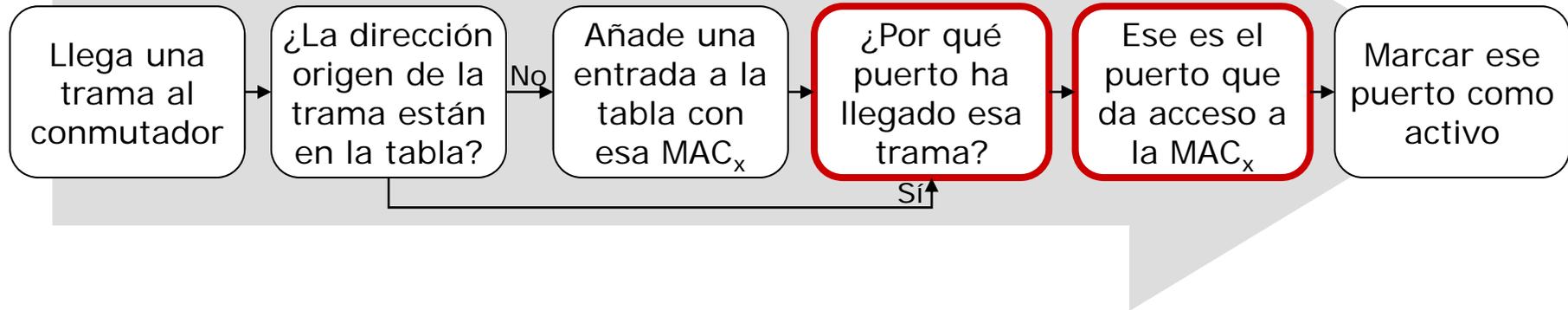
## aprendizaje



MAC	Puerto <sub>salida</sub>	Actividad
MAC <sub>6</sub>	4	0
MAC <sub>4</sub>	1	1
MAC <sub>5</sub>		

# 7.2.1 Tabla de direcciones

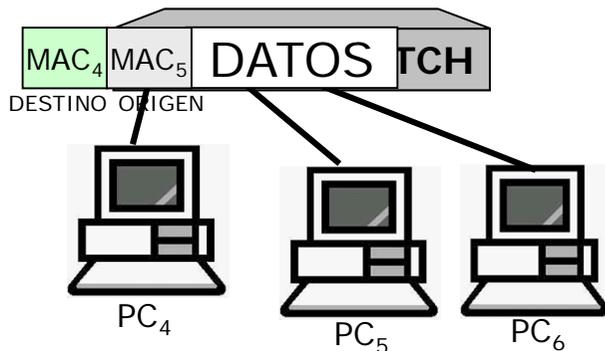
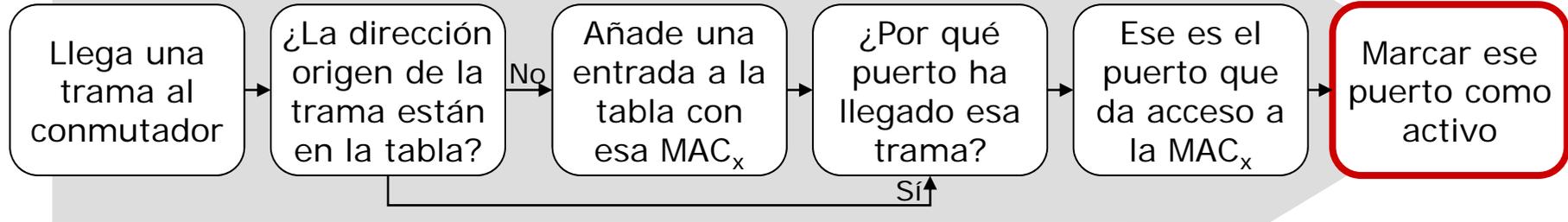
## aprendizaje



MAC	Puerto <sub>salida</sub>	Actividad
MAC <sub>6</sub>	4	0
MAC <sub>4</sub>	1	1
MAC <sub>5</sub>	3	

# 7.2.1 Tabla de direcciones

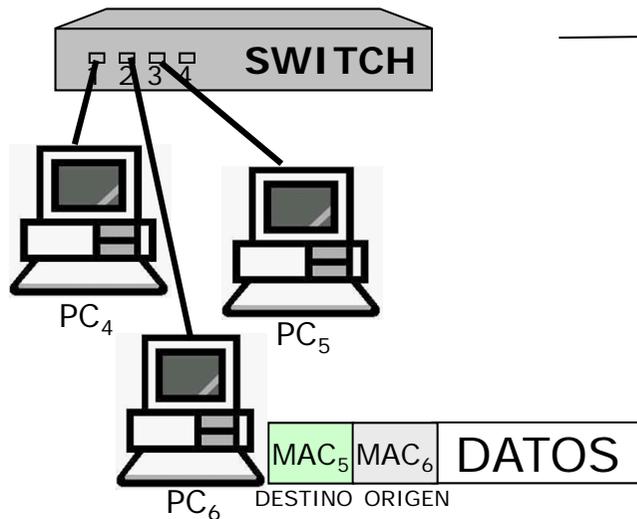
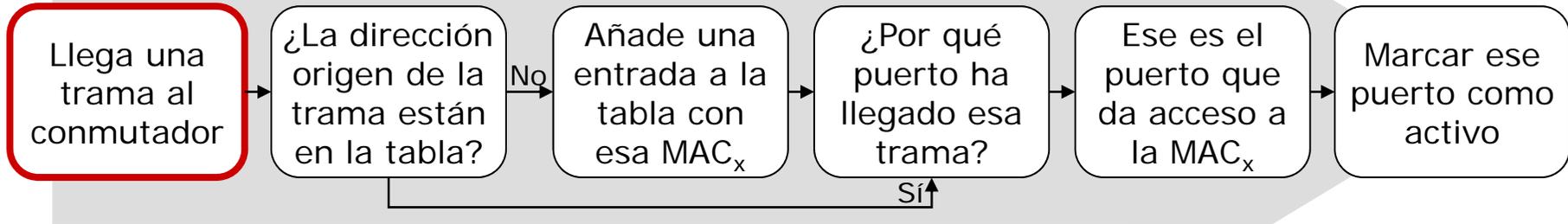
## aprendizaje



MAC	Puerto <sub>salida</sub>	Actividad
MAC <sub>6</sub>	4	0
MAC <sub>4</sub>	1	1
MAC <sub>5</sub>	3	1

# 7.2.1 Tabla de direcciones

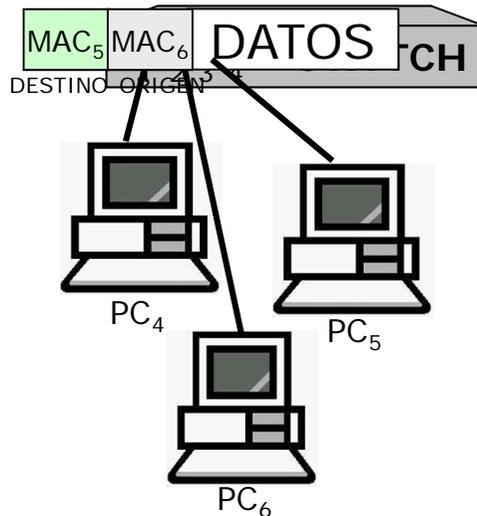
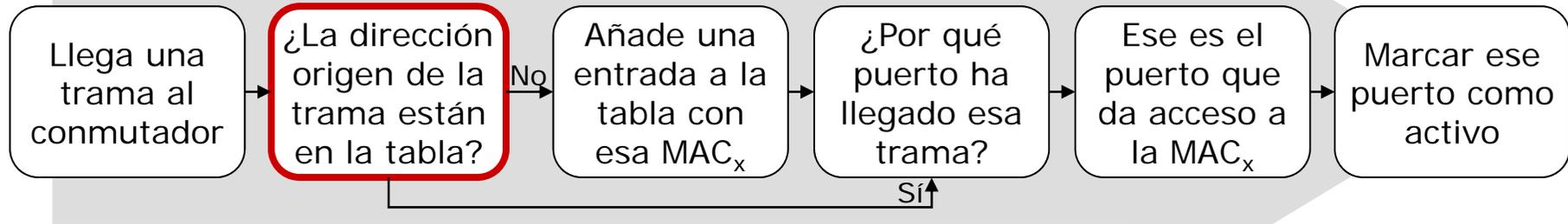
## aprendizaje



MAC	Puerto salida	Actividad
MAC <sub>6</sub>	4	0
MAC <sub>4</sub>	1	1
MAC <sub>5</sub>	3	1

# 7.2.1 Tabla de direcciones

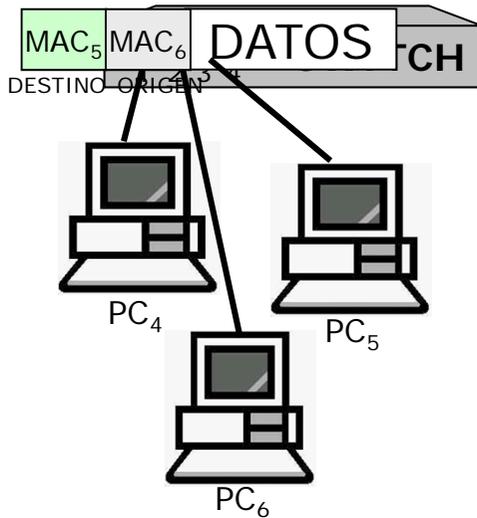
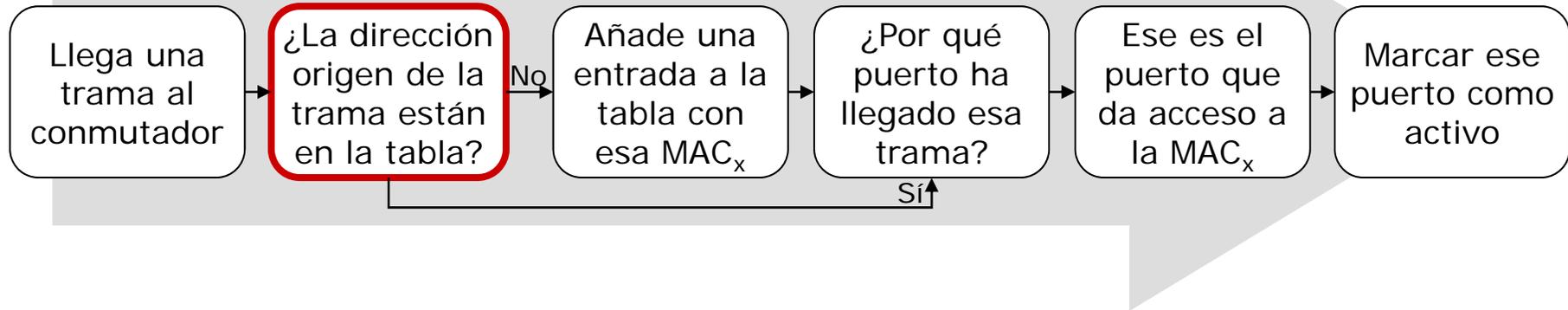
## aprendizaje



MAC	Puerto salida	Actividad
MAC <sub>6</sub>	4	0
MAC <sub>4</sub>	1	1
MAC <sub>5</sub>	2	1

# 7.2.1 Tabla de direcciones

## aprendizaje

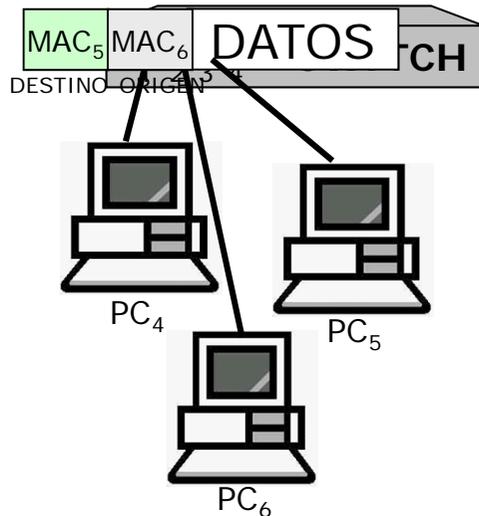


### MAC - Puerto<sub>salida</sub> - Actividad

MAC <sub>6</sub>	4	0
MAC <sub>4</sub>	1	1
MAC <sub>5</sub>	2	1

# 7.2.1 Tabla de direcciones

## aprendizaje

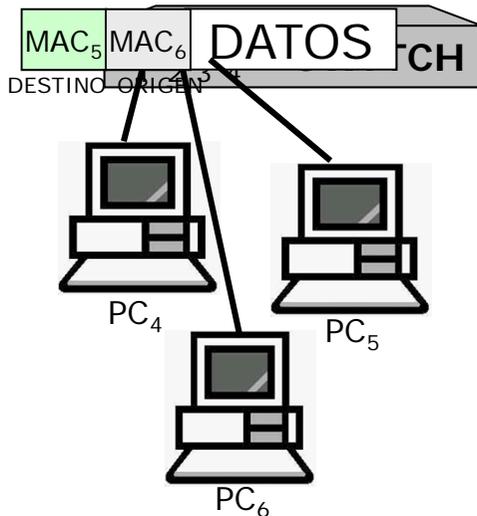
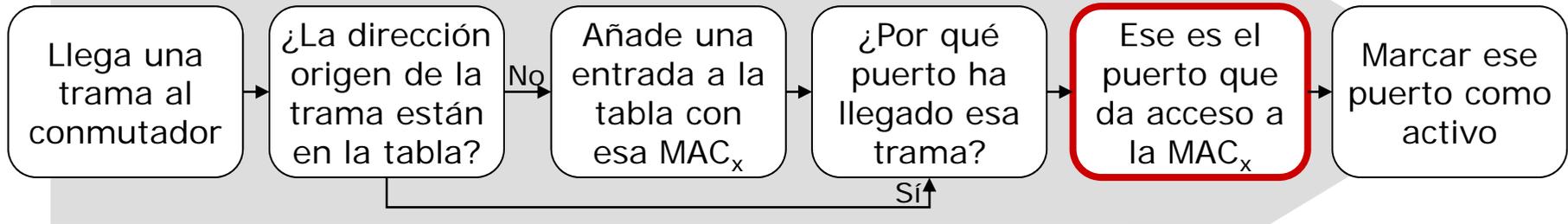


MAC - Puerto<sub>salida</sub> - Actividad

MAC <sub>6</sub>	4	0
MAC <sub>4</sub>	1	1
MAC <sub>5</sub>	2	1

# 7.2.1 Tabla de direcciones

## aprendizaje

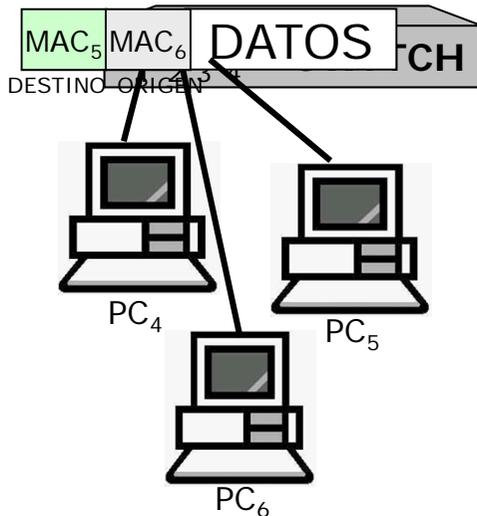
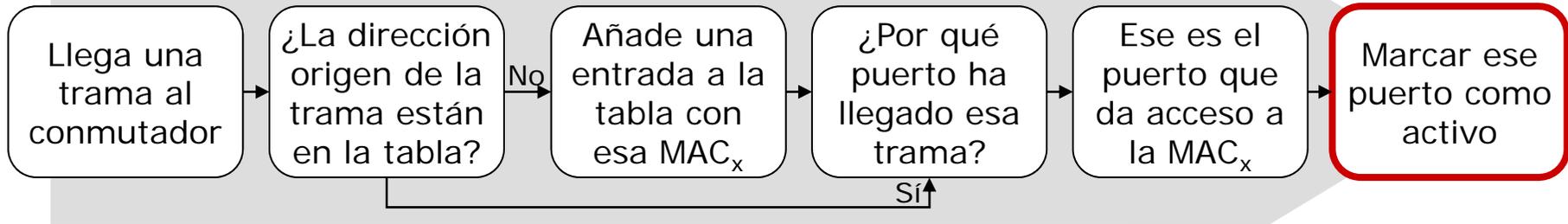


MAC - Puerto<sub>salida</sub> - Actividad

MAC <sub>6</sub>	2	0
MAC <sub>4</sub>	1	1
MAC <sub>5</sub>	2	1

# 7.2.1 Tabla de direcciones

## aprendizaje



MAC - Puerto<sub>salida</sub> - Actividad

MAC <sub>6</sub>	2	1
MAC <sub>4</sub>	1	1
MAC <sub>5</sub>	2	1

## 7.2.1 Tabla de direcciones

---

- Es importante tener en cuenta que:
  - **Las entradas no activas de la tabla de direcciones deben borrarse periódicamente**
    - ① Cuanto más larga sea la tabla → más costoso es el proceso de consulta

Las prestaciones del conmutador dependen, en gran medida, de lo rápido que se hagan las consultas a la tabla de direcciones

Tiene sentido limitar la longitud de la tabla → sólo aquellas estaciones que realmente están activas
    - ② Cuando una estación cambia de puerto, la tabla de direcciones no se actualiza automáticamente

La información que contiene será incorrecta hasta que la estación que ha cambiado de ubicación envíe una trama

Si la estación que ha cambiado de ubicación nunca envía una trama la entrada se borrará

Al borrar la entrada, cuando llegue una trama con ese destino, se activará el mecanismo de *flooding* con lo que finalmente se podrá llegar hasta la estación desplazada

# Contenidos

---

5. Protocolos de Control de Acceso al Medio

6. Direccionamiento

7. Hubs, Puentes y Conmutadores

1. Hubs

2. Puentes y conmutadores

1. Tabla de direcciones

2. Filtrado, reenvío e inundación

3. Procesos a realizar por un conmutador y sus prioridades

4. Cut-Through vs. Store-and-Forward

8. Protocolo STP

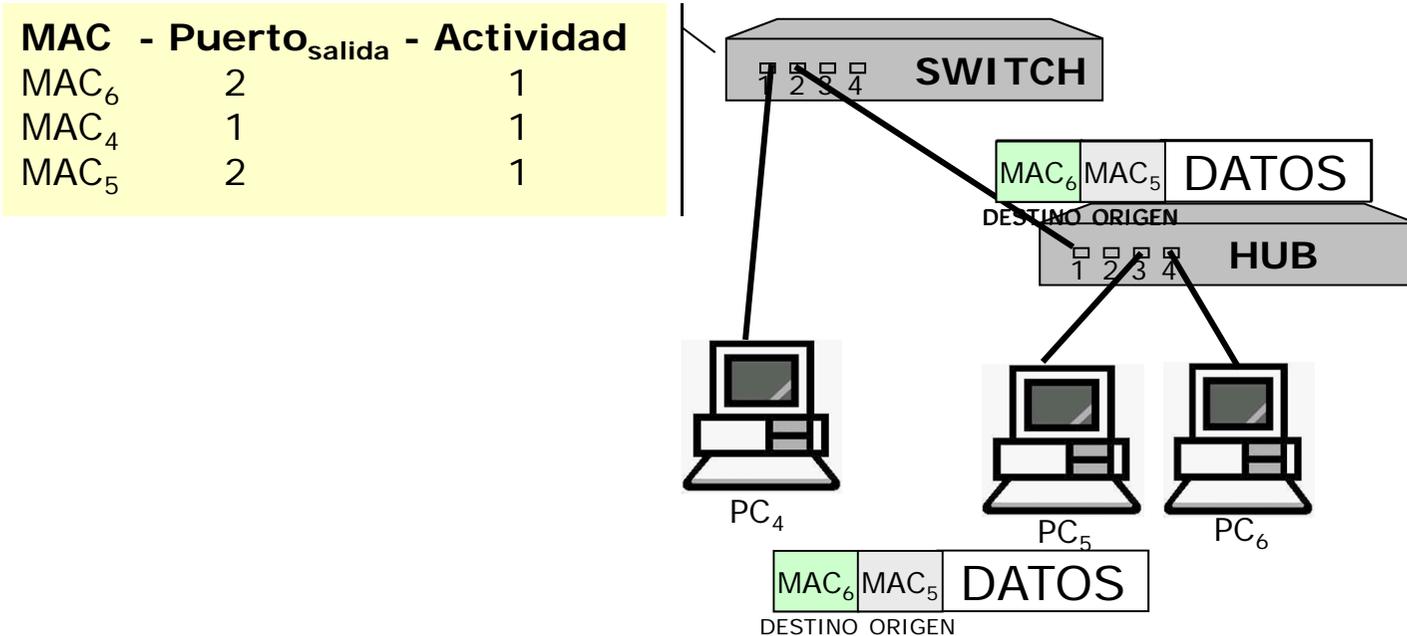
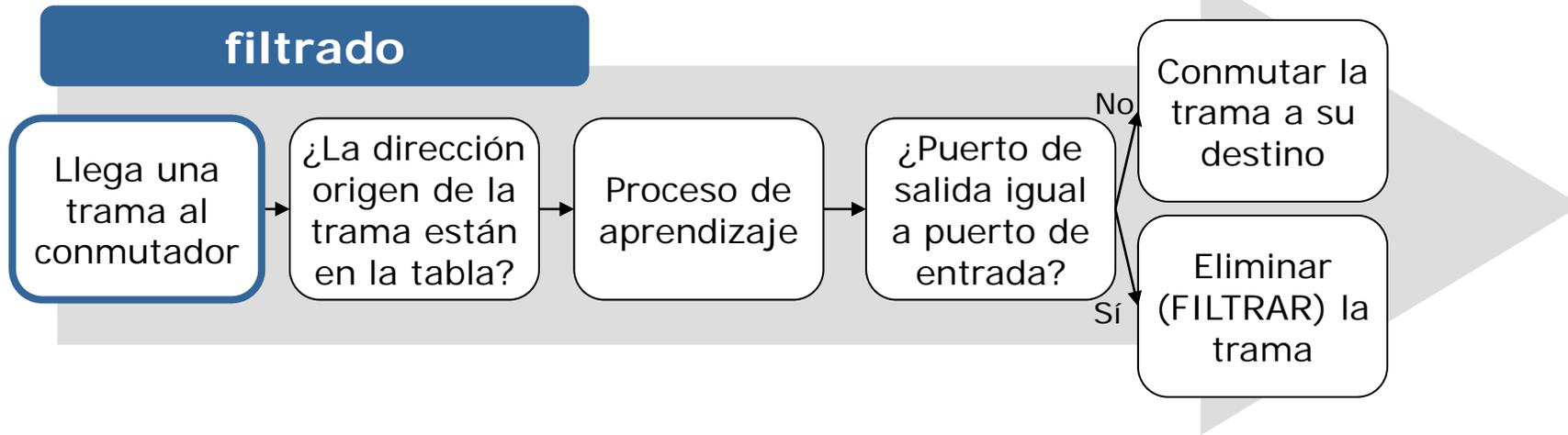
9. Ejemplo de protocolos de nivel de enlace de datos: HDLC

## 7.2.2 Filtrado, reenvío e inundación

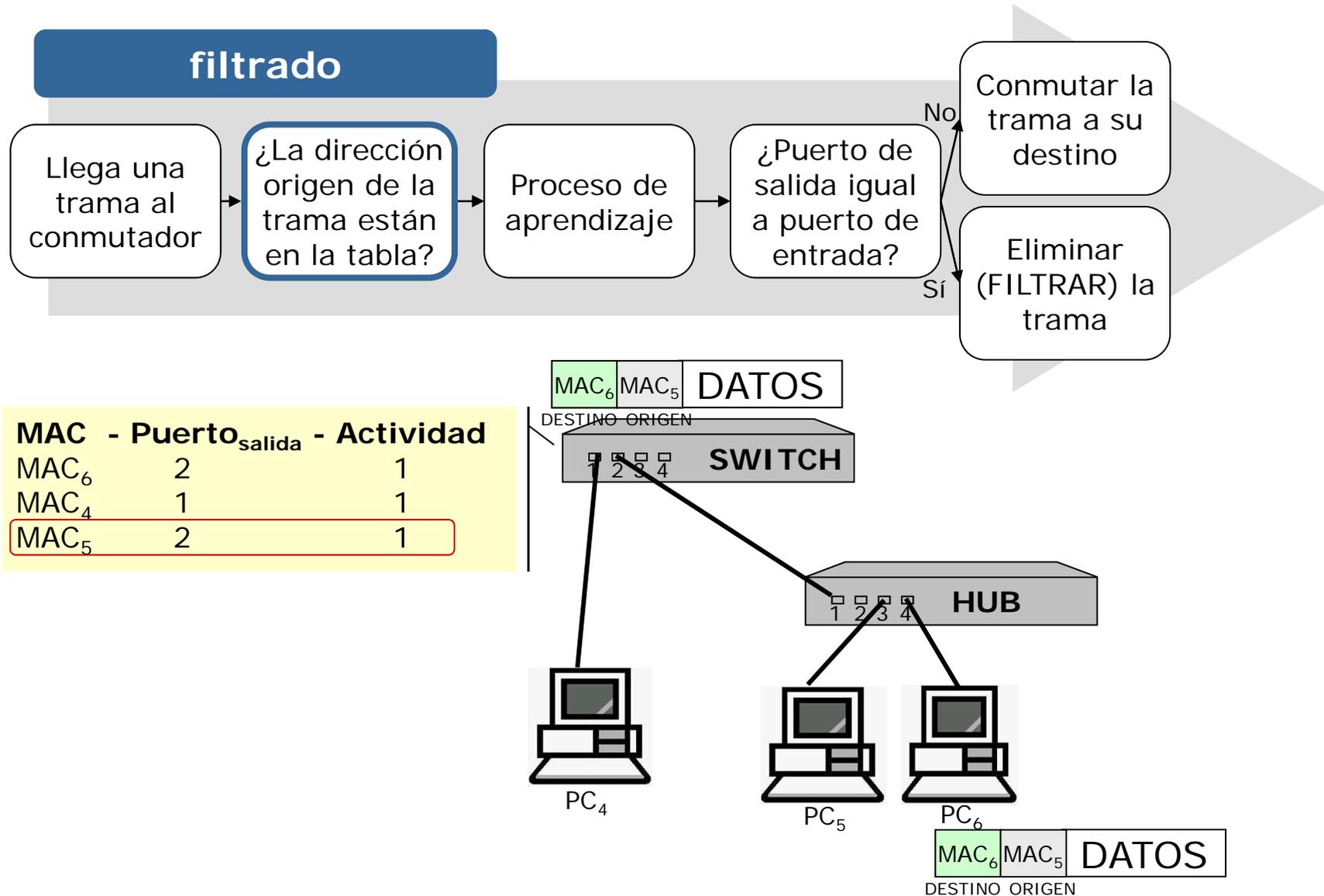
---

- El **filtrado** consiste en no retransmitir una trama si la estación destino está en el mismo dominio de colisión que la estación origen
  - En ese caso, el conmutador no es necesario para que haya comunicación entre las dos estaciones
  - Es imprescindible ya que de lo contrario podrían generarse duplicados y un incremento innecesario del tráfico en la red

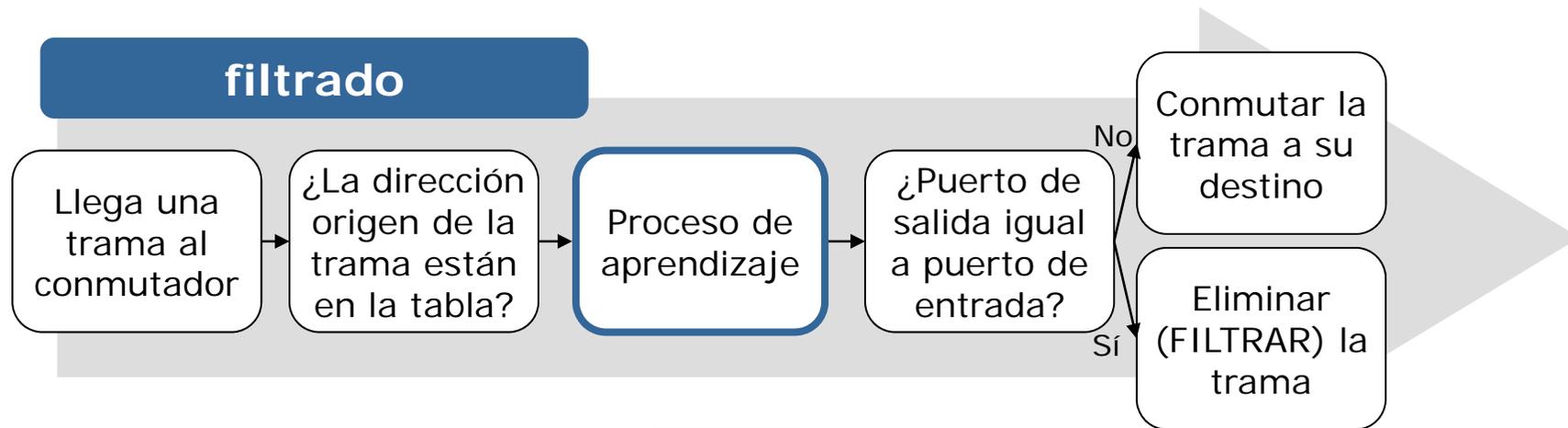
# 7.2.2 Filtrado, reenvío e inundación



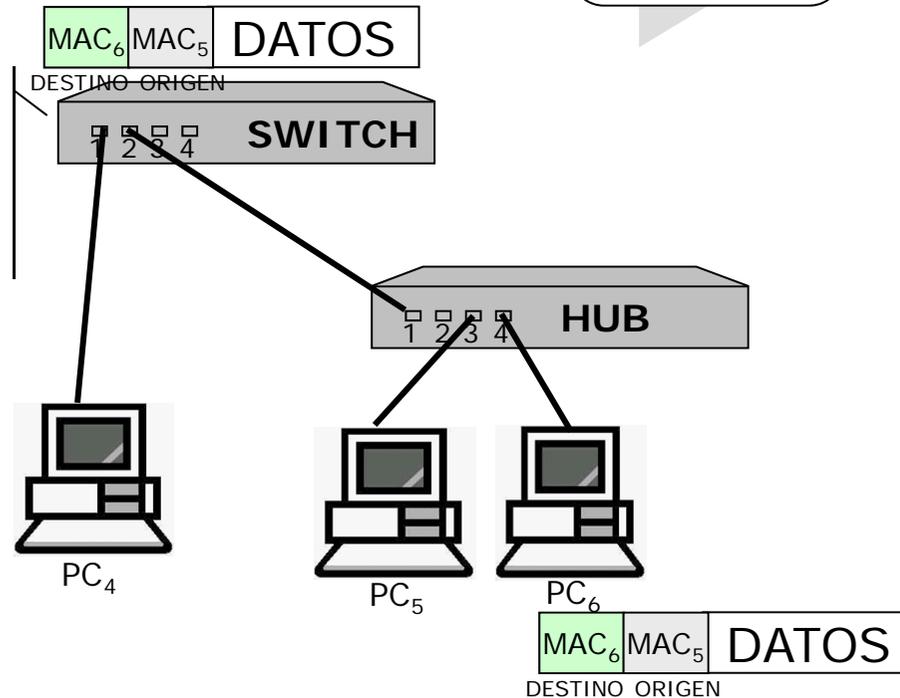
# 7.2.2 Filtrado, reenvío e inundación



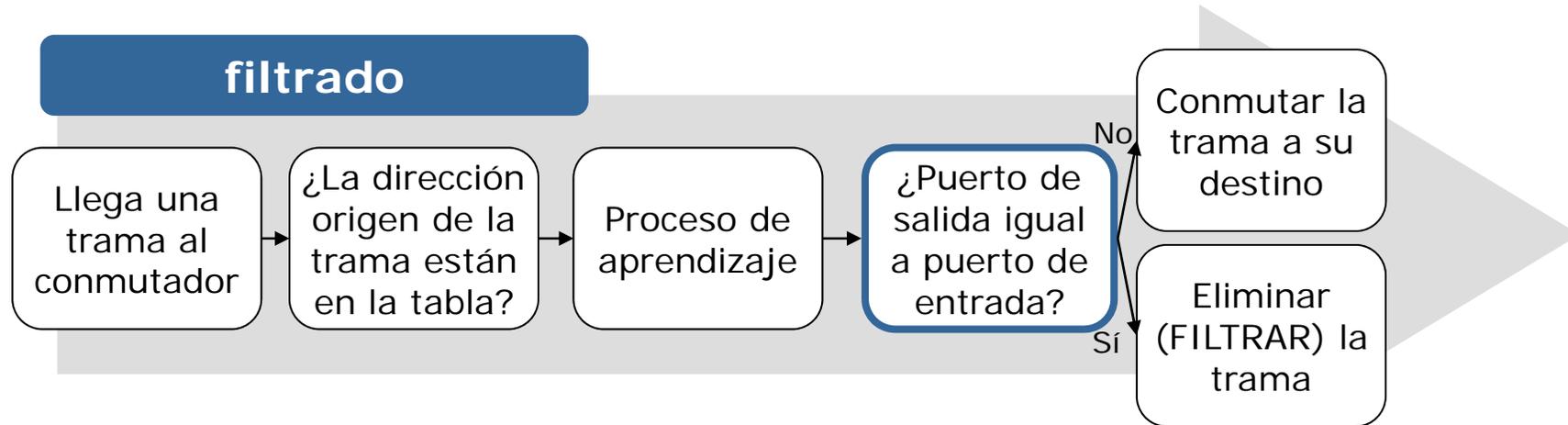
# 7.2.2 Filtrado, reenvío e inundación



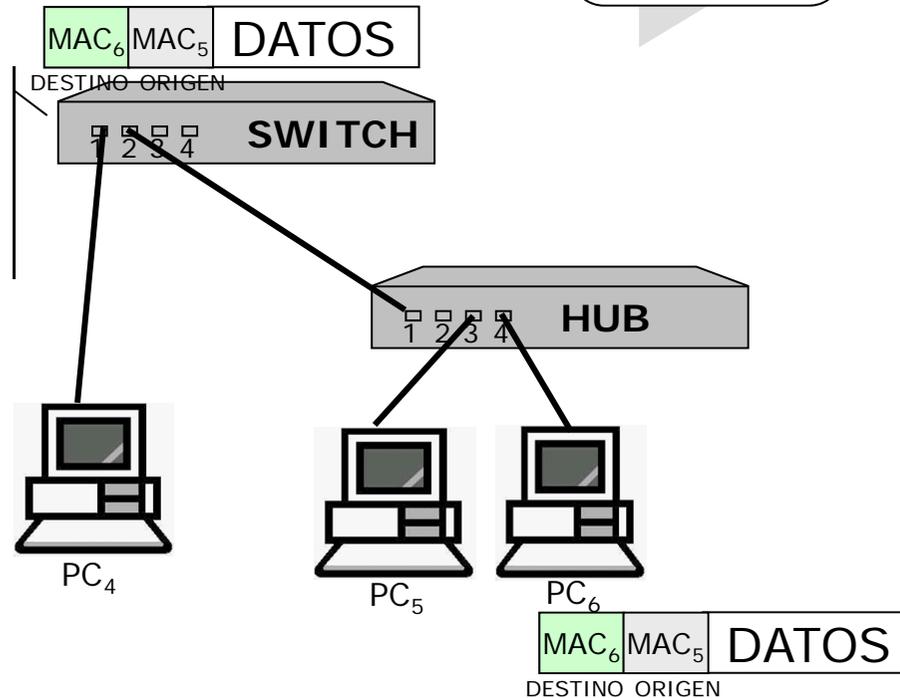
MAC	Puerto <sub>salida</sub>	Actividad
MAC <sub>6</sub>	2	1
MAC <sub>4</sub>	1	1
MAC <sub>5</sub>	2	1



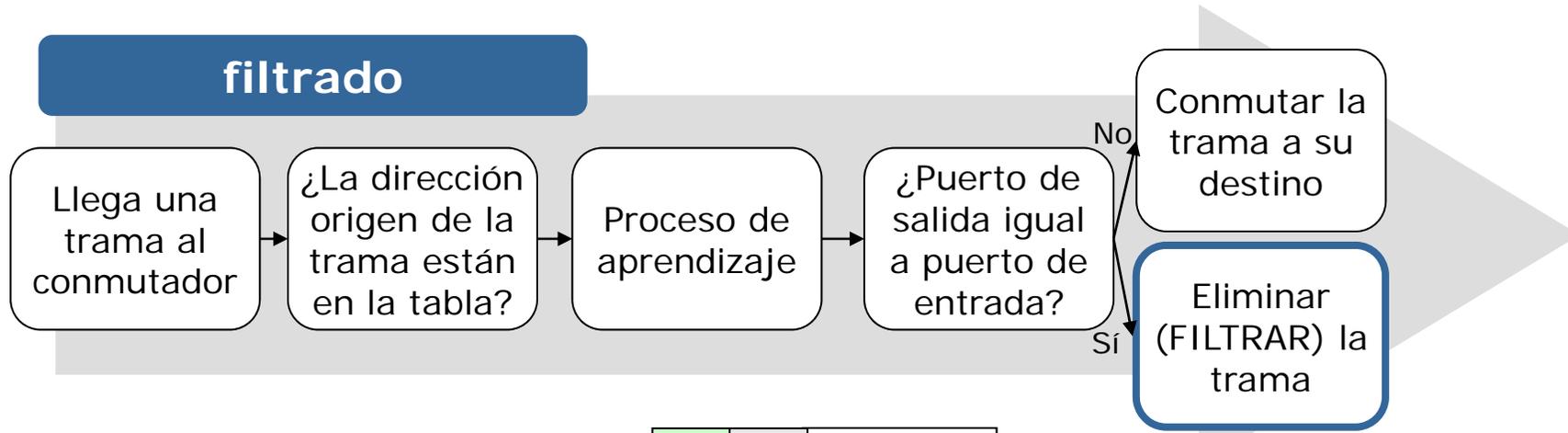
# 7.2.2 Filtrado, reenvío e inundación



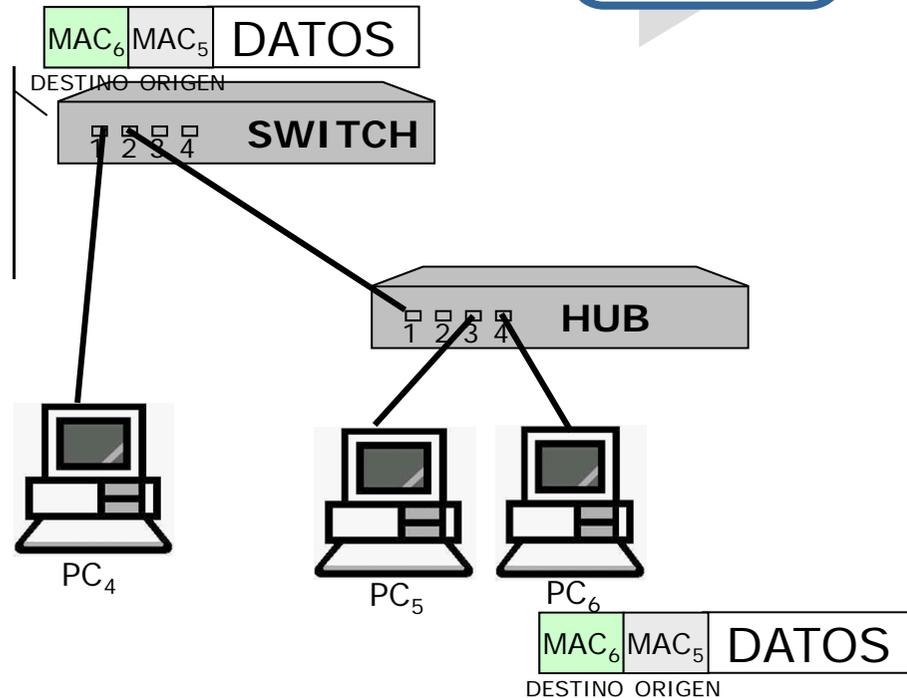
MAC	Puerto <sub>salida</sub>	Actividad
MAC <sub>6</sub>	2	1
MAC <sub>4</sub>	1	1
MAC <sub>5</sub>	2	1



# 7.2.2 Filtrado, reenvío e inundación



MAC	Puerto <sub>salida</sub>	Actividad
MAC <sub>6</sub>	2	1
MAC <sub>4</sub>	1	1
MAC <sub>5</sub>	2	1



## 7.2.2 Filtrado, reenvío e inundación

---

- El **reenvío** consiste conmutar una trama hacia su destino pero teniendo en cuenta el protocolo de acceso al medio que esté en uso
  - El conmutador es un dispositivo más que accederá al medio mediante un protocolo de reserva, de acceso aleatorio (contienda) o por turnos
  - Debe respetar las reglas dictadas por el mecanismo de control de acceso al medio de la subred a la que va dirigida el paquete

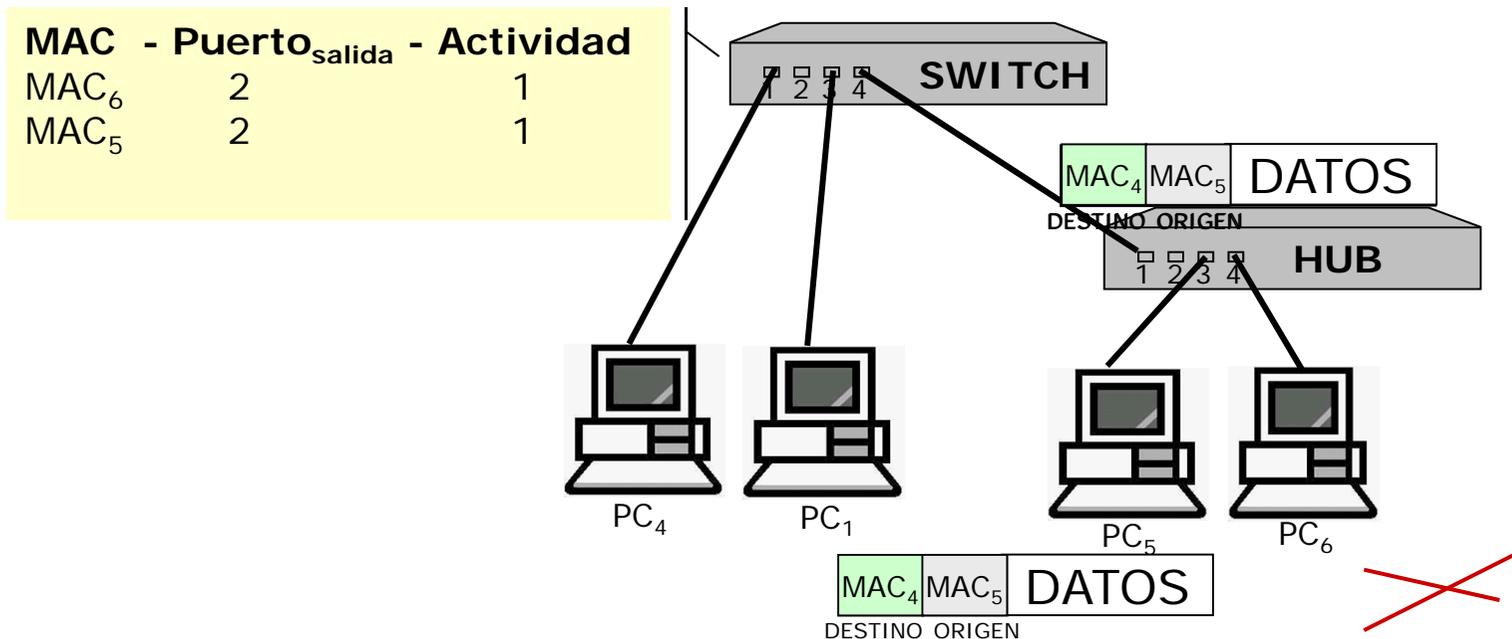
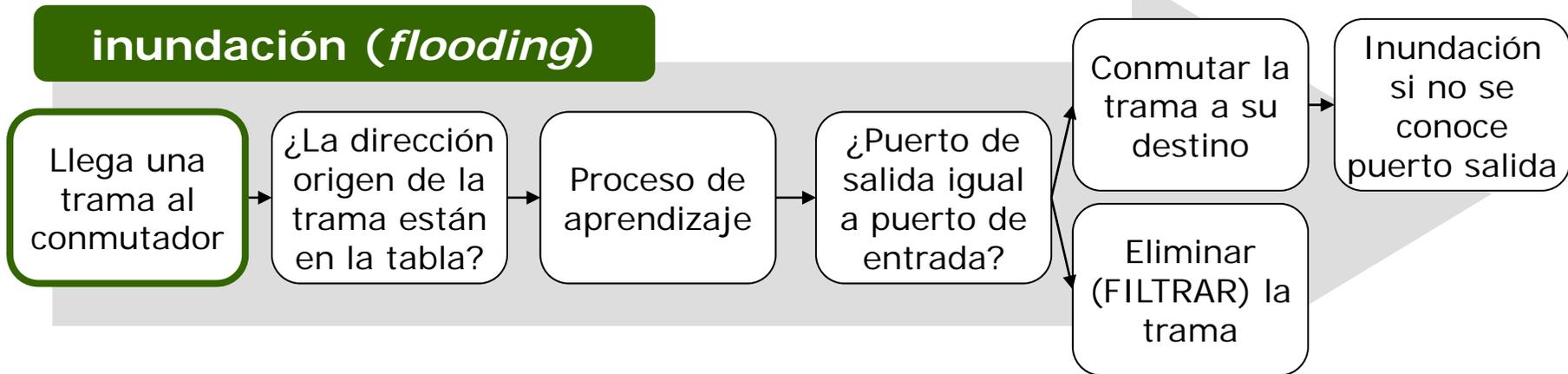
## 7.2.2 Filtrado, reenvío e inundación

---

- La **inundación** (*flooding*) es el proceso que sigue un conmutador cuando no sabe cuál es el puerto para acceder a un equipo
  - Consiste en reenviar la trama por todos los puertos excepto por el puerto por el que la trama llegó al conmutador

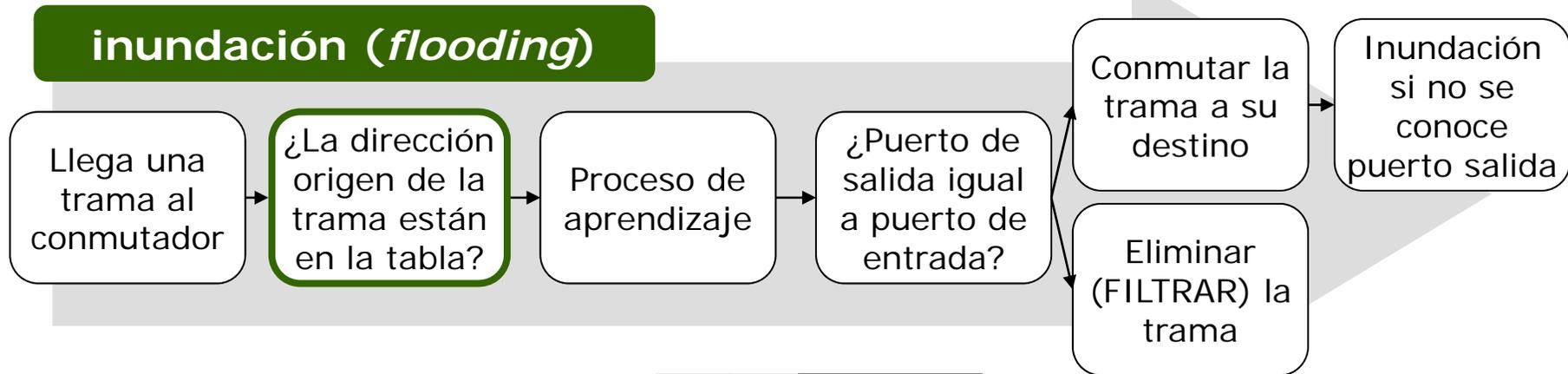
# 7.2.2 Filtrado, reenvío e inundación

## inundación (*flooding*)

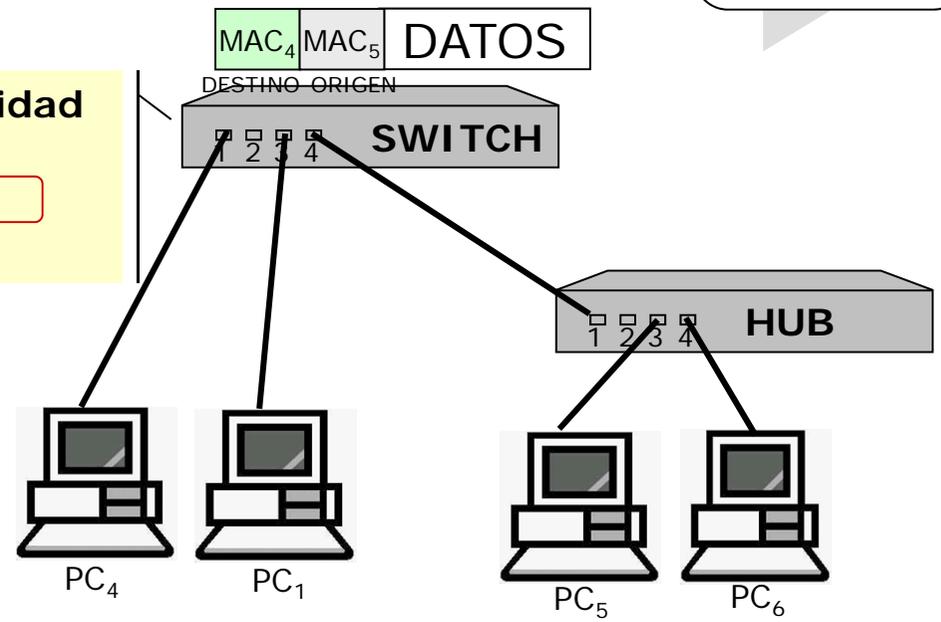


# 7.2.2 Filtrado, reenvío e inundación

## inundación (*flooding*)

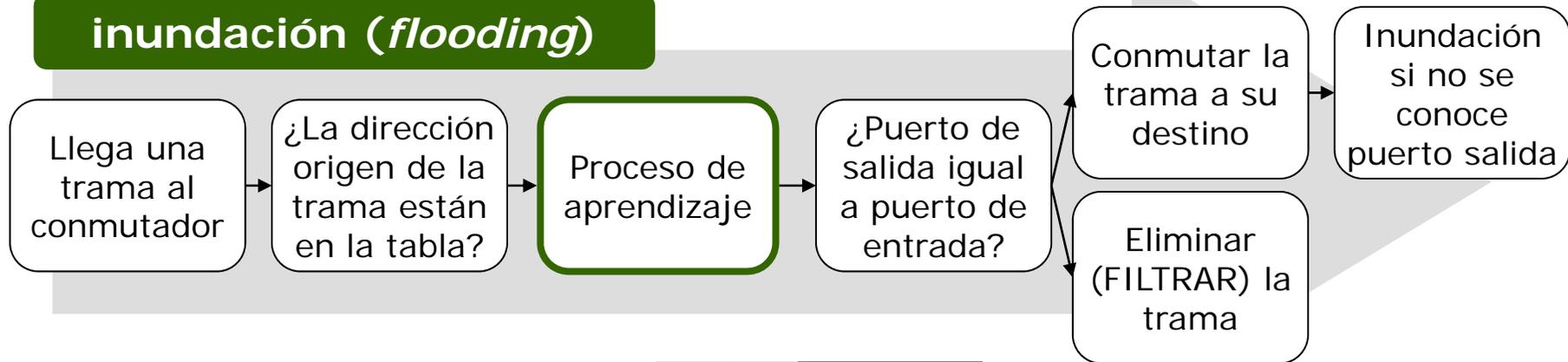


MAC	Puerto <sub>salida</sub>	Actividad
MAC <sub>6</sub>	2	1
MAC <sub>5</sub>	2	1

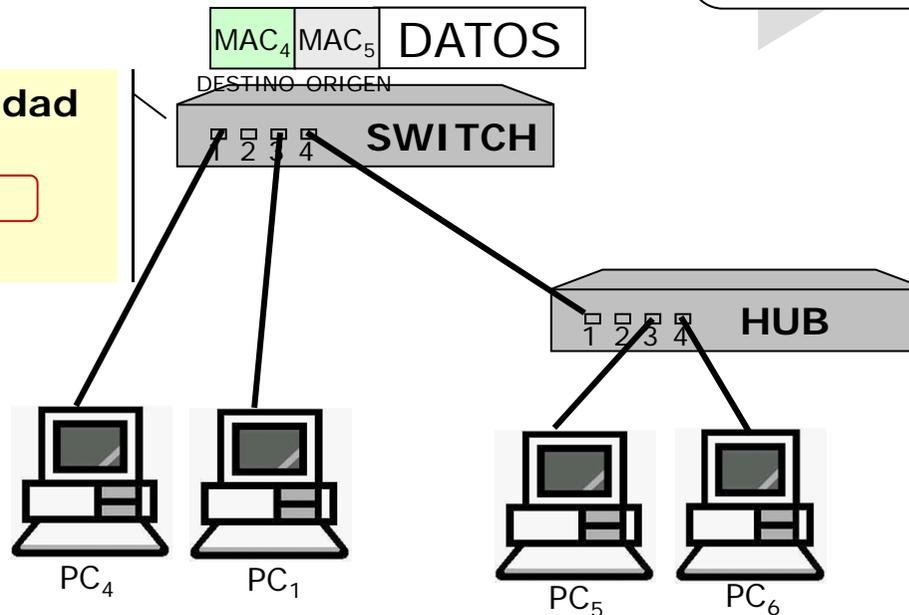


## 7.2.2 Filtrado, reenvío e inundación

### inundación (*flooding*)

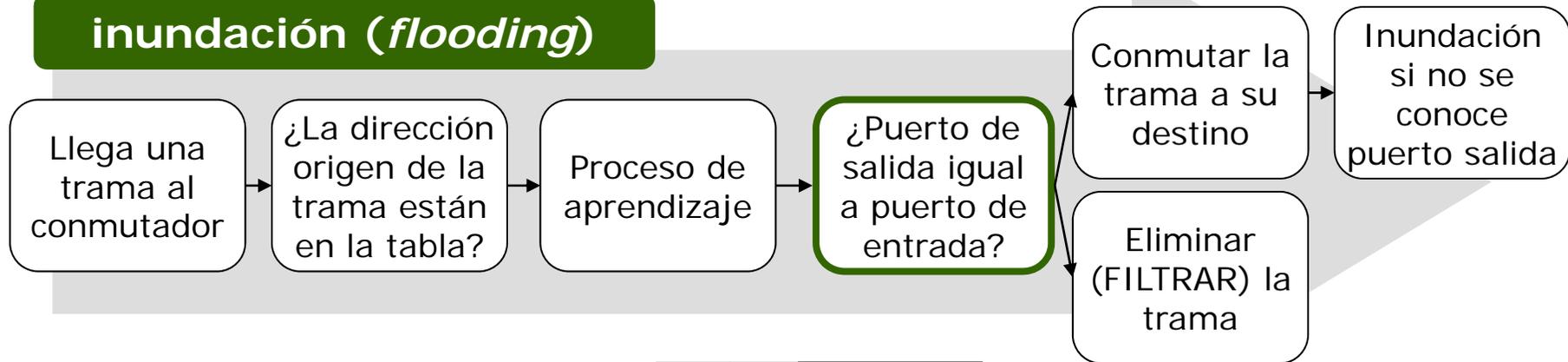


MAC	Puerto <sub>salida</sub>	Actividad
MAC <sub>6</sub>	2	1
MAC <sub>5</sub>	2	1

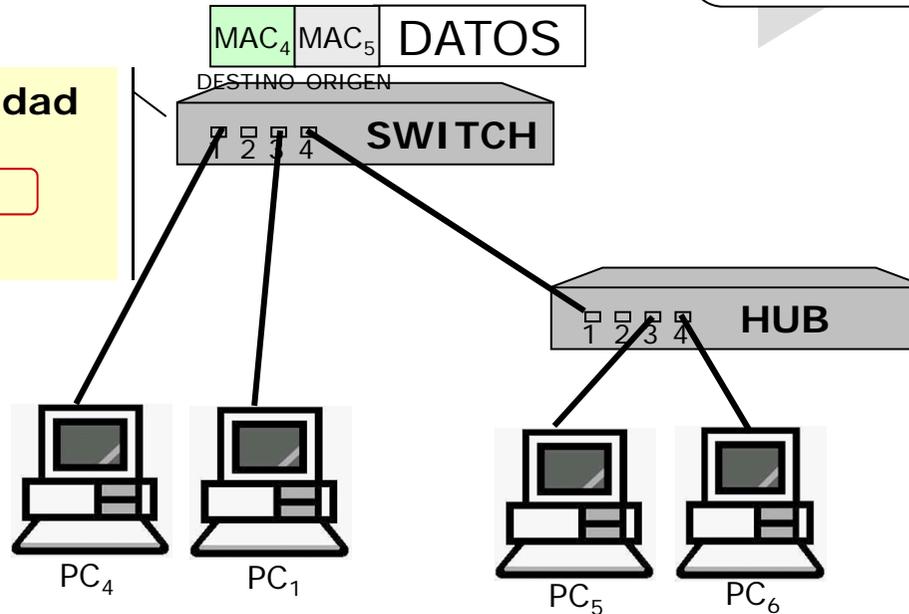


## 7.2.2 Filtrado, reenvío e inundación

### inundación (*flooding*)

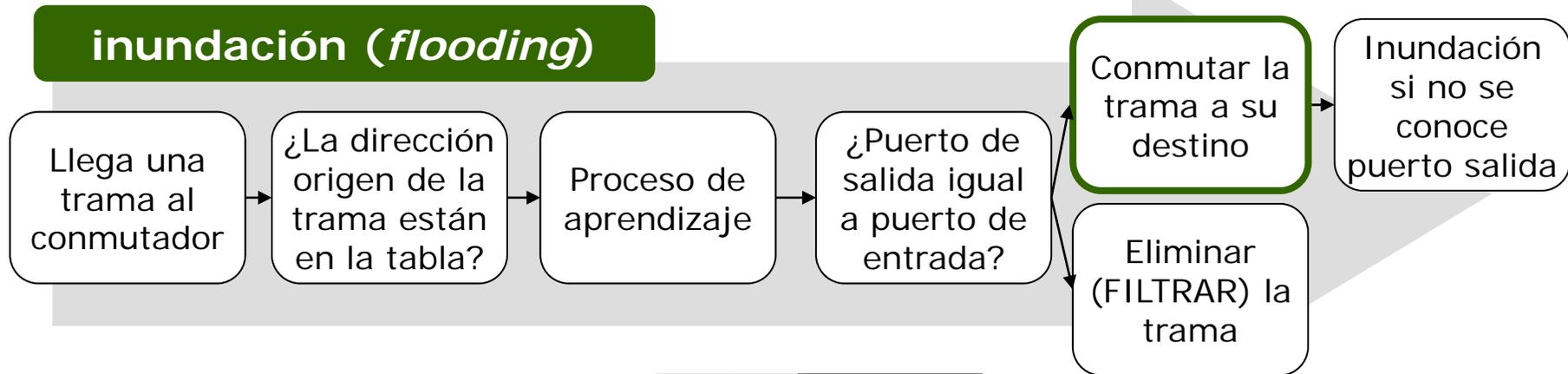


MAC	Puerto <sub>salida</sub>	Actividad
MAC <sub>6</sub>	2	1
MAC <sub>5</sub>	2	1

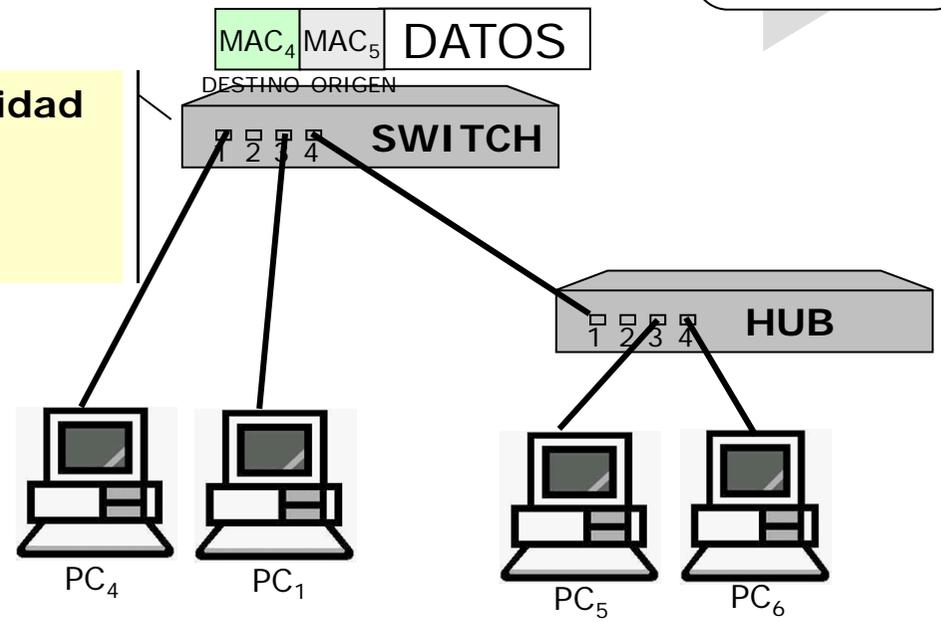


# 7.2.2 Filtrado, reenvío e inundación

## inundación (*flooding*)

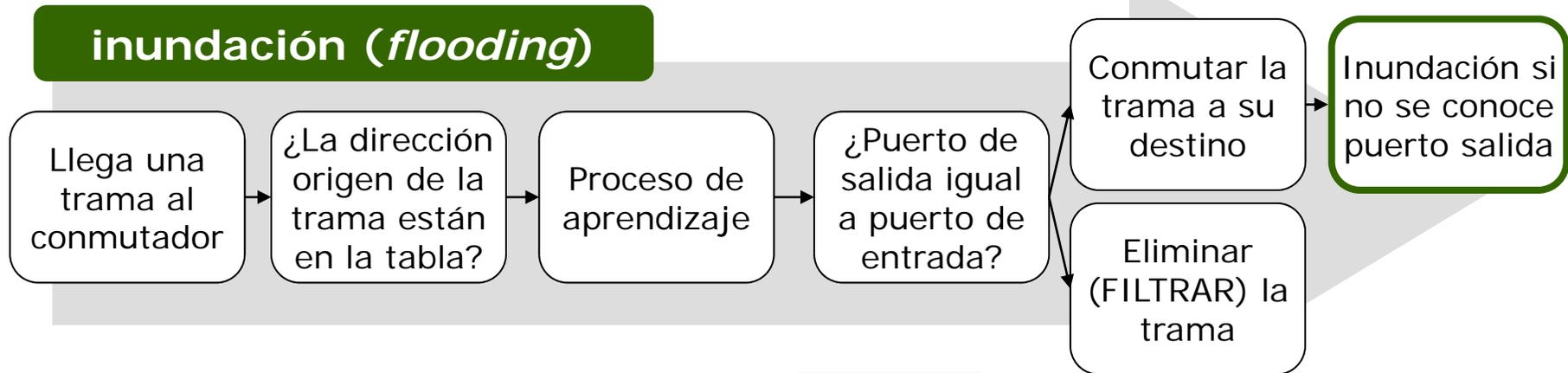


MAC	Puerto <sub>salida</sub>	Actividad
MAC <sub>6</sub>	2	1
MAC <sub>5</sub>	2	1
¿destino?		

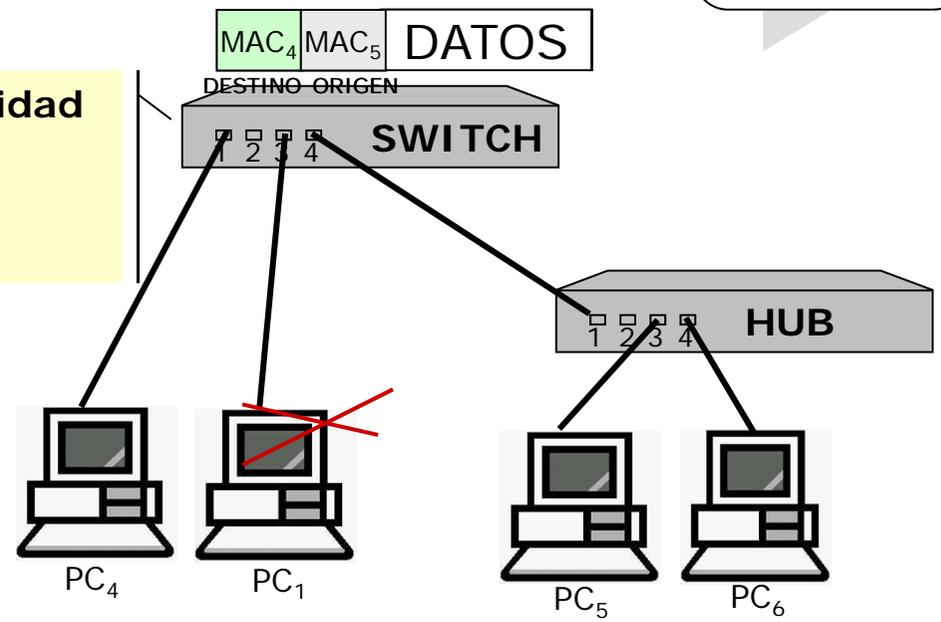


# 7.2.2 Filtrado, reenvío e inundación

## inundación (*flooding*)



MAC	Puerto salida	Actividad
MAC <sub>6</sub>	2	1
MAC <sub>5</sub>	2	1



## 7.2.2 Filtrado, reenvío e inundación

---

- Observa que el conmutador no cambia las direcciones origen y destino de las tramas
- Las estaciones no van a saber si la trama que reciben ha pasado antes por un conmutador
- El conmutador es **transparente**, tanto para las estaciones como para los otros conmutadores

# Contenidos

---

5. Protocolos de Control de Acceso al Medio

6. Direccionamiento

7. Hubs, Puentes y Conmutadores

1. Hubs

2. Puentes y conmutadores

1. Tabla de direcciones

2. Filtrado, reenvío e inundación

3. Procesos a realizar por un conmutador y sus prioridades

4. Cut-Through vs. Store-and-Forward

8. Protocolo STP

9. Ejemplo de protocolos de nivel de enlace de datos: HDLC

## 7.2.3 Procesos a realizar por un conmutador y sus prioridades

---

**CONSULTA DE DIRECCIONES**

**APRENDIZAJE**

**ENVEJECIMIENTO**

## 7.2.3 Procesos a realizar por un conmutador y sus prioridades

---

**CONSULTA DE DIRECCIONES**



- Se encarga de la comparación de la dirección destino de las tramas entrantes con las entradas de la tabla de direcciones
- Determina si hay que filtrar la trama, reenviarla o inundar todos los puertos

**APRENDIZAJE**

**ENVEJECIMIENTO**

## 7.2.3 Procesos a realizar por un conmutador y sus prioridades

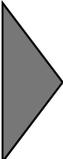
---

### CONSULTA DE DIRECCIONES



- Se encarga de la comparación de la dirección destino de las tramas entrantes con las entradas de la tabla de direcciones
- Determina si hay que filtrar la trama, reenviarla o inundar todos los puertos

### APRENDIZAJE



- Se encarga de comparar la dirección origen de las tramas entrantes con las entradas de la tabla de direcciones
- Actualiza la correspondencia entre direcciones y puertos, actualiza marcadores de actividad o crea nuevas entradas

### ENVEJECIMIENTO

## 7.2.3 Procesos a realizar por un conmutador y sus prioridades

---

### CONSULTA DE DIRECCIONES

- Se encarga de la comparación de la dirección destino de las tramas entrantes con las entradas de la tabla de direcciones
- Determina si hay que filtrar la trama, reenviarla o inundar todos los puertos

### APRENDIZAJE

- Se encarga de comparar la dirección origen de las tramas entrantes con las entradas de la tabla de direcciones
- Actualiza la correspondencia entre direcciones y puertos, actualiza marcadores de actividad o crea nuevas entradas

### ENVEJECIMIENTO

- Se encarga de eliminar de la lista las entradas de la tabla que se hayan quedado obsoletas

## 7.2.3 Procesos a realizar por un conmutador y sus prioridades

---

### CONSULTA DE DIRECCIONES

- Se encarga de la comparación de la dirección destino de las tramas entrantes con las entradas de la tabla de direcciones
- Determina si hay que filtrar la trama, reenviarla o inundar todos los puertos

- Debe realizarse en tiempo real (trama a trama)
- Retardo supone ↓ prestaciones

### APRENDIZAJE

- Se encarga de comparar la dirección origen de las tramas entrantes con las entradas de la tabla de direcciones
- Actualiza la correspondencia entre direcciones y puertos, actualiza marcadores de actividad o crea nuevas entradas

- No es crítico (no necesario con cada trama)
- Siempre se puede usar flooding

### ENVEJECIMIENTO

- Se encarga de eliminar de la lista las entradas de la tabla que se hayan quedado obsoletas

- Puede estar en segundo plano
- Eliminar entradas antiguas sin ser exhaustivo

# Contenidos

---

5. Protocolos de Control de Acceso al Medio

6. Direccionamiento

7. Hubs, Puentes y Conmutadores

1. Hubs

2. Puentes y conmutadores

1. Tabla de direcciones

2. Filtrado, reenvío e inundación

3. Procesos a realizar por un conmutador y sus prioridades

4. Cut-Through vs. Store-and-Forward

8. Protocolo STP

9. Ejemplo de protocolos de nivel de enlace de datos: HDLC

## 7.2.4 Cut-Through vs. Store-and-Forward

---

- El modo de operación del conmutador es almacenar y reenviar (***Store and Forward***):
  - Las tramas recibidas, primero se almacenan
  - Se comprueba el CRC y se descartan aquellas tramas que tengan un CRC erróneo
  - Se lee su dirección destino y se lanza el proceso de consulta de tabla de direcciones
  - La trama se reenvía hacia uno de los puertos donde espera a ser transmitida (hasta que dicho puerto esté disponible)
  - Siempre cumpliendo con las normas marcadas por el mecanismo de control de acceso al medio

**...como las direcciones MAC origen y destino son los primeros campos de la trama, ¿podríamos consultar la tabla antes de tener la trama completa?**

## 7.2.4 Cut-Through vs. Store-and-Forward

---

### □ El modo *Cut-Through*

- Una vez conocida la MAC origen se inicia todo el proceso
- Una vez conocida la MAC destino comienza a conmutar la trama por el puerto de salida correspondiente (puede que la trama aún esté llegando por el puerto de entrada)
- En principio *Cut-Through* reduce la latencia introducida por el conmutador:
  - Sólo es cierto si el reenvío puede empezar inmediatamente después de conocer el puerto de salida
  - Si el puerto está ocupado o hay que esperar a que llegue el testigo o hay otra trama en el canal de salida ⇒ la latencia no cambia (necesario almacenar la trama para reenviarla después)
  - Probablemente sólo se puede usar *Cut-Through* en condiciones de poco tráfico (cuando menos se necesita)
  - No se puede utilizar con direcciones *multicast* o direcciones desconocidas puesto que todos los puertos tendrían que estar disponibles
  - Propaga las tramas erróneas ⇒ no se comprueba el CRC (últimos *bytes* de la trama) antes de reenviar una trama

# Próximo día

---

1. Introducción
  1. Funciones de un protocolo de enlace de datos
2. Entramado
  1. Protocolos orientados a carácter
  2. Protocolos orientados a bit
3. Corrección de errores
  1. Códigos de control de errores
  2. Códigos polinómicos
4. Técnicas de control de flujo y protocolos de control de errores

# Próximo día

---

5. Protocolos de Control de Acceso al Medio

6. Direccionamiento

7. Hubs, Puentes y Conmutadores

1. Hubs

2. Puentes y conmutadores

1. Tabla de direcciones

2. Filtrado, reenvío e inundación

3. Procesos a realizar por un conmutador y sus prioridades

4. Cut-Through vs. Store-and-Forward

**8. Protocolo STP**

9. Ejemplo de protocolos de nivel de enlace de datos