

Universidad de Alcalá

MPLS

MultiProtocol Label Switching

Ingeniería de Tráfico — GITT

Introducción I

- Mecanismo de reenvío de paquetes de manera **conectiva** en redes **no conectivas**.
  - **Conexión**: establecimiento de **estado** en cada nodo.
- Tecnología que **separa** las dos funciones básicas de los *routers* (Figura 1):
  - **Plano de control**: Aprendizaje y selección de rutas…
    - … mediante **protocolos de encaminamiento** (e.g. OSPF).
    - … que requiere gestionar **tablas de encaminamiento**.
  - **Plano de datos**: Reenvío de paquetes…
    - … mediante **etiquetas** previamente distribuidas.
    - … que requiere gestionar **tablas de conmutación de etiquetas**.

Arquitectura de un *router*

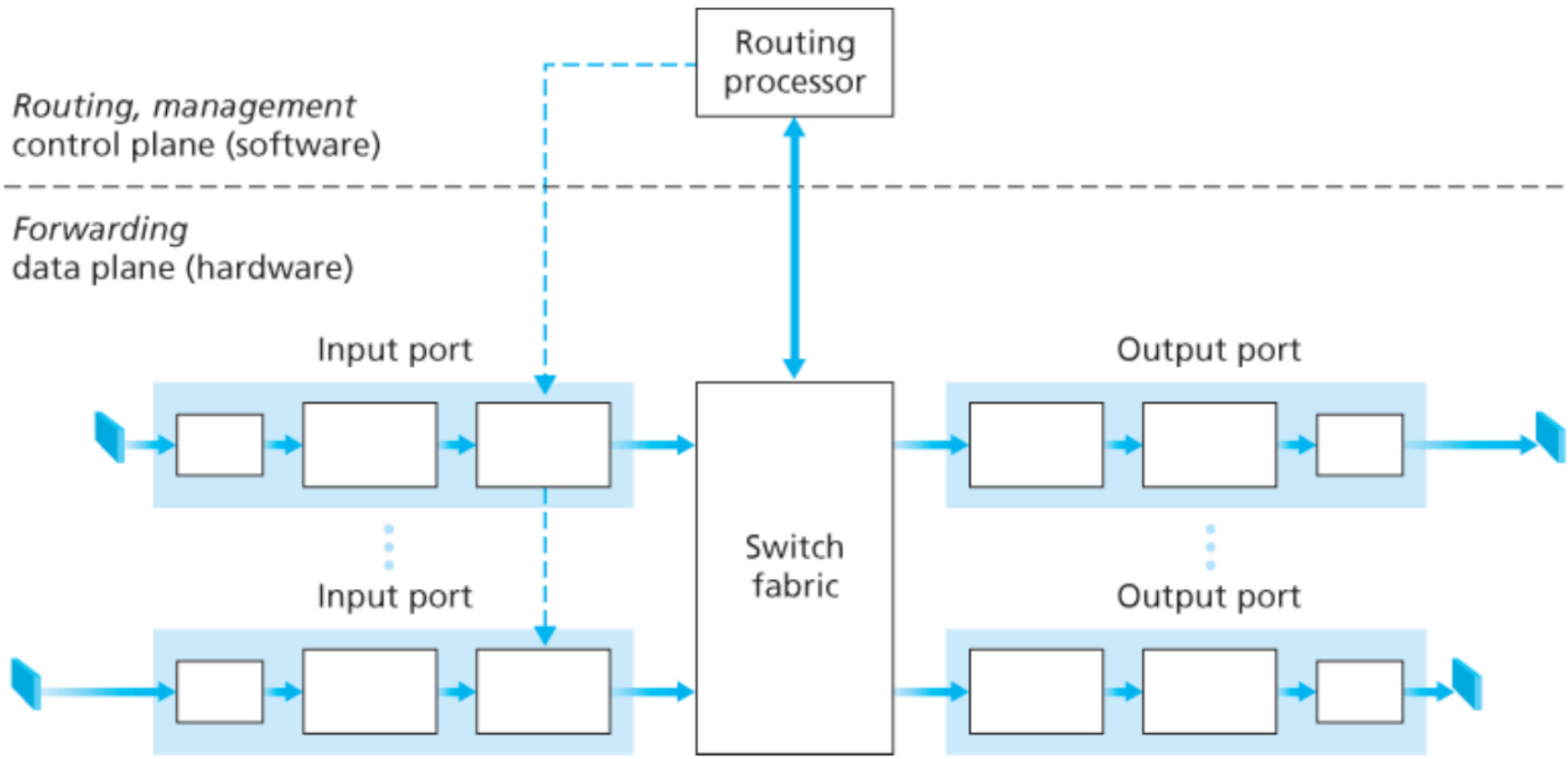


Figura 1: Obtenida de [1].

Introducción II

- MPLS **combina** lo mejor de las tecnologías **ATM** e **IP**:
  - **ATM**: **Eficiencia** de circuitos virtuales (VCs) y rutas preestablecidas.
  - **IP**: **Flexibilidad** para gestionar automáticamente redes grandes y dinámicas.
- MPLS es una tecnología **multiprotocolo**:
  - En L3 solo para **IPv{4,6}**; en L2 para **PPP**, **Ethernet**, **ATM**, **FR**.
- Motivación inicial: acelerar la operativa de los *routers*:
  - La conmutación en IP emplea **prefijos** de direcciones longitud **variable**.
    - Hace uso de tablas de encaminamiento: Forwarding Information Base (FIB).
    - Búsqueda en las FIB según destino lenta… ¡TCAMs solucionan el problema!
  - La conmutación MPLS emplea etiquetas de longitud fija:
    - Hace uso de tablas de conmutación de etiquetas: Label FIB (LFIB).
    - $L_{\text{LFIB}} \ll L_{\text{FIB}} \rightarrow T_{\text{LFIB}} \ll T_{\text{FIB}}; \quad L \equiv \text{longitud}, T \equiv \text{tiempo de búsqueda}.$

Una *celda* ATM

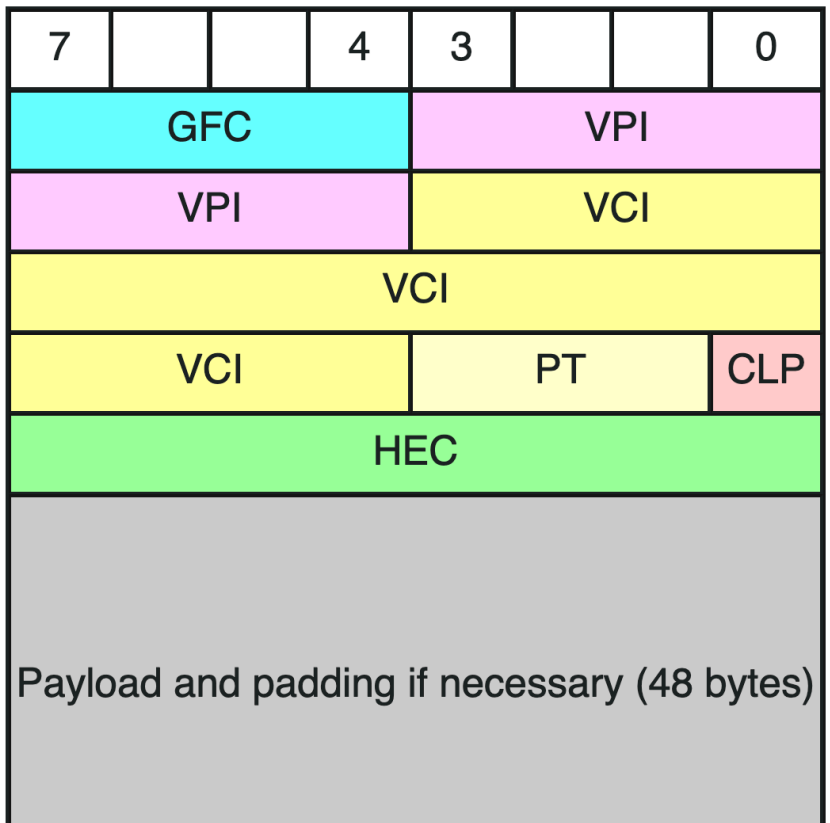


Figura 2: Los campos VPI y VCI identifican un VC [2].

Introducción III

- Ahora interesan las **aplicaciones** de MPLS: VPNs, *Traffic Engineering* (TE) y QoS.
- Encaminamiento y reenvío en contextos IP:
  - A partir de tablas de encaminamiento estáticas o dinámicas en cada *router*.
  - Basado en Forward Equivalency Classes (FECs):
    - FEC: grupo de datagramas tratados de igual manera respecto al reenvío.
    - Normalmente FECs en función del destino, aunque pueden usarse otros campos.
- Encaminamiento y reenvío en MPLS:
  - A partir de tablas de conmutación de etiquetas (tablas MPLS) en cada *router*.
  - Basado en FEC - MPLS: paquetes con igual etiqueta siguen la misma ruta y tratamiento.
  - Reenvío más ligero, rápido y **flexible** que el reenvío IP convencional.

Definiciones I

- **MPLS**: procedimiento de reenvío de paquetes basado solo en etiquetas.
- **PDU-MPLS**: resultado de encapsular una PDU_3 para encaminarla por un trayecto con MPLS.
- **LSR** (Label Switching Router): *router* IP con capacidad MPLS que puede:
  - Reenviar paquetes según *routing* IP convencional (i.e. FEC según destino).
  - Asociar etiquetas a FEC.
  - Reenviar paquetes mediante manejo de etiquetas.
- **Nodo MPLS**: nodo con capacidad MPLS que no tiene necesariamente capacidad de reenvío mediante FEC.
- **Dominio MPLS**: Conjunto de LSR y Nodos MPLS pertenecientes al mismo dominio de encaminamiento o administrativo.

Dominio MPLS

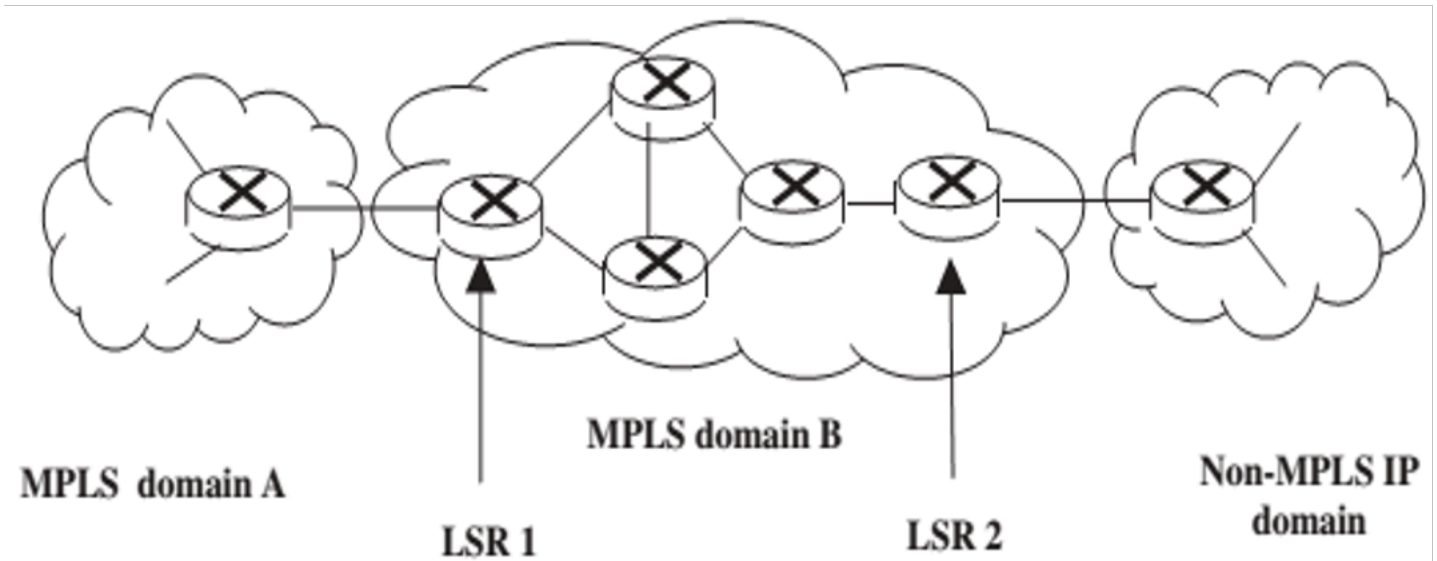
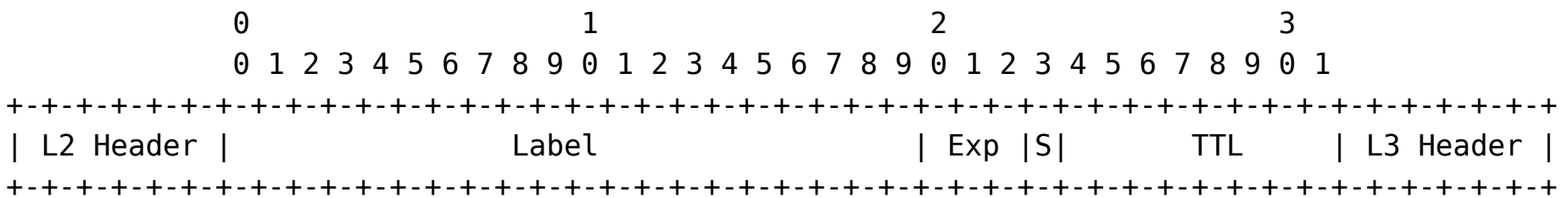


Figura 3: A los LSR de entrada y salida del dominio se les denomina LER (Label Edge Router) o LSR Frontera [3].

Definiciones II

- **Etiqueta**:
  - **Campo** de **longitud fija** añadido a los paquetes IP al **ingresar** a un dominio MPLS.
  - Relacionado con información relativa al **reenvío** dentro de un dominio MPLS:
    - Según destino, prioridad, pertenencia a una VPN, QoS, TE…
  - Se emplea como **puntero** a la LFIB en cada nodo MPLS. Cada entrada indica:
    - Siguiente “salto” del paquete (i.e. interfaz de salida).
    - Valor de la nueva etiqueta.
  - Se puede codificar de dos maneras:
    - Mediante campos existentes en L{2,3} redefinidos: FR, ATM e IPv6.
    - Mediante *shim header* para IPv4:
      - Define nuevos campos para la **cabecera MPLS**.

El *shim header*




```
Label:  Label Value, 20 bits
Exp:    Experimental Use, 3 bits, por ejemplo para indicar clases de servicio (CoS)
S:      Bottom of Stack, 1 bit
TTL:    Time to Live, 8 bits, para uso similar al campo homónimo de cabecera IP
```

Listado 1: Estructura del *shim header* [4].

MPLS en Linux v6.19.11 I

```
static inline struct mpls_entry_decoded mpls_entry_decode(
  struct mpls_shim_hdr *hdr)
{
  struct mpls_entry_decoded result;
  unsigned entry = be32_to_cpu(hdr->label_stack_entry);

  result.label = (entry & MPLS_LS_LABEL_MASK) >> MPLS_LS_LABEL_SHIFT;
  result.ttl   = (entry & MPLS_LS_TTL_MASK)   >> MPLS_LS_TTL_SHIFT;
  result.tc    = (entry & MPLS_LS_TC_MASK)    >> MPLS_LS_TC_SHIFT;
  result.bos   = (entry & MPLS_LS_S_MASK)     >> MPLS_LS_S_SHIFT;

  return result;
}
```

Listado 2: Decodificación de cabeceras MPLS (`net/mpls/af_mpls.c`).

MPLS en Linux v6.19.11 II

```
/* Reference: RFC 5462, RFC 3032
 *
 *  0                   1                   2                   3
 *  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
 * +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 * |                Label                  | TC  |S|       TTL     |
 * +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 *
 *  Label:  Label Value, 20 bits
 *  TC:     Traffic Class field, 3 bits
 *  S:      Bottom of Stack, 1 bit
 *  TTL:    Time to Live, 8 bits
 */

 #define MPLS_LS_LABEL_MASK      0xFFFFF000
 #define MPLS_LS_LABEL_SHIFT     12
 #define MPLS_LS_TC_MASK         0x00000E00
 #define MPLS_LS_TC_SHIFT        9
 #define MPLS_LS_S_MASK          0x00000100
 #define MPLS_LS_S_SHIFT         8
 #define MPLS_LS_TTL_MASK        0x000000FF
 #define MPLS_LS_TTL_SHIFT       0
```

```
#define MPLS_HLEN 4

struct mpls_shim_hdr {
  __be32 label_stack_entry;
};
```

Listado 3: Constantes auxiliares para decodificar cabeceras MPLS (`include/net/mpls.h` e `include/uapi/linux/mpls.h`).

Elixir

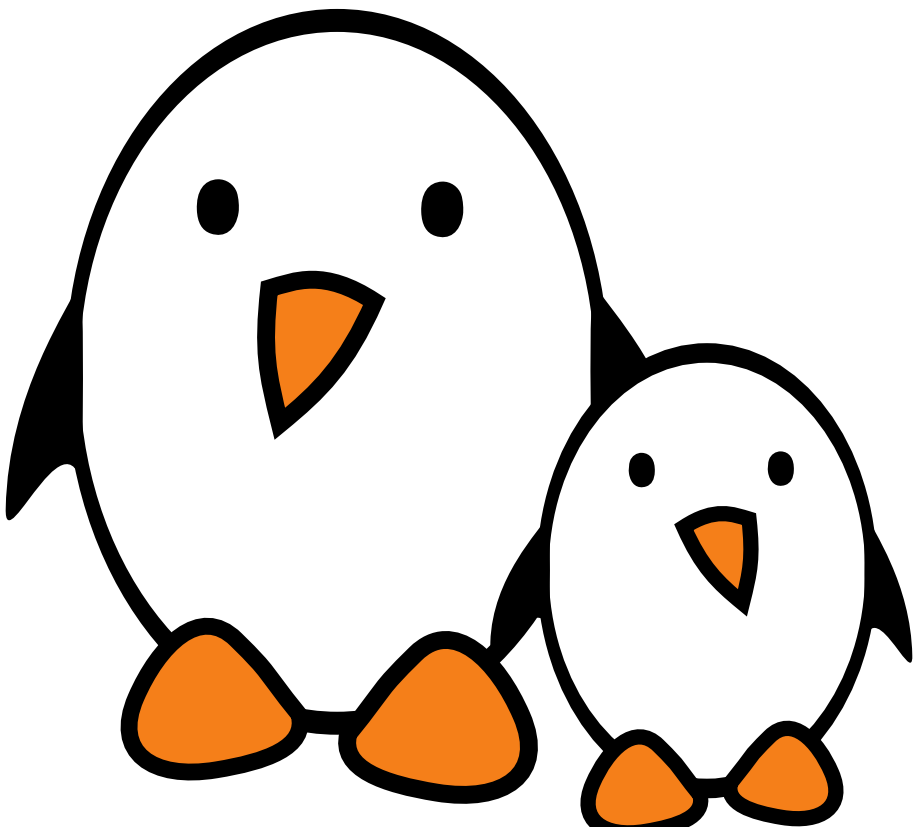


Figura 4: Kernel cross-referencer.

Bibliografía

[1] J. F. Kurose y K. W. Ross, *Computer networking: a top-down approach*, Seventh edition. Pearson, 2017.

[2] Wikipedia contributors, «Asynchronous Transfer Mode — Wikipedia, The Free Encyclopedia». 2026.

[3] «The Multi-Protocol Label Switching (MPLS) Architecture», en *Connection-Oriented Networks*, John Wiley & Sons, Ltd, 2005, chap. 6, pp. 131-148. doi: https://doi.org/10.1002/0470016361.ch6.

[4] D. Tappan *et al.*, «MPLS Label Stack Encoding». [En línea]. Disponible en: https://www.rfc-editor.org/info/rfc3032