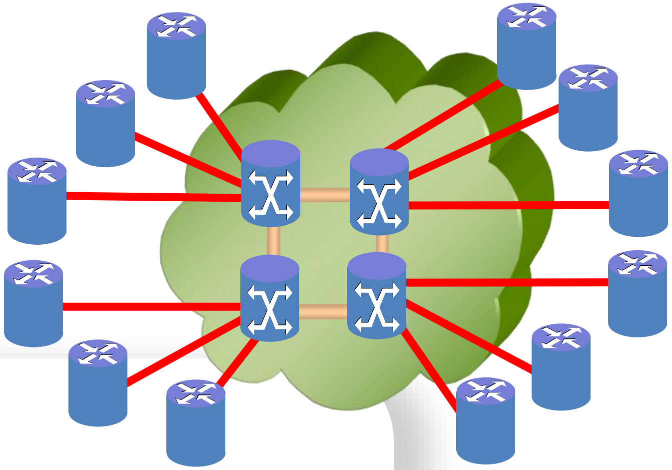
MPLS

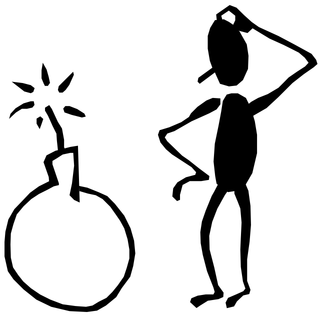


Problems with IP routing

Problems with IP routing

IP is great – but not perfect !

- scalability
  - router table overload
  - routing convergence slow-down
  - increase in queuing time and routing traffic
  - problems specific to underlying L2 technologies
- hard to implement load balancing
- QoS and Traffic Engineering
- problem of routing changes
- difficulties in routing protocol update
- lack of VPN services



Scalability

When IP was first conceived, scalability was not a problem
but as number of hosts increases, routing shows stress

Simplistic example

- *N* hosts
- each router serves M hosts
- each router entry takes **a** bytes

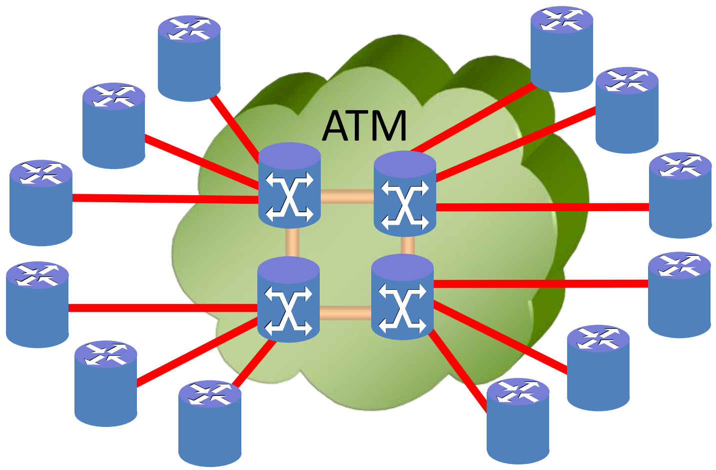
hence

- router table size **a** N ~ *N*
- *N* / *M* ~ *N* routers (more routers => slower convergence)
- packet processing time* ~*N* (since have to examine entire table)
- ~*N* routers send to ~*N* *routers* tables of size ~*N*
  so routing table update traffic increases $\sim N^3$ (or $\sim N^4$)

L2 Backbone Doesn't Help!

Instead of expensive and slow IP routers

core once used faster/cheaper Asynchronous Transfer Mode (CO) switches



Classical IP over ATM

but this actually makes things worse!

ATM switches are transparent to routers

ATM switches do not participate in IP routing protocols

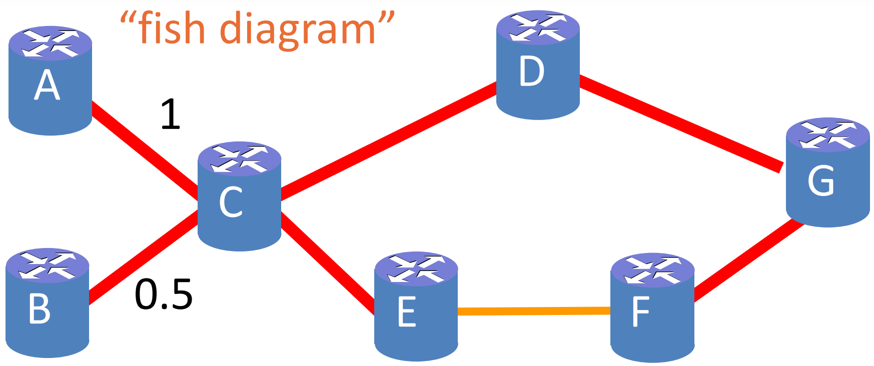
so every IP router becomes logically adjacent to every other

and we need ~ N^2 ATM VCs !

If only the ATM switches could understand IP routing protocols…

Unfortunately, this is impossible (without label switching)!

Load Balancing



traffic from A to G = 1Gb
traffic from B to G = 500Mb
all links 1Gb except EF - 500 Mb

Were C,D,E, and F CO switches there would be no problem
(1Gb over ACDG, 500Mb over BCEFG)

With standard IP hop-count cost function, all traffic over CDG
resulting in 1.5Gb there (congestion) and CEFG idle

With administrative cost on CDG we can force all the traffic to CEFG
even worse congestion !

Finally with administrative cost and Equal Cost MultiPath (ECMP)
750 Mb over CDG and CEFG, link EF is still congested !

It would be great if we could add **T**raffic **E**ngineering to IP

Unfortunately, this is hard (without label switching)!

QoS and TE

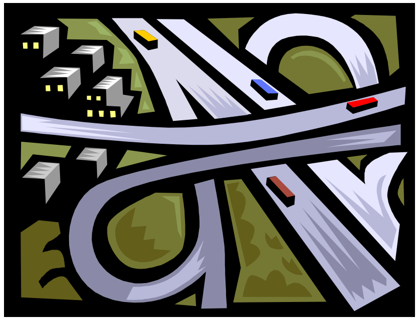
CL networks can not guarantee path QoS
- you can't reserve resources for handling a packet
- since you don't know where the packet is going to go !

But other protocols in the IP suite can help
- TCP adds CO layer compensating for loss and mis-ordering
  - but that is correcting for mishaps that already occurred
- IntServ (RSVP) sets up path with reserved resources
  - but that modifies IP so much that it never caught on
- DiffServ (DSCP) prioritizes packets
  - but that doesn't provide any guarantees

So IP network managers use **N**etwork **E**ngineering
- throw BW at the problem

rather than **T**raffic **E**ngineering
- optimally exploiting the BW that is there

TE is great, but not possible in IP (without label switching)!



Routing Changes

IP routing is satisfactory in the steady state
but what happens when something changes?

Any change in routing information
(new router, router failure, new inter-router connection, etc)
necessitates updating of tables of all routers

Convergence is generally slow

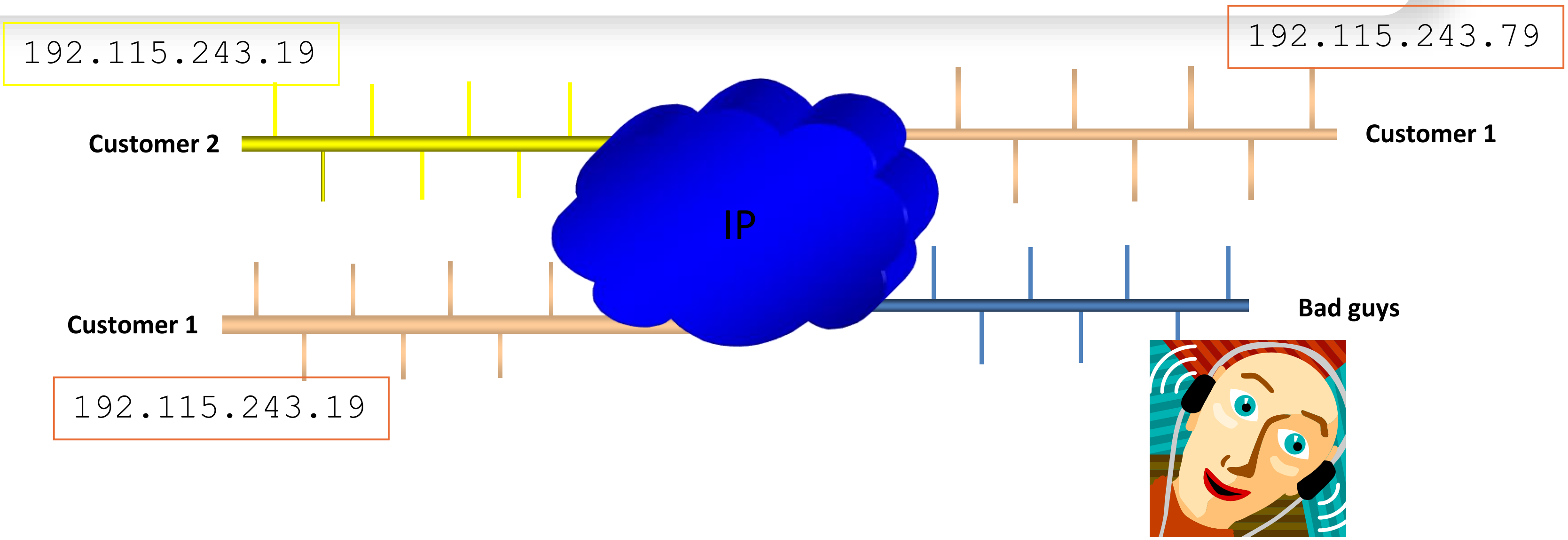
A change in the routing protocol is even worse
(e.g. Bellman-Ford to present, classed to classless, IPv4 to IPv6)
since it necessitates upgrade of all router software

Upgrades may have to be simultaneous

What we need is a more complete separation
of forwarding from routing functionality (ForCES)

Unfortunately, this separation is only recently starting to appear in IP!

VPN Services

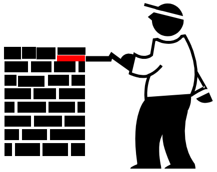


IP was designed to **inter**connect **net**works
not to provide VPN services

When we connect routers from different customers
the security isolation is weak

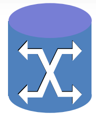
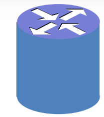
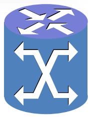
LANs may use non globally unique addresses (present solution - NAT)

Interconnect may entail complex provider-customer relationships



Label Switching

Solution - Label Switching

CO forwarder (switch)  CL forwarder (router)  LSR 

label switching adds the strength of CO to CL forwarding

label switching involves three stages:

- routing (topology determination) using L3 protocols
- path setup (label binding and distribution) perhaps using new protocol(s)
- data forwarding

label switching the solution to *all* of the above problems

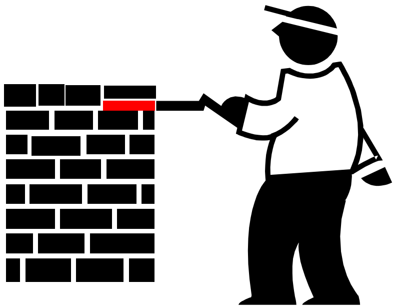
- speeds up forwarding
- decreases forwarding table size (by using local labels)
- enables support for QoS and arbitrary granularity FECs
- load balancing by explicitly setting up paths
- complete separation of routing and forwarding algorithms
  - no new routing algorithm needed
  - but new signaling algorithm may be needed

Where is it?

Unlike TCP, the label switching CO layer lies *under* the CL layer

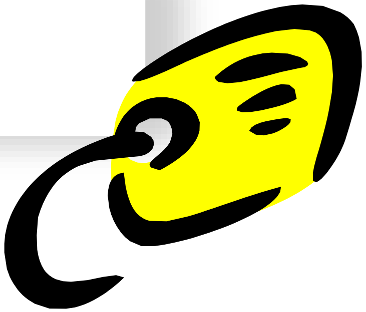
If there is a broadcast L2 (e.g. Ethernet), the CO layer lies *above* it

higher layers
layer 3 (e.g. IP)
label switching (layer 2.5) ← shim header
layer 2 (e.g. ethernet)
physical layer



Hence, label switching is sometimes called layer 2.5 switching

Labels



A label is a short, fixed length, structure-less address

The following are not labels:

- telephone number (not fixed length, country-code+area-code+local-number)
- Ethernet address (too long, note vendor-code is *not* meaningful structure)
- IP address (too long, has fields)
- ATM address (has VP/VC)

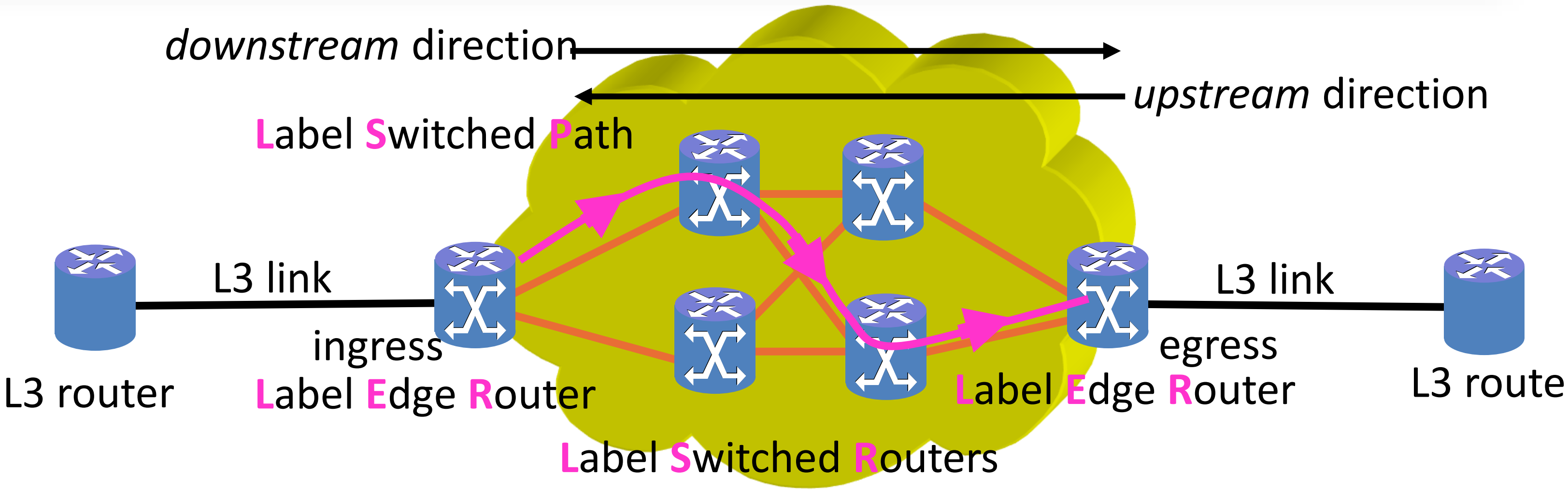
Not an explicit requirement, but normally only ***local*** in significance

Label(s) added to CL packet, *in addition* to L3 address

Layer 2.5 forwarding

- requires a flow setup process and signaling protocol
- may find a different route than the L3 forwarding
  - and thus support higher granularity FECs
- may be faster than L3 forwarding

Label Switching Architecture



Label switching is needed in the **core**, access can be L3 forwarding*

Core interfaces the access at the edge (ingress, egress)

LSR router that can* perform label switching

LER LSR with non-MPLS neighbors (LSR at edge of core network)

LSP unidirectional path used by label switched forwarding (ingress to egress)

* not every packet needs label switching

Label Switched Forwarding

LSP needs to be setup before data is forwarded
and should be torn down once no longer needed

LSR performs
- label switched forwarding* for labeled packets

label space may be
- per platform (unique to LSR) or
- per port (unique to input interface - like ATM)

LSRs optionally support L3 forwarding for unlabeled packets

ingress LER
- assigns packet* to FEC
- labels packet
- forwards it *downstream* using label switching

*** once packet is assigned to a FEC and labeled no other LSR looks at the L3 headers**

egress LER
- removes label
- forwards packet using L3 forwarding
- exception: PHP (discussed later)

Hierarchical Forwarding

Many networks use hierarchical routing
- decreases router table size
- increases forwarding speed
- decreases routing convergence time

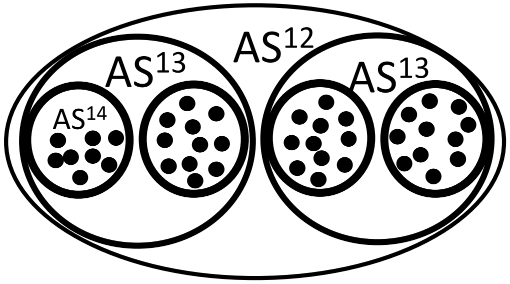
telephone numbers	*Country-Code*	*Area-Code*	*Exchange-Code*	*Line-Number*
	972	**2**	**588**	**9159**

Internet URLs

host … *SLD TLD*
myrad . rad . co . il

Ethernet/802.3 address space is flat (even though written in byte fields)

IP can even support arbitrary levels of hierarchy
by hierarchical ASes and advertising aggregated addresses
but the exploitation is not optimal



Label Stacks

Since labels are structure-less, the label space is flat

Label switching can support arbitrary levels of hierarchy by using a *label stack*

top label
another label
yet another label
bottom label

Label forwarding based only on top label

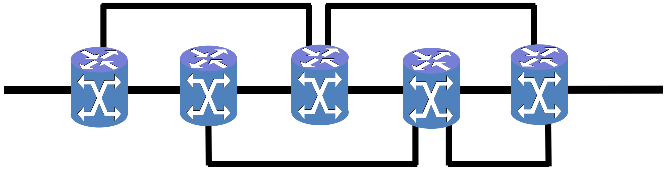
Before forwarding, three possibilities (listed in NHLFE) :

- read top label and pop
- read top label and swap
- read top label, swap, and push new label(s)

Example Uses of Label Stack

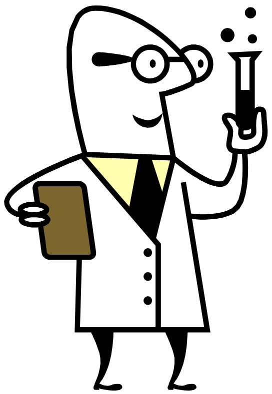
Example applications that exploit the label stack

- fast rerouting



- VPNs
- PWE^3

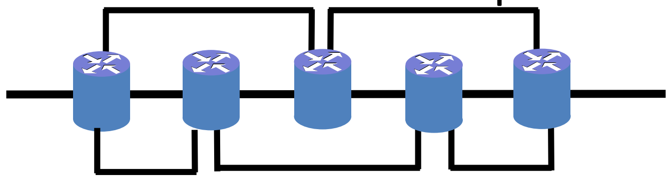
Note: three labels is usually more than enough



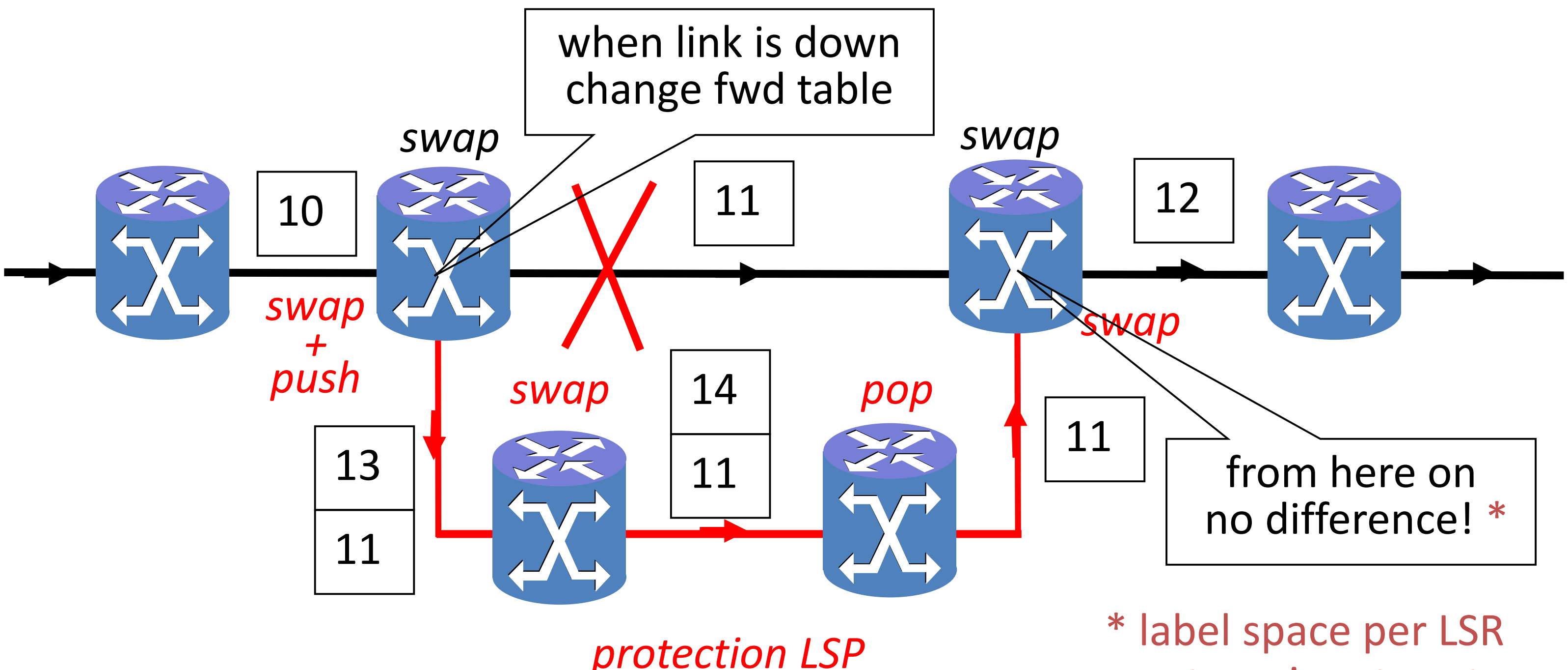
Fast ReRoute

IP has no inherent recovery method (like SDH)

in order to ensure resilience we can provide local detours



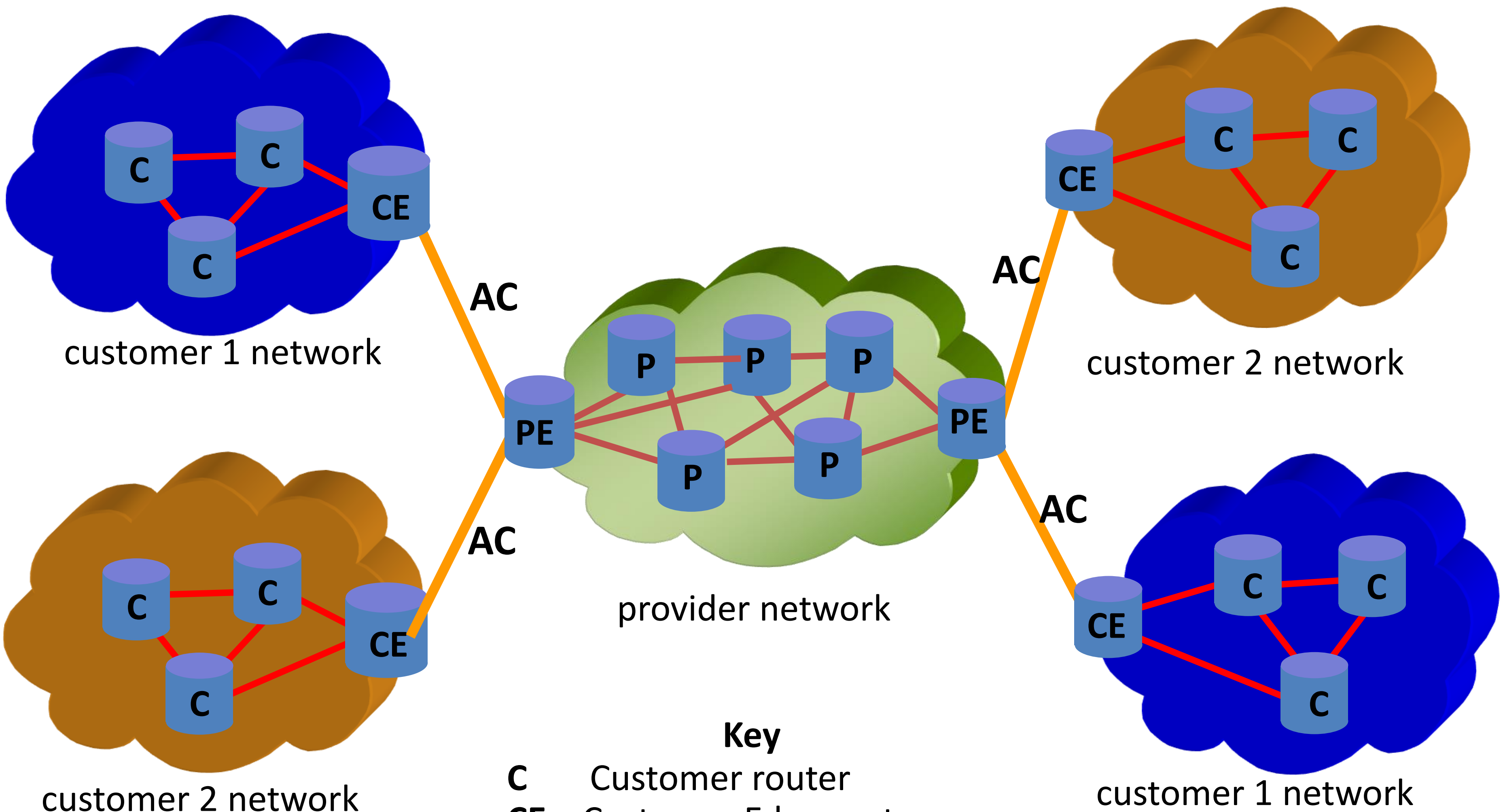
to reroute quickly we *pre*-prepare labels for the bypass links



* label space per LSR not per input port

to bypass a failed link we need to reach the **N**ext **H**op (NH)
to bypass a failed LSR we need to reach the **N**ext **N**ext **H**op (NNH)

Label Switched VPNs



Key

C Customer router
CE Customer Edge router
P Provider router
PE Provider Edge router

Label Switched VPNs (cont.)

If customers 1 and 2 use overlapping IP addresses
C-routers have incompatible tables

Ingress PE (LER) inserts two labels

egress PE
egress CE
IP header
payload

Only PEs know about customers

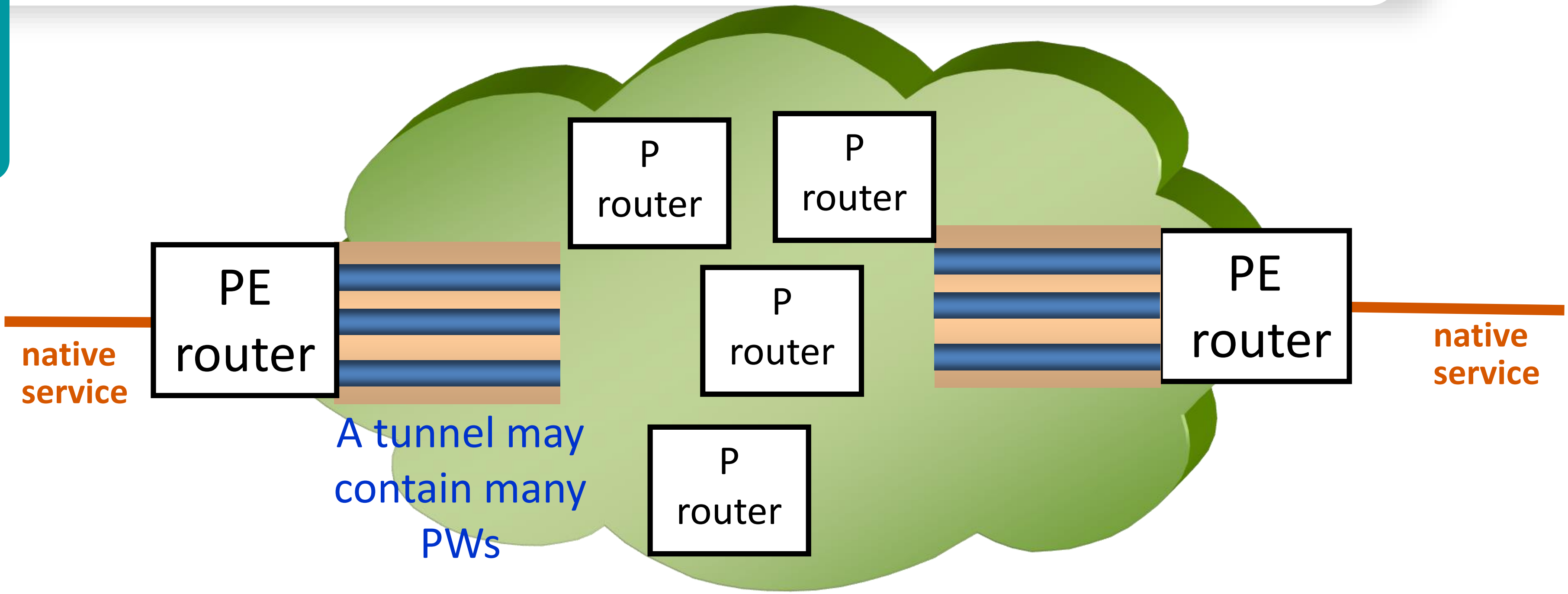
P-routers see only the label of the egress PE-router

- P-routers don’t see IP addresses, so there is no ambiguity
- they don’t know about VPNs at all
- no need to understand customer configuration
- smaller tables
- no rerouting if customer reconfigures

Ingress PE router only knows about CE routers

- no need to understand customer configuration (C-routers)

Pseudowires



PW label is not a *real* label
- it just identifies native service instance

P routers don't know about PWs
- just how to get to egress PE

With MS-PWs, PW labels becomes real labels

tunnel label
PW label
PW control word
payload

Label (20b)	TC(3b)	S(1b)	TTL (8b)

MPLS

MPLS history

Many different label switching schemes were invented

- **C**ell **S**witching **R**outer (Toshiba) <RFC 2098,2129>
- IP Switching (Ipsilon, bought by Nokia) <RFC 2297>
- Tag Switching (Cisco) <RFC 2105>
- **A**ggregate **R**oute-based **I**P **S**witching (IBM)
- IP Navigator (Cascade bought by Ascend bought by Lucent)

so the IETF decided to standardize a *single method*

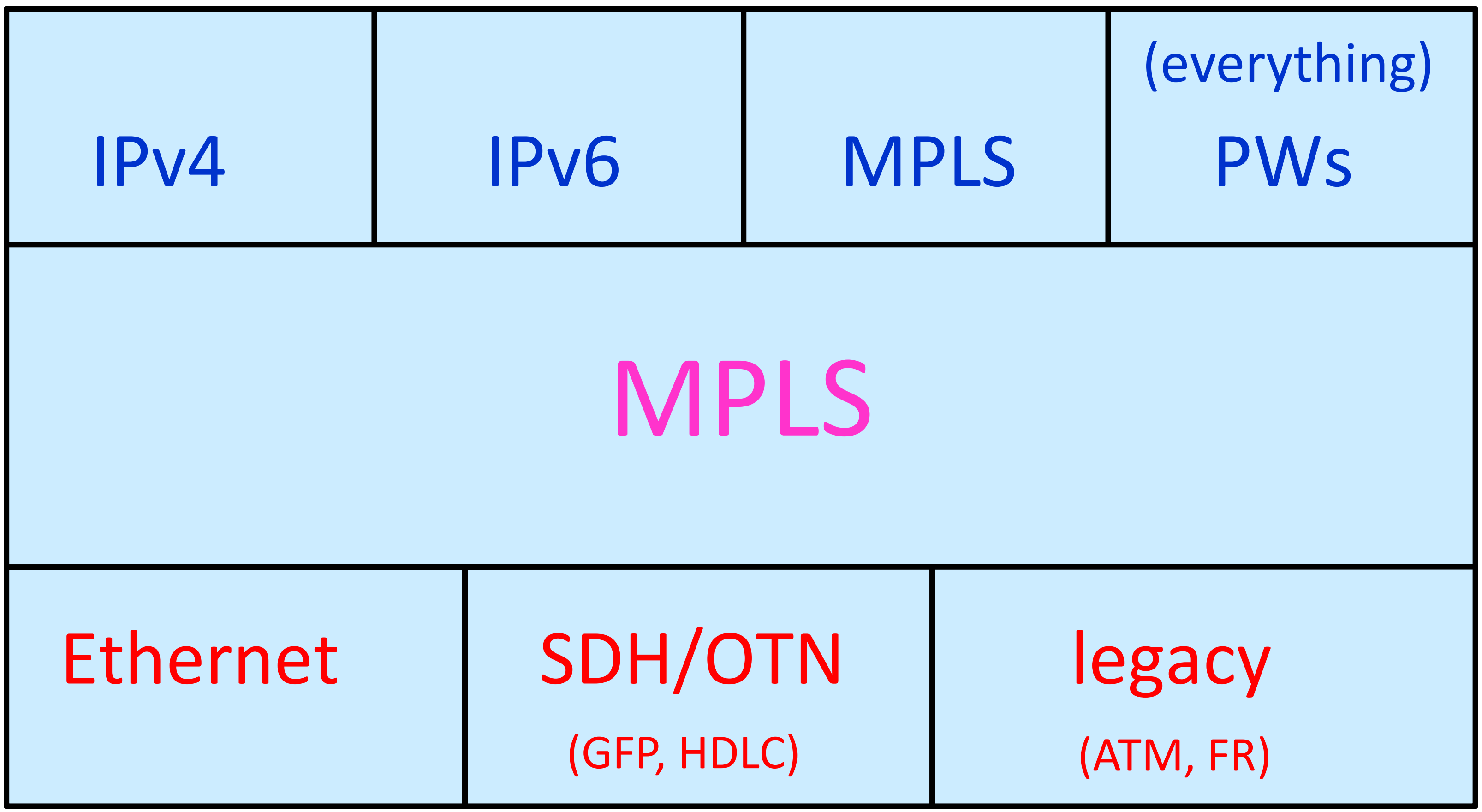
This method is called MPLS

MPLS

Of all possible label switching technologies
what is special about MPLS ?

- **m**ulti**p**rotocol - from above and below
- label shim header (label may also be in legacy L2)
- single forwarding algorithm, including for multicast and TE
- control plane consisting of
  - topology discovery via IP routing protocols
  - and label distribution via
    - piggybacked on existing routing protocols
    - via the **L**abel **D**istribution **P**rotocol
    - via RSVP-TE (and historically CRLDP) for Traffic Engineering

MultiProtocol Label Switching



MPLS *Shim* Header

Label (20b)	TC(3b)	S(1b)	TTL (8b)

The shim format is:

Label there are 2^{20} different labels **(+ 2^{20} multicast labels)**

Traffic Class (ex-EXP)

- was **CoS** in Cisco Tag Switching
- may influence packet queuing
- QoS may be via E-LSP or L-LSP

Stack bit S=1 indicates bottom of label stack

TTL decrementing hop count
- used to eliminate infinite routing loops
- and for MPLS traceroute
- generally copied from/to IP TTL field

Special (reserved) labels

0	IPv4 explicit null
1	router alert
2	IPv6 explicit null
3	implicit null
13	MPLS-TP GAL
14	Y.1711 OAM label

S=0	top label
S=0	another label
S=0	yet another label
S=1	bottom label

Single Forwarding Algorithm

IP uses different *forwarding* algorithms

for unicast, unicast w/ ToS, multicast, etc.

L**S**R uses one *forwarding* algorithm (L**E**R is more complicated)

- read top label *L*
- consult **I**ncoming **L**abel **M**ap (forwarding table) [Cisco terminology LFIB]
- perform label stack operation (pop *L*, swap *L* - *M*, swap *L* - *M* and push *N*)
- forward based on *L*'s Next Hop Label Forwarding Entry

NHLFE contains:

- next hop (output port, IP address of next LSR)
  - if next hop is the LSR itself then operation must be pop
  - for multicast there may be multiple next hops, and packet is replicated
- label stack operation to be performed
- any other info needed to forward (e.g. L2 format, how label is encoded)

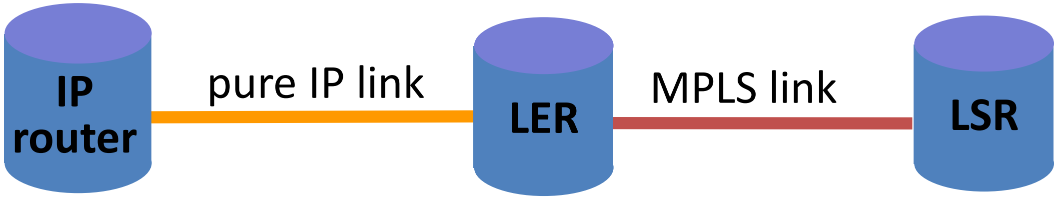
ILM contains:

- a NHLFE for each incoming label
- possibly multiple NHLFEs for a label, but only one used per packet

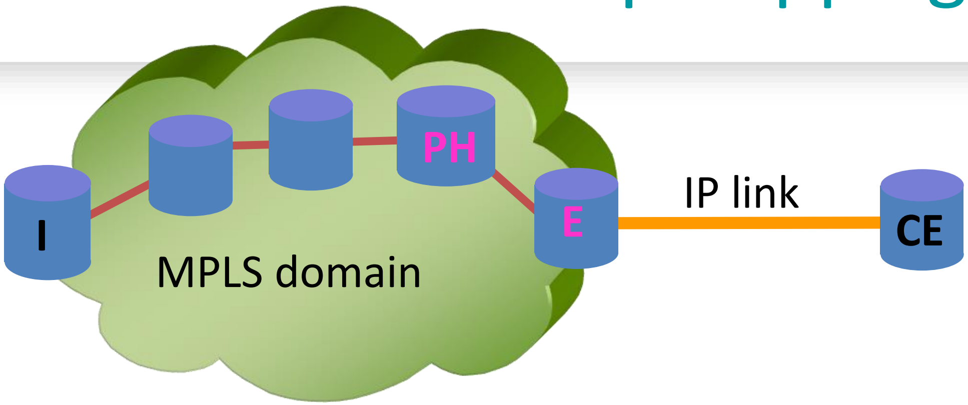
LER Forwarding Algorithm

LER's forwarding algorithm is more complex

- check if packet is labeled or not
- if labeled
  - then forward as LSR
  - else
    - lookup destination IP address in FEC-To-NHLFE Map [Cisco terminology LIB]
    - if in FTN
      - then prepend label and forward using LSR algorithm
      - else forward using IP forwarding



Penultimate Hop Popping



The egress LER E also may have to work *overtime*:

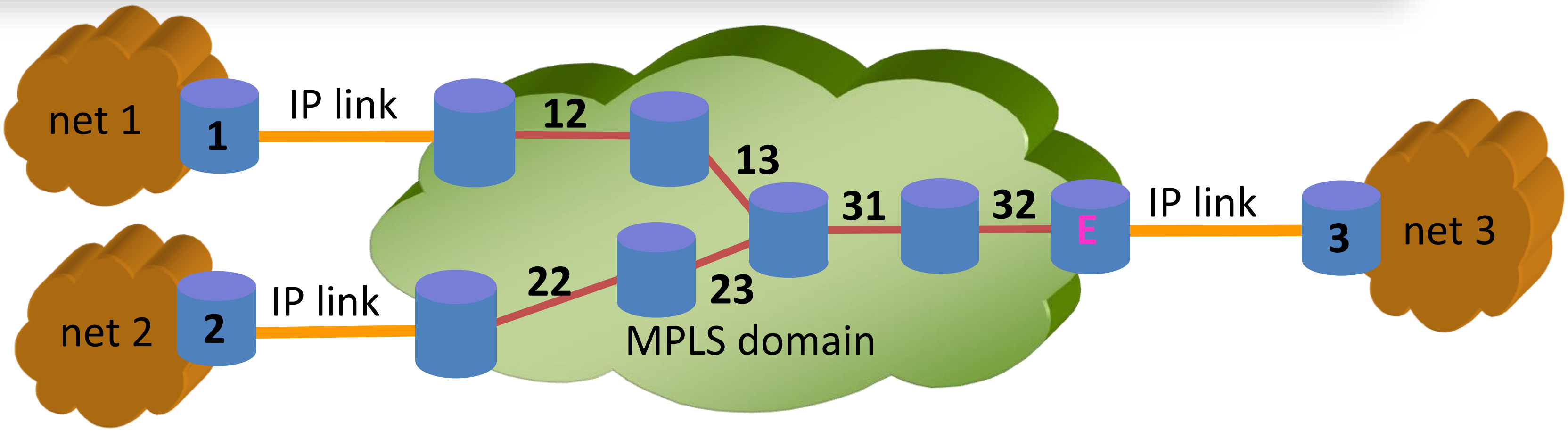
- read top label
- lookup label in ILM
- find that in NHLFE that the label must be popped
- lookup IP address in IP routing
- forward to CE using IP forwarding

We can save a lookup (and the first 3 steps) by performing PHP but pay in loss of OAM capabilities

penultimate LSR PH performs the following:

- read top label
- lookup label in ILM
- pop label revealing IP address of CE router
- forward to CE using IP forwarding

Route Aggregation (VC merge)



Traffic from both network 1 and network 2 is destined for network 3

Scalability advantages

- fewer labels
- conserve table memory

Disadvantages

- IP forwarding may be required
- OAM backwards trail is destroyed

Data and control planes

control plane

- all IP routing protocols (OSPF, BGP, etc)
- procedure to bind label to FEC (label assignment)
- protocol to distribute label binding information
- procedure to create forwarding table

user (data) plane

- procedure to label incoming packet
- forwarding procedure
  - forwarding table lookup
  - label stack operations

Label Distribution Protocols

When an LSR creates/removes a FEC - label binding
it needs to inform other LSRs of its decision

MPLS allows piggybacking label distribution on routing protocols

- protocols already in use (don't need to invent or deploy)
- eliminates race conditions (when route or binding, but not both, defined)
- ensures consistency between binding and routing information
- only for distance vector or path vector routing protocols (not OSPF, IS-IS)
- not all routing protocols are sufficiently extensible (RIP isn't)
- has been implemented for **BGP**-4

MPLS WG invented a new protocol **LDP** for "plain" label distribution

- messages sent reliably using TCP/IP
- messages encoded in TLVs
- discovery mechanism to find other LSRs

… and extended RSVP to LSPs for QoS - **RSVP-TE**

New approach – use of **OpenFlow** for LSP set-up

MPLS flavors

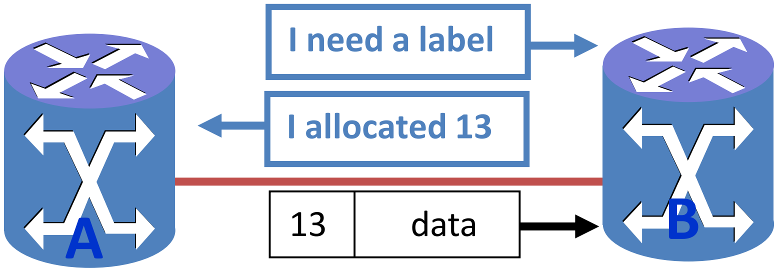
We can now distinguish *four* flavors of MPLS :

1. plain **vanilla MPLS** (usually with LDP, perhaps with RSVP-TE for FRR)
   not true CO – pinned to route not to NEs
   used in Internet core

2. MPLS for **L3VPN** services (RFC 4364 <ex-2547> using BGP)
   used to deliver VPN services to businesses

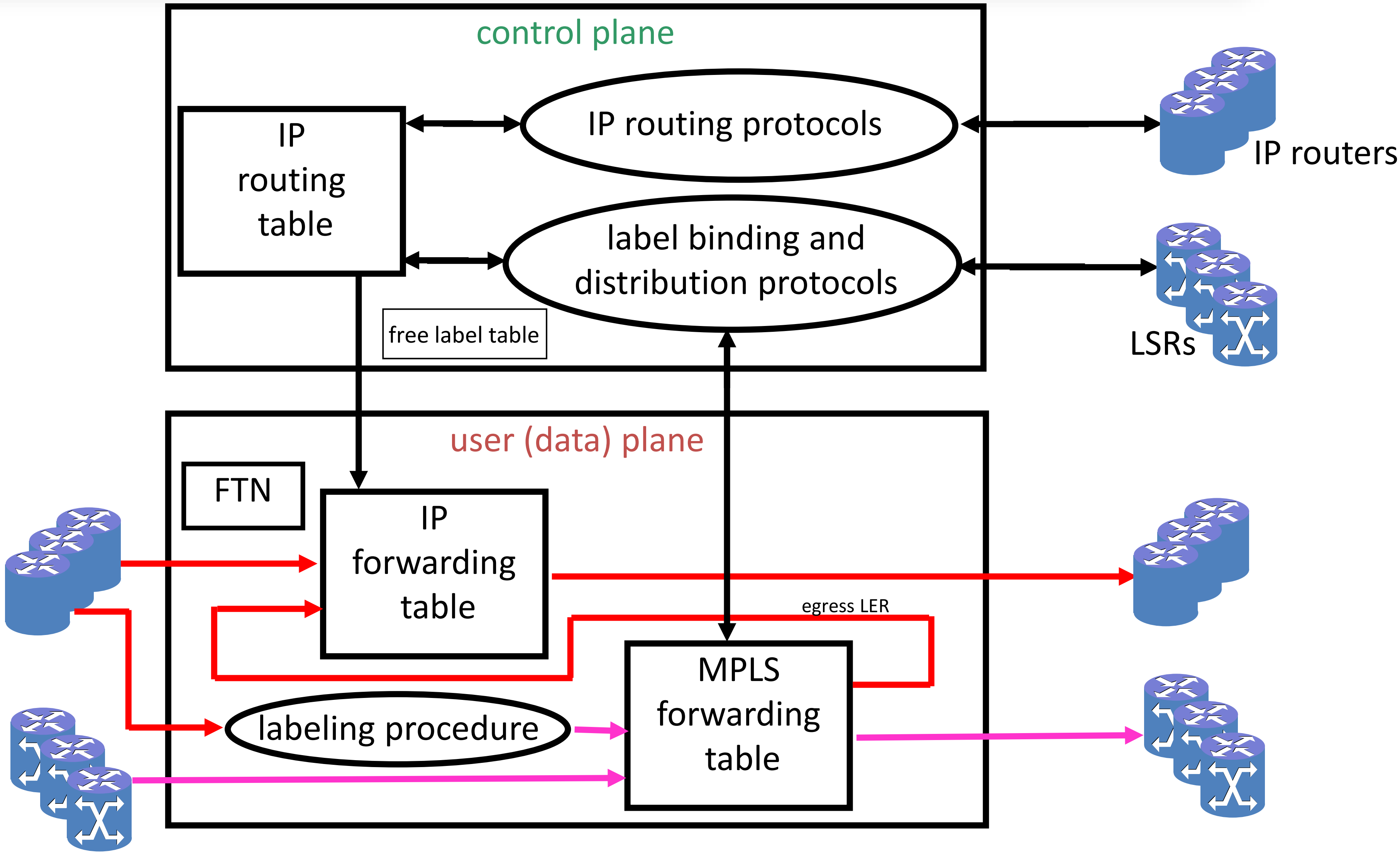
3. MPLS-TE (currently with RSVP-TE)
   true CO with resource reservation
   used when SLA guarantees given

4. MPLS-TP (usually with management system, can use RSVP-TE)
   does not assume the existence of IP forwarding plane
   does not require control plane – can work with management OSS
   implements OAM and APS functionality



MPLS control plane

LER Architecture



All the Tables

FEC table

FEC	protocol	input port	handling
192.115/16	IPv4	2	best-effort

Free Labels 128-200 presently free

FTN

FEC	port/label in	port/label out
192.115/16	2/17	3/137

ILM

NHLFE

port/label in	port/label out	next hop	operation
2/17	3/137	5.4.3.2	swap

Binding and Distribution Options

label binding (assignment)

- per port or per LSR label space
- control driven vs. data driven (traffic driven)
- liberal vs. conservative label retention

label distribution (advertisement)

- downstream vs. upstream
- downstream on-demand (dod) vs.
  downstream unsolicited (du)
- independent vs. ordered

Per Port Label Space

LSR may have a separate label space for each input port (I/F)

or a single common label space

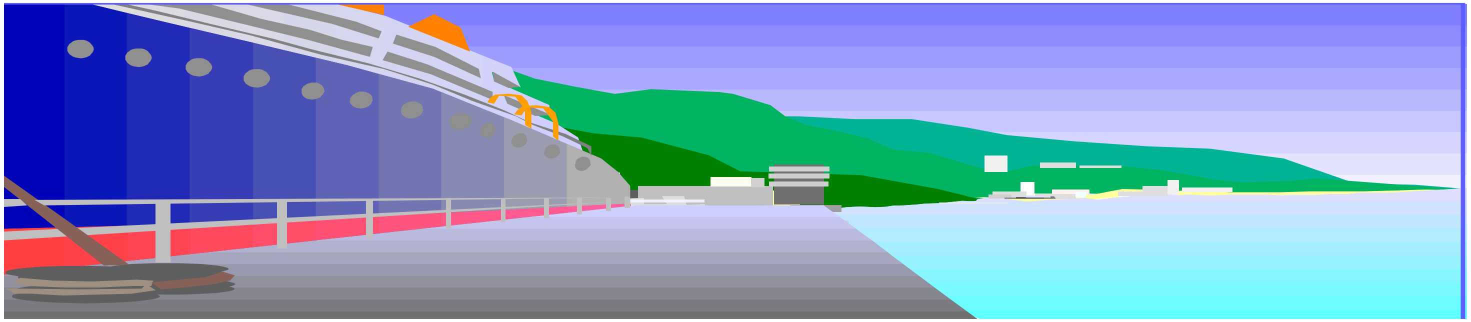
or any combination of the two

Separate labels spaces means separate forwarding tables per port

ATM LSR had only per port label spaces (leads to interleave problem)

per port label spaces increases number of available labels

common label space facilitates several MPLS mechanisms (e.g. FRR)



Control vs. Data Driven

there are two philosophies as to when to create a binding

data-driven (traffic-driven) binding (Toshiba CSR, Ipsilon IP-Switching)

automatically create binding when data packets arrive

(from first packet?, after enough packets? when tear LSP down?)

control-driven binding (Cisco Tag Switching, IBM ARIS)

create binding when routing updates arrive

(only update when topology changes? update upon request?)

Although not specifically stated in the architecture document MPLS assumes control driven binding *

* two implementations of control driven are possible:

- topology-driven (routing tables are consulted)
- control-traffic driven (only routing update messages are used)

Liberal vs. Conservative Retention

LSR receives "advertisements" (label distribution messages)
from other LSRs

conservative label retention
LSR retains only label-to-FEC bindings that are presently needed

liberal label retention
LSR stores all bindings received (more labels need to be maintained)

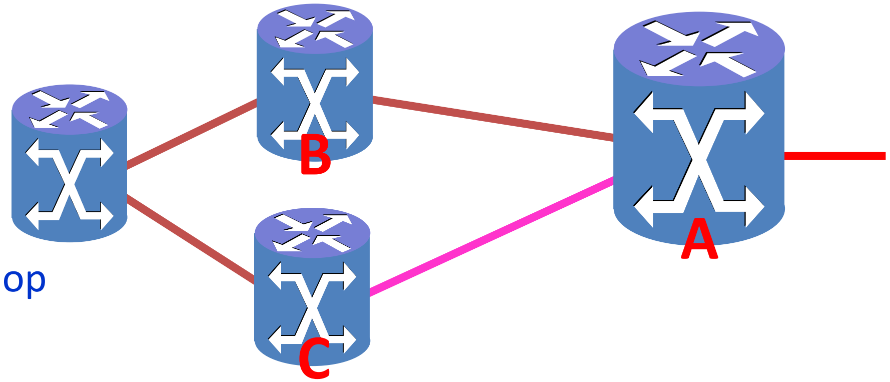
using liberal retention can speed response to topology changes

LSRs must agree upon mode to be used

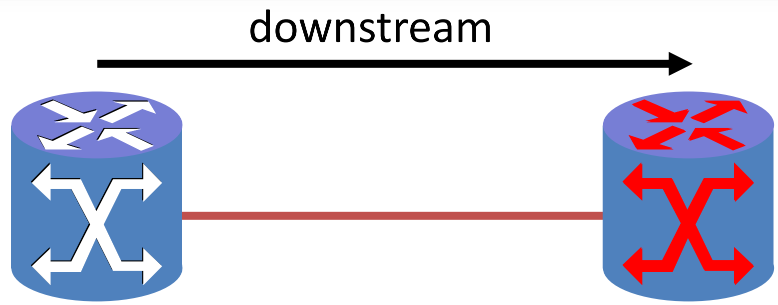
A advertises label

B is previous hop LSR
but C retains label anyway

later routing change makes C the previous hop
C immediately can start forwarding



Downstream vs. Upstream



binding means to allocate a label to a FEC

LSR allocates the label from “free label” pool

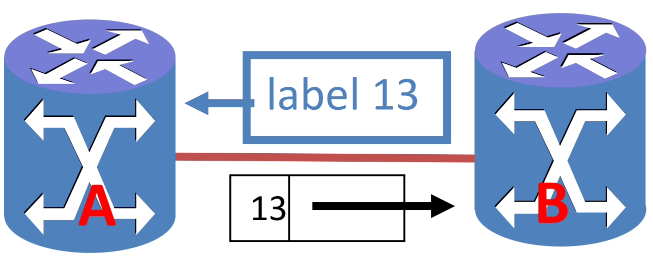
Which LSR allocates the label ?

Unicast MPLS uses downstream binding

- label allocated by LSR downstream from the LSR that prepends it
- label distribution information flows upstream
  reverse in direction from data packets

To set up LSP through link from LSR A to LSR B :

- LSR B binds label 13 to FEC
- B advertises label to LSR A
- LSR A sends packets with label 13 to B



On-demand vs. Unsolicited

downstream on-demand label distribution

LSRs may explicitly request a label from its downstream LSR

unsolicited label distribution

LSR distributes binding to upstream LSR w/o a request
(e.g. based on time interval, or upon receipt of topology change)

LSR may support on-demand, unsolicited, or both

adjacent LSRs must agree upon which mode to be used

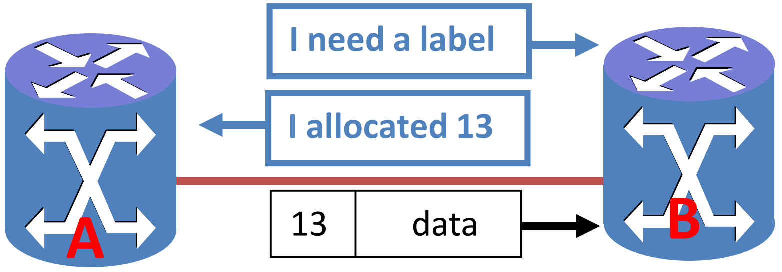
LSR A needs to send a packet to LSR B

LSR A requests a label from LSR B

B binds label 13 to the FEC

B distributes the label

A starts sending data with label 13



Independent vs. Ordered

independent binding (Tag Switching)

– each LSR makes independent decision to bind and distribute

ordered binding (ARIS)

- egress LSR binds first and distributes binding to neighbors
- LSR that believes that it should be the penultimate LSR binds and distributes to its neighbors
- binding proceeds in orderly fashion until ingress LSR is reached

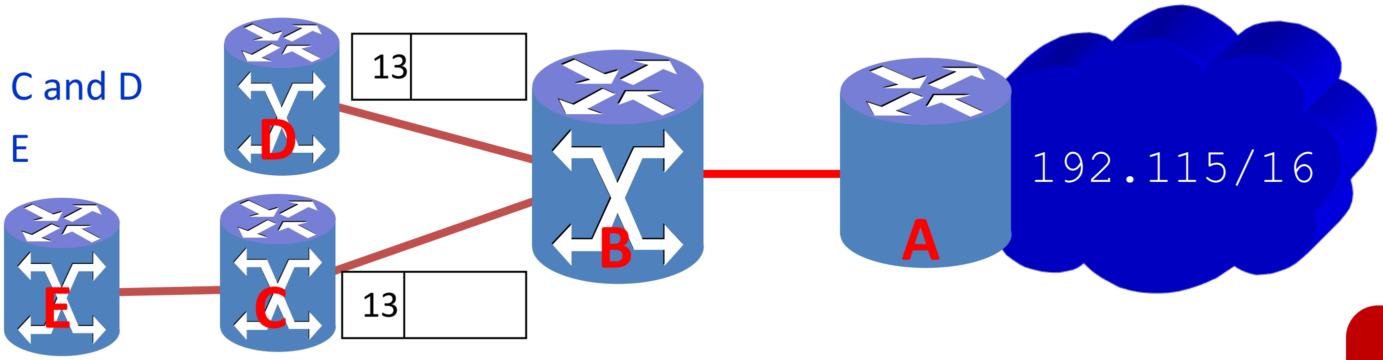
LSRs must agree upon mode to be used

B sees that it is egress LSR for 192.115.6

B allocates label 13

B distributes label to C and D

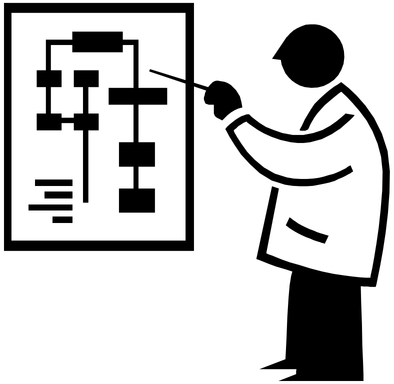
C distributes label to E



LDP tasks

A **label distribution protocol** is a signaling protocol that can perform the following tasks:

- discover LSR peers
- initiate and maintain LDP session
- signal label request
- advertise binding
- signal label withdrawal
- loop prevention
- explicit routing
- resource reservation



Label Distribution Protocols

Label distribution can be performed using various protocols

There are presently the following options:

- Management protocols
- LDP
  - MPLS-enhanced IP networks
  - used as basis for **PWE3 control protocol**
- BGP4-MPLS
  - mainly for RFC 4364 VPNs
- RSVP-TE
  - traffic engineering support
- CR-LDP
  - constraint based (no longer recommended by IETF)

MPLS-TP

Background

IP is the most popular packet-switched protocol

MPLS and Ethernet are the most popular server layers under IP
but neither is a *transport* network

At least some Service Providers want a

- packet-based transport network
- similar to present transport networks
- optimized for carrying IP

Characteristics of transport networks

1. High availability
   - **F**ault **M**anagement OAM
   - **A**utomatic **P**rotection **S**witching
2. Efficient utilization, SLA support, and QoS mechanisms
   - high determinism
   - **C**onnection **O**riented behavior
   - **P**erformance **M**anagement OAM
3. Management plane (optionally control plane)
   - configuration management similar to traditional
   - efficient provisioning of p2p, p2m and m2m services
4. Scalability - must scale well with increase in
   - end-points
   - services
   - bandwidth

Possible solutions

There are two popular server network protocols for carrying IP

- Ethernet
- MPLS

(in the past there were ATM, frame relay, IP over SDH, etc.)

Extensions to both were proposed :

- **P**rovider **B**ackbone **T**ransport (which became PBB-TE)
- **T**ransport**-MPLS** (which became MPLS-TP)

PBT advanced in IEEE standardization (802.1ah + 802.1Qay)
but is now dead in the market

MPLS-TP was developed by both the IETF and the ITU-T
which eventually led to 2 incompatible versions

PBT and T-MPLS

PBT was invented by engineers at BT and Nortel

- standardization attempted at the IETF
- standardization attempted at the ITU
- standardization succeeded at the IEEE

PBT uses the regular Ethernet encapsulation, but

- turns off Ethernet learning, aging, flooding, STP
- requires use of Y.1731 Ethernet OAM, APS, etc.
- uses management plain to set up CO connections (SDH-like)
- supports client/server layering through use of MAC-in-MAC

T-MPLS was invented by Alcatel

- standardization performed at the ITU (SG13/SG15)

T-MPLS is a derivative of MPLS, but

- does not require IP
- does not require a control plane
- has ITU style OAM and APS
- uses management plain to set up CO connections (SDH-like)

MPLS-TP

MPLS-TP is a *profile* of MPLS, that is

- it reuses existing MPLS standards
- its data plane is a (minimal) subset of the full MPLS data plane
- it interoperates with existing MPLS (and PWE) protocols
  without gateways

TP is similar to other transport networks (*including look and feel*)

TP is multi-vendor (in a single domain and between domains)

TP supports static provisioning via management plane
  a control plane is defined but **not mandatory** to use

TP networks can be configured and operate **w/o IP forwarding**

TP's data plane is physically/logically separated from management/control planes

TP adds OAM and APS functionality to MPLS

The OAM issue

Since it strives to be a carrier-grade transport network
TP has strong OAM requirements
OAM has been the most contentious issue in standardization

It is agreed that OAM will be generally in the GACh

But two different OAM protocols break MPLS-TP into two distinct flavors

1. IETF bases its OAM on **B**idirectional **F**orwarding **D**etection
   BFD is a "hello" protocol originally between routers
   before TP IETF standardized it for IP, MPLS, and PWs (in VCCV)

2. ITU bases its OAM on Ethernet OAM **Y.1731** (802.1ag)

The mechanisms can not interoperate

The APS issue

MPLS-TP requires linear and ring protection mechanisms

Similar to what happened in OAM, the IETF and ITU developed different APS

The ITU adapted Ethernet APS mechanisms to MPLS

The IETF developed new mechanisms with the same functionality

The mechanisms can not interoperate

Control and Management

Every MPLS-TP network element must connect (directly or indirectly) to an **O**perations **S**ystem

When the connection is indirect, there must be a **M**anagement **C**ommunication **C**hannel

When there is a control plane, there is also a **S**ignaling **C**ommunication **C**hannel

TP management plane functionality includes:

- configuration management (of system, CP, paths, OAM, APS)
- fault management (supervision, validation, alarm handling)
- performance management (characterization, measurement)
- security management

TP defines a control plane (but it is not mandatory to use)

- for setting up LSPs MPLS-TP uses
  - RSVP-TE and extensions
  - OSPF-TE (RFC 4203 and 5392) or ISIS-TE
- for setting up PWs MPLS-TP uses the PWE3 control protocol RFC4447

Identifiers

In order to configure, manage, and monitor network elements
they require unique identifiers

In IP networks, IP addresses serve as a unique identifiers
but MPLS-TP must function *without* IP

PWs set up by PWE3 control protocol have unique identifiers
RFC 4447 defines **A**ttachment **I**ndividual **I**dentifiers

In carrier networks network elements can be uniquely identified by
Country_Code:ICC:Node_ID
Country_Code is two upper case letters defined in ISO 3166-1
ICC is a string of one to six alphabetic/numeric characters
Node_ID is a unique 32-bit unsigned integer

For MPLS-TP any of these can be used

The GACh

Generic Associated Channel

MPLS-TP must be able to forward management and control plane messages without an IP forwarding plane

MPLS-TP must be able to inject OAM messages that fate-share with the user traffic

MPLS-TP needs to send status indications

MPLS-TP must support APS protocol messages

How are all these messages sent ?

Associated channels

PWs have an **A**ssociated **Ch**annel (**ACh**)
in which one can place OAM (VCCV)
that will *fate-share* with user traffic

The ACh is defined in RFC 4385
and is based on use of the PWE3 Control Word

0 0 0 1	VER	RES=0	Channel Type

MPLS-TP also needs an ACh for its OAM
but MPLS LSPs do not have a CW!

Y.1711 defined a mechanism for MPLS (pre-TP) OAM
based on use of reserved label 14 and an OAM type code
The ITU wanted to use this mechanism for T-MPLS as well
but the IETF did something a little bit different

GACh

RFC 5586 defines the **G**eneric **A**ssociated **Ch**annel (**GACh**)
based on the **G**eneric **A**ssociated channel **L**abel (**GAL**)

For the simplest case :

<table>
<tr><td colspan="3">MPLS label</td><td colspan="2">TC</td><td>S</td><td>TTL</td><td>MPLS label stack</td></tr>
<tr><td colspan="3">GAL label = 13</td><td colspan="2">TC</td><td>S</td><td>TTL</td><td>GAL</td></tr>
<tr><td>0001</td><td>0000</td><td>RESERVED</td><td colspan="4">Channel Type</td><td>ACH header</td></tr>
<tr><td colspan="7">Zero or more ACh TLVs</td><td></td></tr>
<tr><td colspan="7">GACh message</td><td></td></tr>
</table>

What can be carried in the GACh ?

Defined Channel Types (IANA registry) :

Value	Description	TLVs	Reference
0x0000	Reserved		
0x0001	MCC	No	RFC5718
0x0002	SCC	No	RFC5718
0x0007	BFD w/o IP header	No	RFC5885
0x0021	IPv4 packet	No	RFC4385
0x0057	IPv6 packet	No	RFC4385
0x0058	Fault OAM (temporary)	No	draft-ietf-mpls-tp-fault
0x7FF8-0x7FFF	Experimental Use		RFC5586

The GACh can thus be used for:

1. OAM (FM/PM) – using BFD, Y.1731, … (see next chapter)
2. status signaling for static (non-LDP) PWs
3. management traffic (e.g., when no IP forwarding plane)
4. control traffic (e.g., when no IP forwarding plane)
5. other uses ?