# Virtual LANs

# Virtual LANs

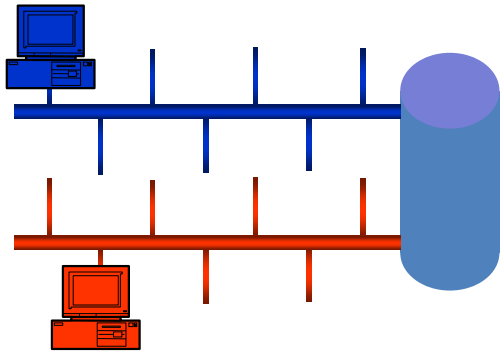
Up to now we assumed that each LAN has its own infrastructure

- 1 broadcast domain per set of cables and hubs
- all stations on LAN see all traffic

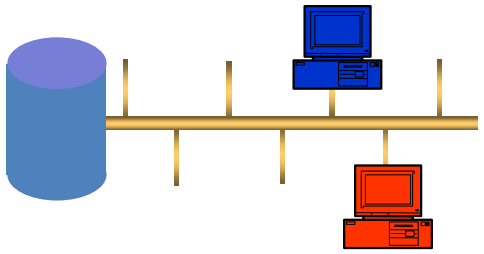
We often want a single physical infrastructure to support many LANs

- simpler and less expensive than maintaining separate infrastructures
- multiple low-speed LANs on one high-speed infrastructure
- segment broadcast domains (lower BW/processing) without routers
- security for different departments in company / groups in campus

Separation may be based on switch ports or MAC address or VLAN ID (tag)



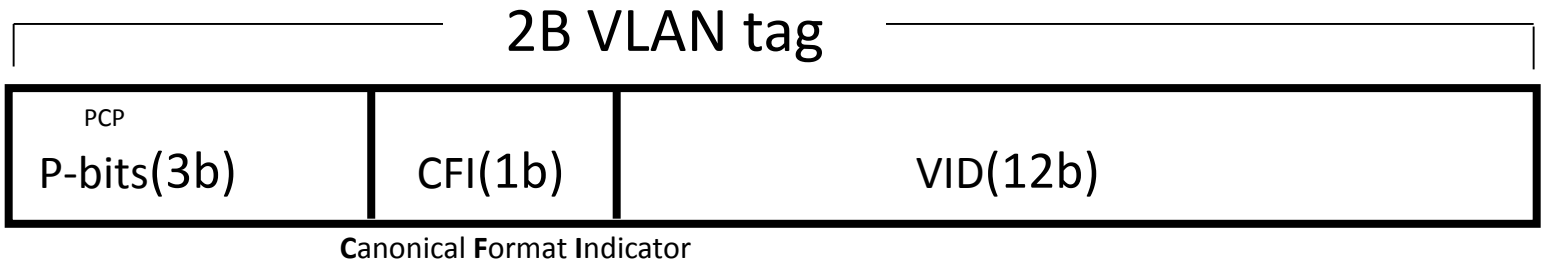
port-based VLAN



VID-based VLAN

# 802.1Q

802.1Q & 802.1p projects defined format, protocols, and procedures for VLANs
Ethertype=8100 defines a VLAN tag, and is followed by 2 bytes




- 2 bytes carry PCP (priority) bits
- CFI (not important here, always 0) replaced by **D**rop **E**ligibility **I**ndicator
- 4094 possible VID values (0 and 4095 are reserved)
- VID=0 frames are priority tagged, able to carry P bits

802.1ad and 802.1ah further extend tagging formats and procedures

# VLAN-aware switches

VLAN-aware switches

- take VID into account when forwarding
- perform VID insertion/removal
- never output a priority-tagged frame

when VLAN-aware switch receives

- VLAN tagged frame – treats according to VID
- untagged frame – may push permanent VID (PVID) of receive port
- priority-tagged frame treated like untagged frame (VLAN tag MAY be added)

Insertion / removal of VLAN tag necessitates recomputing FCS
and may require adjusting padding (minimum frame size in 802.1Q has changed)

# VLAN-aware switch operation

A VLAN-aware switch performs 5-stage processing:

- ingress rule checking
  - 2 modes: *admit only tagged*, *admit all*
  - classify every incoming frame to a VID (if untagged to PVID)
  - discard frame not obeying rules, e.g.
    - port not in VID member set
    - untagged with admit only tagged
- active topology enforcement
  - check that frame *should* be forwarded, discard if, e.g.
    - spanning tree forbids forwarding or forwarding is back to port of origin
    - port not in forwarding state
    - MTU exceeded
- frame filtering (according to MAC, VID and filtering DB entry)
- egress rule checking
  - discard if VID not in member set
  - add/remove tag as needed
- queuing for transmission

# SVL (SFD) switches

Shared VLAN Learning
(Single Forwarding
Database)

| MAC address | PORT |
|---|---|
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |

How do VIDs interact with 802.1D (forwarding and learning) ?

There are two different answers to this question

SVL switches still maintain a single 802.1D filtering database
but use VLAN in ingress/egress classification

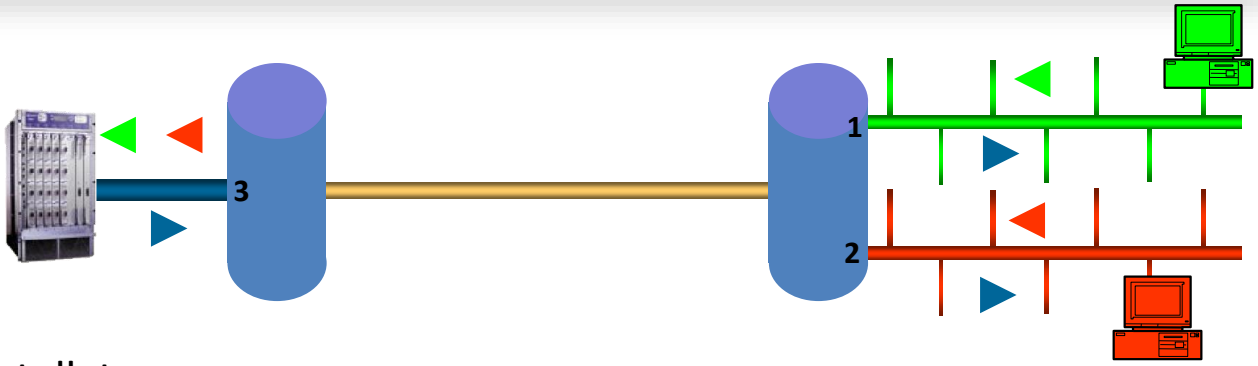
MAC addresses are learnt together for all VLANs (flood MAC once)

The path to a MAC does not depend on VID

A MAC address belongs to at most 1 VLAN

Asymmetry possible

# Asymmetry case




Green PC can talk to server

- untagged frames on port 1 are tagged with VID=1
- tags removed when sent to server

Red PC can talk to server

- untagged frames on port 2 are tagged with VID=2
- tags removed when sent to server

Server can reply to both

- untagged frames are tagged with VID=3
- tags removed when sent out of ports 1 or 2

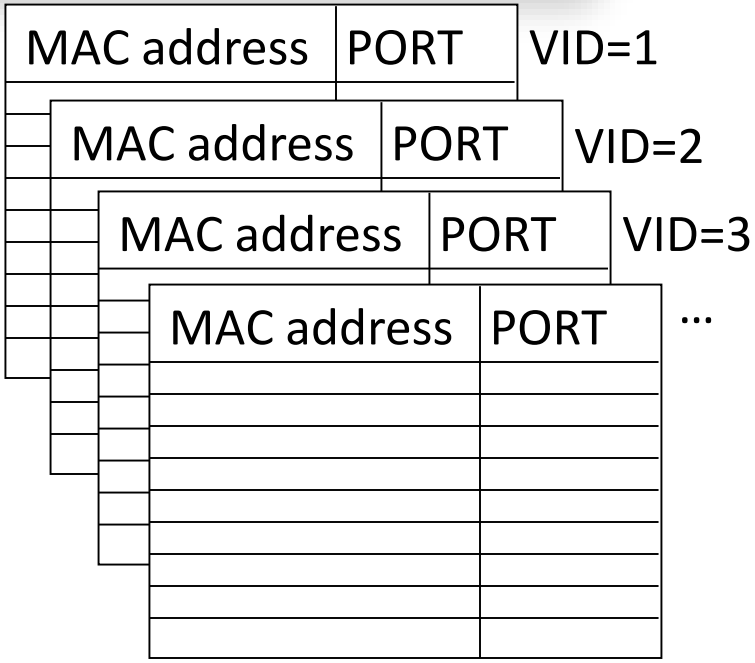
But green and red PCs can NOT talk to each other

- VID=1 traffic can not be sent to red PC with MAC associated with VID=2
- VID=2 traffic can not be sent to green PC with MAC associated with VID=1

# IVL (MFD) switches

Independent VLAN Learning

(Multiple Forwarding Databases)




IVL switches maintain separate 802.1D filtering databases per VID

MAC addresses are learnt independently for each VID (more flooding)

1 MAC address can belong to several VLANs

- but path to MAC depends on VID

asymmetry impossible

Note: IVL switch can be implemented as 60 (48+12) bit lookup

# VLAN stacking

We tag Ethernet frames by using Ethertype 8100


The first Ethertype is set to 8100

The second Ethertype is payload protocol (as in untagged frame)

but what if we add another Ethertype to 8100 ?

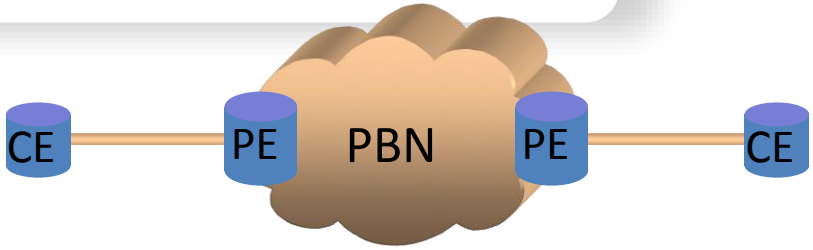

This is called VLAN stacking
or (for obvious reasons) Q-in-Q

Stacking is not mentioned in 802.1Q
but not ruled out either!

**Warning:**

although superficially Q-in-Q looks like MPLS stacking
there is no network layering here
Note that the DA remains the same!

# Provider Bridges – 802.1ad




Why stack VLANs?

The main reason is Provider Bridge Networks

Customers often use VIDs for internal reasons
and will want them maintained across the PBN

But what if multiple customers use the same VIDs ?

The provider must either:

- manage customer VIDs
  - allocate ranges to customers
  - swap at ingress and egress or
- treat them as Customer VIDs, and push a Service VID (customer ID)

802.1ad
approved Dec 2005
published May 2006

# CVID and SVID

Customer frames have C-TAGs, may only have **priority** S-TAGs

PBN frames have S-TAGs and usually have C-TAGs

- C-TAG contains a C-VID
- S-TAG contains an S-VID

C-TAGs are standard format VLAN tags

802.1ad S-TAGs are similar, but use Ethertype 88A8
since they have Drop Eligible Indicator instead of CFI

*Service transparency*

- PE inserts S-TAG, C-TAG becomes invisible to PBN
- PE removes S-TAG, C-TAG becomes visible to customer network

| | | | |
|---|---|---|---|
| 88A8 (S-TAG) | P | D<br>E<br>I | S-VID |
| 8100 (C-TAG) | P | C<br>F<br>I | C-VID |

# Problems with PBNs

Q-in-Q PBNs simplify provider-customer interface, but

- are limited to 4K SVIDs, i.e. 4K customers
  (VLANs derive from enterprise (not carrier) applications)
- do not provide true client/server (decoupled) relationship
  - customer VIDs are hidden
    but not customer MAC addresses
  - no true OAM trace functionality
- provider switches still have to learn customer MAC addresses

We would really like full decoupling, i.e. a client/server relationship

This can be done via MAC-in-MAC (true client/server network layering!)

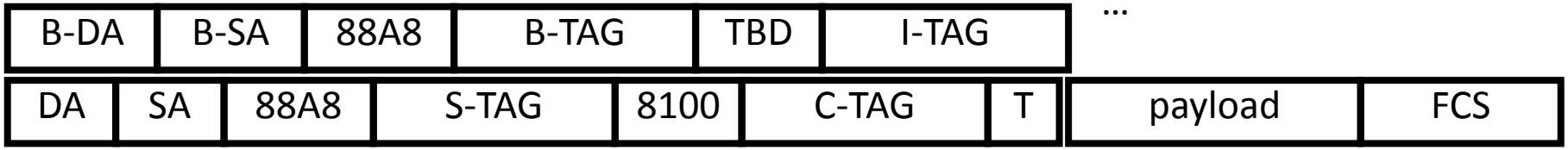
# Progress to MAC-in-MAC

802.1D

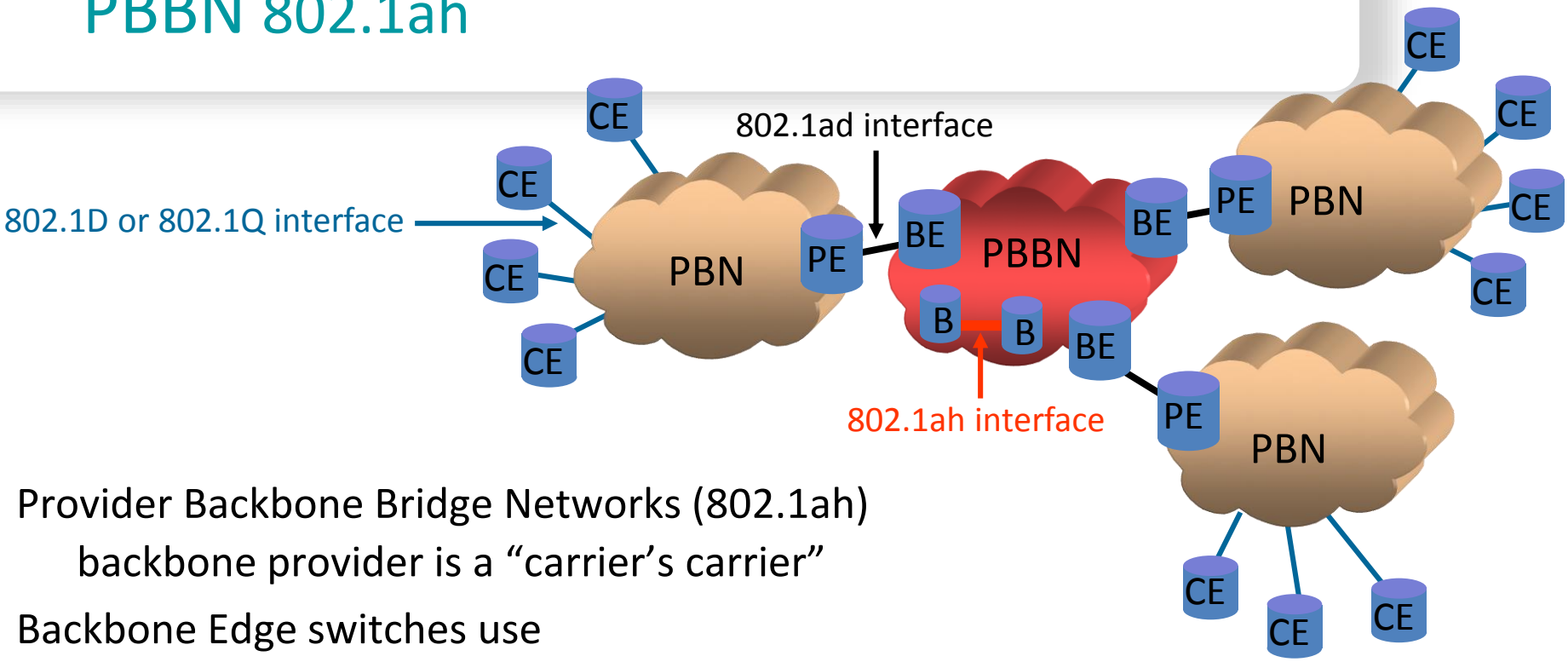

802.1Q


802.1ad


802.1ah



# PBBN 802.1ah




Provider Backbone Bridge Networks (802.1ah)

backbone provider is a "carrier's carrier"

Backbone Edge switches use

- I-TAGs with I-SID
- B-TAG with B-VID
- one or more I-TAGs and a B-TAG

I-TAG is a new format, I-SID is a 24-bit label (no more 4K limitation)

4B I-TAG

| I-P(3b) | I-DEI(1b) | RES(4b) | ISID (24b) |
|---|---|---|---|

# PBT (PBB-TE) 802.1Qay

Provider Backbone Transport builds on 802.1ah PBBNs
to achieve reliable, deterministic, carrier-class behavior

802.1ah gives backbone provider its own addressing space (MAC + VLAN)

Addresses are not shared with customer (no need to learn customer MACs)
and so provider is free to use address space as it sees fit

We saw before that IVL switches forward based on 60-bit addresses
but their capabilities limited due to limitations of STP, flooding, etc

But most IVL switches allow static FID (preconfigured forwarding behavior)
and support turning off learning/STP/flooding/aging

We can thus set up pure connection-oriented topology

We can use
- network management systems or
- control plane protocols (e.g. RSVP-TE)

to populate filtering database tables